

A
PROJECT-REPORT ON
TO BUILD DNA
NUCLEOTIDE COUNT
WEB APPLICATION

SUBMITTED BY,

MR. PRADEEP BHIMRAO GANAP

SUBMITTED TO,

VIVEKANAND COLLEGE,
DEPARTMENT OF BIOTECHNOLOGY
KOLHAPUR

FOR PARTIAL FULFILMENT OF BACHELOR OF
THE YEAR - 2023-2024

SCIENCE IN BIOTECHNOLOGY

UNDER THE GUIDANCE OF,

Miss. S. D. Potdar

Assistant Professors, Department of

Biotechnology.

"Education for Knowledge, Science and Culture"

-Dr. Bapuji Salunkhe



Shri Swami Vivekanand Shikshan Sanstha's



VIVEKANAND COLLEGE KOLHAPUR
[Empowered Autonomous]

Certificate

This is to certify that, **Mr. Pradeep Bhimrao Ganap**,
Exam Number _____, has satisfactorily completed a Project
report on "**DNA Nucleotide Count Web Application**" as a part
of syllabus prescribed by Department of Biotechnology,
Vivekanand College, Kolhapur [EMPOWERED AUTONOMOUS]
for B.Sc III course in Biotechnology (Optional) and this project
represents her bonafide work of the year 2023-24.

Place: Kolhapur

Date : 18/3/2024

Spotdaz

Project Guide

[Signature]

Examiner

[Signature]

Head of
Department

HEAD
DEPARTMENT OF BIOTECHNOLOGY (OPTIONAL)
VIVEKANAND COLLEGE, KOLHAPUR
(EMPOWERED AUTONOMOUS)



DECLARATION

I hereby declare that the project work entitled "To Build DNA Nucleotide Web Application" submitted to Vivekanand College, Kolhapur [EMPOWERED AUTONOMOUS] for the award of the degree of "Bachelor of Science in Biotechnology" is the result of bonafied work carried out by me under the guidance of **Asst. Prof. Miss. S. D. Potdar.**

I further declare that the results presented here have not been the basis for the reward of any other degree.

Place: Kolhapur

Date: 18/3/2024



Mr. Pradeep Bhimrao Ganap

ACKNOWLEDGEMENT

This project work is a successful outcome of the contribution and guidance of other person which I express my deep gratitude.

I also express our thanks to **Prof. Mrs. S. B. Mulla** Head of Department of Biotechnology (Optional), Vivekanand College, Kolhapur for availing me with the laboratory facilities to the biotechnology Department to carry experiment work.

I also express our gratitude towards **Asst Prof. Miss. S. D. Potdar** and **Asst Prof. Miss. D. A. Wajantri** as a project guide for their guidance and who gave me encouragement and support throughout the course of study so that could complete my project work.

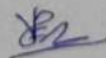
I also wish to express my gratitude to the laboratory staff for completing the project work. Lastly, I express my gratitude to my parents, all our friends and classmates for their support and co-operation. I am also grateful to all those who have directly or indirectly supported me in completion of this work.

RESULT

CONCLUSION

REFERENCES

FUTURE PROSPECTS



Mr. Pradeep Bhimrao Ganap

INDEX

SR. NO.	CONTENTS	PAGE NO.
1.	INTRODUCTION	1-11
2.	REVIEW OF LITERATURE	12-13
3.	AIM AND OBJECTIVES	14-15
4.	MATERIALS AND METHODS	16-31
5.	RESULT	32-34
6.	CONCLUSION	25-36
7.	REFERENCES	37-39
8.	FUTURE PROSPECTS	40-41

INTRODUCTION

1. BIOINFORMATICS

The term "Bioinformatics" was invented by Paulein Hogeweg and Ben Hesper in 1970 as "the study of informatic processes in biotic systems.

It is a hybrid science that links biological data with techniques for information storage, distribution and analysis to support multiple areas of scientific research, including biomedicine. It is fed by high-throughput data generating experiments, including genomic sequence determinations and measurements of gene expression patterns.

The classic data of bioinformatics include DNA sequences of genes or full genomes; amino acid sequences of protein; and three-dimensional structures of proteins, nucleic acids and protein-nucleic acid complexes.

In bioinformatics, data banks are used to store and organize data. Many of these entities collect DNA and RNA sequences from scientific papers and genome projects.

The development of efficient algorithms for measuring sequence similarity is an important goal of bioinformatics. Another goal of bioinformatics is the extension of experimental data by predictions. A fundamental goal of computational biology is the prediction of protein structure from an amino acid sequence.

APPLICATIONS: -

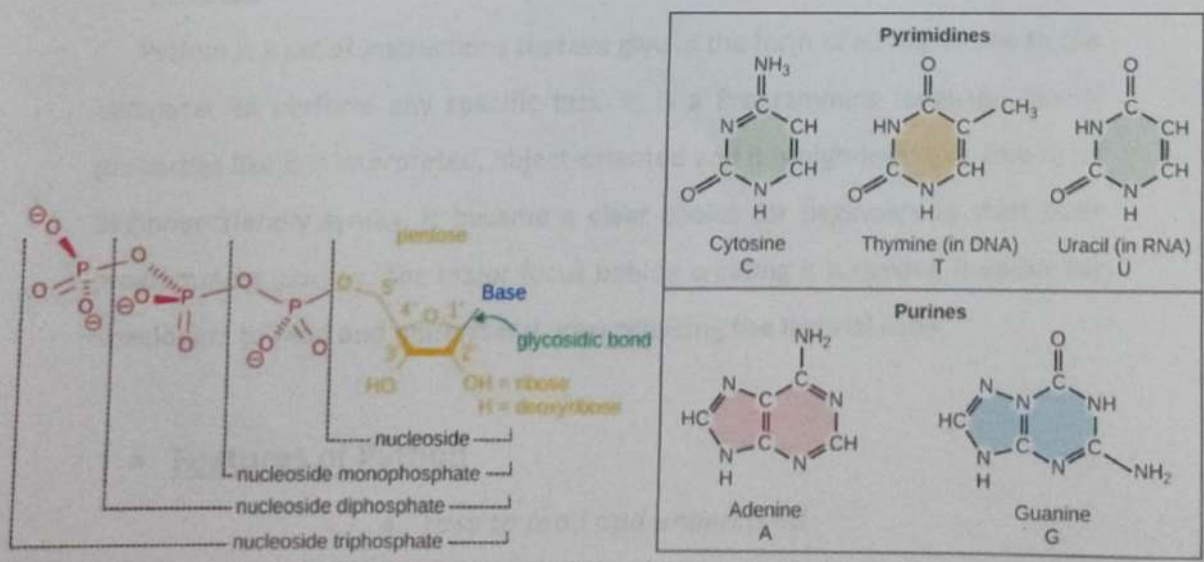
It has application in various science fields and some of the examples are:

It is used in molecular medicine, drug development, gene therapy, biotechnology, bio weapon creation, forensic analysis, veterinary science, personalized medicine, etc....

2. Nucleotides

A molecule that is the basic building block of the nucleic acids DNA and RNA. A nucleotide is made up of a nitrogen-containing base (adenine, guanine, thymine, and cytosine in DNA, and adenine, guanine, uracil, and cytosine in RNA), a phosphate group, and a sugar molecule (deoxyribose in DNA, and ribose in RNA). DNA and RNA are polymers made up of many nucleotides.

Structure of DNA. Most DNA is found inside the nucleus of a cell, where it forms the chromosomes. Chromosomes have proteins called histones that bind to DNA. DNA has two strands that twist into the shape of a spiral ladder called a helix. DNA is made up of four building blocks called nucleotides: adenine (A), thymine (T), guanine (G), and cytosine (C). The nucleotides attach to each other (A with T, and G with C) to form chemical bonds called base pairs, which connect the two DNA strands. Genes are short pieces of DNA that carry specific genetic information.



3. Web Application

A web application (web app) is an application program that is stored on a remote server and delivered over the internet through a browser interface. Web services are web apps by definition and many, although not all, websites contain web apps.

Web applications do not need to be downloaded since they are accessed through a network. Users can access a web application through a web browser, such as Google Chrome, Mozilla Firefox or Safari.

For a web app to operate, it needs a web server, application server and database. Web servers manage the requests that come from a client, while the application server completes the requested task. A database stores any necessary information.

Some of the tools we used to build Web Application are Google Colab, Anaconda Navigator, Streamlit, Streamlit Local Tunnel for Google Colab etc.

4. Google Colab

➤ Python

Python is a set of instructions that we give in the form of a Programme to our computer to perform any specific task. It is a Programming language having properties like it is interpreted, object-oriented and it is high-level too. Due to its beginner-friendly syntax, it became a clear choice for beginners to start their programming journey. The major focus behind creating it is **making it easier for developers to read and understand, also reducing the lines of code.**

• Features of Python

- a. *Easy to read and understand*
- b. *Interpreted language*

- c. *Object-oriented programming language*
- d. *Free and open-source*
- e. *Free and open-source*
- f. *Versatile and Extensible*
- g. *Multi-platform*
- h. *Hundreds of libraries and frameworks*
- i. *Flexible, supports GUI*
- j. *Dynamically typed*
- k. *Huge and active community*

- **Python's Module, Package and Library**

What is Module in Python?

The module is a simple Python file that contains collections of functions and global variables and with having a `.py` extension file. It is an executable file and to organize all the modules we have the concept called Package in Python.

Examples of modules:

- i. Datetime
- ii. Regex
- iii. Random etc.

Example: Save the code in a file called `demo_module.py`

What is Package in Python?

The package is a simple directory having collections of modules. This directory contains Python modules and also having `__init__.py` file by which the interpreter interprets it as a Package. The package is simply a namespace. The package also contains sub-packages inside it.

Examples of Packages:

- i. Numpy
- ii. Pandas (used in project)

What is Library in Python?

The **library** is having a collection of related functionality of codes that allows you to perform many tasks without writing your code. **It is a reusable chunk of code that we can use by importing it into our program**, we can just use it by importing that library and calling the method of that library with a period(.). However, it is often assumed that while a package is a collection of modules, a library is a collection of packages.

Examples of Libraries:

- i. Pytorch
- ii. Matplotlib
- iii. Pygame
- iv. Seaborn
- v. Sreamlit (used in project)
- vi. Altair (used in project)

Example:

Importing pandas library and call read_csv method using an alias of pandas i.e. pd.

➤ Google Colab :

- Google Colaboratory, or Colab, is an as-a-service version of Jupyter Notebook that enables you to write and execute Python code through your browser.
- Jupyter Notebook is a free, open source creation from the Jupyter Project. A Jupyter notebook is like an interactive laboratory notebook that includes not just notes and data, but also code that can manipulate the data. The code can be executed within the notebook, which, in turn, can capture the code output. Applications such as Matlab and Mathematica pioneered this model, but unlike those applications, Jupyter is a browser-based web application.

- Google Colab is built around Project Jupyter code and hosts Jupyter notebooks without requiring any local software installation. But while Jupyter notebooks support multiple languages, including Python, Julia and R, Colab currently only supports Python.
- Colab notebooks are stored in a Google Drive account and can be shared with other users, similar to other Google Drive files. The notebooks also include an autosave feature, but they do not support simultaneous editing, so collaboration must be serial rather than parallel.
- Colab is free, but has limitations. There are some code types that are forbidden, such as media serving and crypto mining. Available resources are also limited and vary depending on demand, though Google Colab offers a pro version with more reliable resourcing. There are other cloud services based on Jupyter Notebook, including Azure Notebooks from Microsoft and SageMaker Notebooks from Amazon.



Google Colaboratory

- **Why Should You Use Google Colab?**

Google Colab provides many exciting features that any modern IDE offers, and much more. Some of the most exciting features are listed below.

- **Pre-installed libraries:**

Google Colab comes pre-installed with the most popular machine-learning libraries. Colab comes pre-installed with Keras, PyTorch, and TensorFlow, which saves you the time and hassle of setting up a local environment.

- **Saved on the cloud:**

Every Notebook you create in the Google Google Colab is saved on the cloud. This lets you access and work with those Notebooks from any machine. You only need a browser and a reliable network connection, and you can work from anywhere and anytime.

- **Collaboration :**

Collaboration is another fantastic reason to choose Google Google Colab when you are working on a project with a team of developers. You can share your Notebook with your teammates and assign them roles so they can only perform operations that fit their roles. The various options available for each role is shown below:

- Editors can change permissions and share – Viewers and commenters can see the option to download, print, and copy

- **Free GPU and TPU use:**

Google Colab provides free access to GPUs and TPUs developed by Google Research. So you can work on your projects with powerful GPUs irrespective of your local machine.

The screenshot shows the Google Colaboratory web interface. At the top, it says 'Welcome To Colaboratory' with a 'Sign In' button. Below that is a menu bar with 'File', 'Edit', 'View', 'Insert', 'Runtime', 'Tools', and 'Help'. There are also buttons for '+ Code', '+ Text', and 'Copy to Drive'. The main content area has a title 'What is Colaboratory?' and the following text:

Colaboratory, or "Colab" for short, allows you to write and execute Python in your browser, with

- Zero configuration required
- Free access to GPUs
- Easy sharing

Whether you're a student, a data scientist or an AI researcher, Colab can make your work easier. Watch [Introduction to Colab](#) to learn more, or just get started below!

Getting started

The document you are reading is not a static web page, but an interactive environment called a **Colab notebook** that lets you write and execute code.

For example, here is a **code cell** with a short Python script that computes a value, stores it in a variable, and prints the result:

```
[ ] seconds_in_a_day = 24 * 60 * 60
seconds_in_a_day

86400
```

To execute the code in the above cell, select it with a click and then either press the play button to the left of the code, or use the keyboard shortcut "Command/Ctrl+Enter". To edit the code, just click the cell and start editing.

5. Streamlit

Streamlit is a powerful and user-friendly open-source Python library that makes it easier to build interactive web applications for machine learning and data science. With Streamlit, developers and data scientists can create engaging, informative, and visually appealing apps with just a few lines of code.

To install streamlit and import :

```
!pip uninstall streamlit

import streamlit as st
import pandas as pd
import matplotlib.pyplot as plt
```

One of the main benefits of Streamlit is its simplicity. You can write your code in a familiar environment and use the library's high-level APIs to quickly build complex and interactive web applications. Whether you're a seasoned software engineer or a beginner in data science, Streamlit makes it easy to get started

Streamlit also offers a wide range of tools for creating data visualizations. Whether you want to plot a scatterplot, display a histogram, or create a map, Streamlit has you covered. You can even use it to build interactive dashboards that allow users to explore your data in real-time.

Another benefit of Streamlit is its flexibility. You can use it to build apps for a wide range of use cases, from simple data visualizations to complex machine learning models. Streamlit even supports deployment, so you can easily share your apps with others.

This code will create a web app that displays a line graph of x and y :

```
st.title("Line Plot Example")
data = pd.DataFrame({
    "x": [1, 2, 3, 4], # Load sample data
    "y": [10, 20, 30, 40]})
plt.plot(data["x"], data["y"]) # Plot the data
st.pyplot() # Show the plot in the Streamlit app
```

When you run this code, Streamlit will create a new web page that you can interact with. You can pan and zoom the plot, and you can use the Streamlit app to explore your data in real-time.

This example demonstrating some of Streamlit's unique features, including :

- The slider component, slider allows you to choose a value within a specified range by dragging a handle along a track. You can use sliders to select a number, a date, or any other type of value.

```
num_points = st.slider("Number of points", min_value=100, max_value=1000,
value=500, step=100)
```

- The text input component, which updates in real-time as the user types

```
text_input = st.text_input("Enter some text:")
st.write("The name :", text_input)
```

- The checkbox component, a Checkbox is a toggle switch that allows the user to turn an option on or off. You can use Checkboxes to allow the user to select multiple options.

```
show_plot = st.checkbox("Show plot", value=True)
if not show_plot:
    plt.close()
```

- The Select box component, a SelectBox allows you to choose one option from a dropdown list. You can use Select Box to present a list of options for the user to choose from.

```
plot_color = st.selectbox("Plot color", ["blue", "red", "green"])
plt.plot(data["x"], data["y"], color=plot_color)
```

- The camera input helps to take a photo on the site . comes in handy when deploying and using computer vision models .

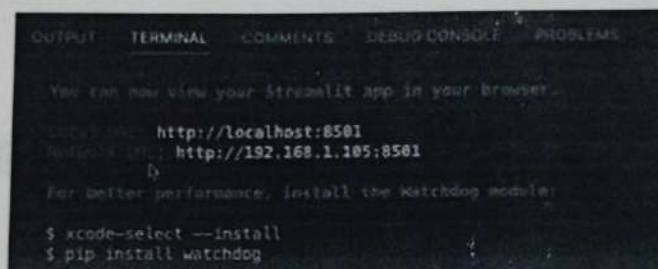
```
st.camera_input("Take a picture")
```

- You can display a progress bar in a Streamlit app . You can use this to display the progress of a long-running task, such as a download or data processing.

```
st.progress(progress_variable_1_to_100)
```

When everything done , this is how you launch it .

```
streamlit run app.py
```



```
OUTPUT  TERMINAL  COMMENTS  DEBUG CONSOLE  PROBLEMS

You can now view your Streamlit app in your browser.

Local URL: http://localhost:8501
Network URL: http://192.168.1.105:8501

For better performance, install the Watchdog module:

$ xcode-select --install
$ pip install watchdog
```

Take the local URL and put it in chrome . There you go ,now you have your streamlit web app up and running .

6. Github

GitHub is a developer platform that allows developers to create, store, manage and share their code. It uses Git software, providing the distributed version control of Git plus access control, bug tracking, software feature requests, task management, continuous integration, and wikis for every project. Headquartered in California, it has been a subsidiary of Microsoft since 2018.

It is commonly used to host open source software development projects. As of January 2023, GitHub reported having over 100 million developers and more than 420 million repositories, including at least 28 million public repositories. It is the world's largest source code host as of June 2023.

Review of literature

- Chaninm Nantasenamat - [github.com/ dataprofessor/ code/ tree – master - streamlit - part8](https://github.com/dataprofessor/code/tree/master/streamlit-part8) – 30/09/2020
- Oliwier Szymański - [github.com/ oliszymanski/ Count-DNA-Nucleotides/blob/ main/ README.md](https://github.com/oliszymanski/Count-DNA-Nucleotides/blob/main/README.md) – 12/08/2021
- rohan007 - [medium.com/ web-app-8eb996440904](https://medium.com/web-app-8eb996440904), medium blog - Aug10, 2021
- Taofeek Abiodun Olasunkanmi Yusuf - [github.com/ taofeekaoyusuf – dna – nucleotide – count – web -app/ blob/ main/ dnanucleotidecountapp.py](https://github.com/taofeekaoyusuf/dna-nucleotide-count-web-app/blob/main/dnanucleotidecountapp.py) – (2021)
- Grace Hephzibah - [devpost.com/ software – dna - nucleotide – app](https://devpost.com/software-dna-nucleotide-app) – (2021)

AIM AND OBJECTIVES

1. To Build DNA Nucleotide Count Web Application using Google Colab.
2. The Main Objectives of these App is to determine the precise order of the four nucleotide bases – adenine, guanine, cytosine and thymine - that make up a strand of DNA.
3. To Count the amount of nucleotides (in a specific sequence).
4. To construct the result in form of table, series and in plot frequency.
5. These bases provide the underlying genetic basis (the genotype) for telling a cell what to do, where to go and what kind of cell to become (the phenotype). Nucleotides are not the only determinants of phenotypes, but are essential to their formation. Each individual and organism has a specific nucleotide base sequence.

MATERIAL AND METHODS

MATERIALS:

- Google Colab Account :- <https://colab.research.google.com/>
- Python Compiler :- <https://onecompiler.com/python> or <https://www.online-python.com/>
- GitHub Account for code repository : <https://github.com/>
- Streamlit Account to deploy App : <https://share.streamlit.io/>
- NCBI Link for getting DNA sequences of any species :- <https://www.ncbi.nlm.nih.gov/> or random dna sequence generator :- molbiotools.com/randomsequencegenerator.php
- Computer with Internet Access

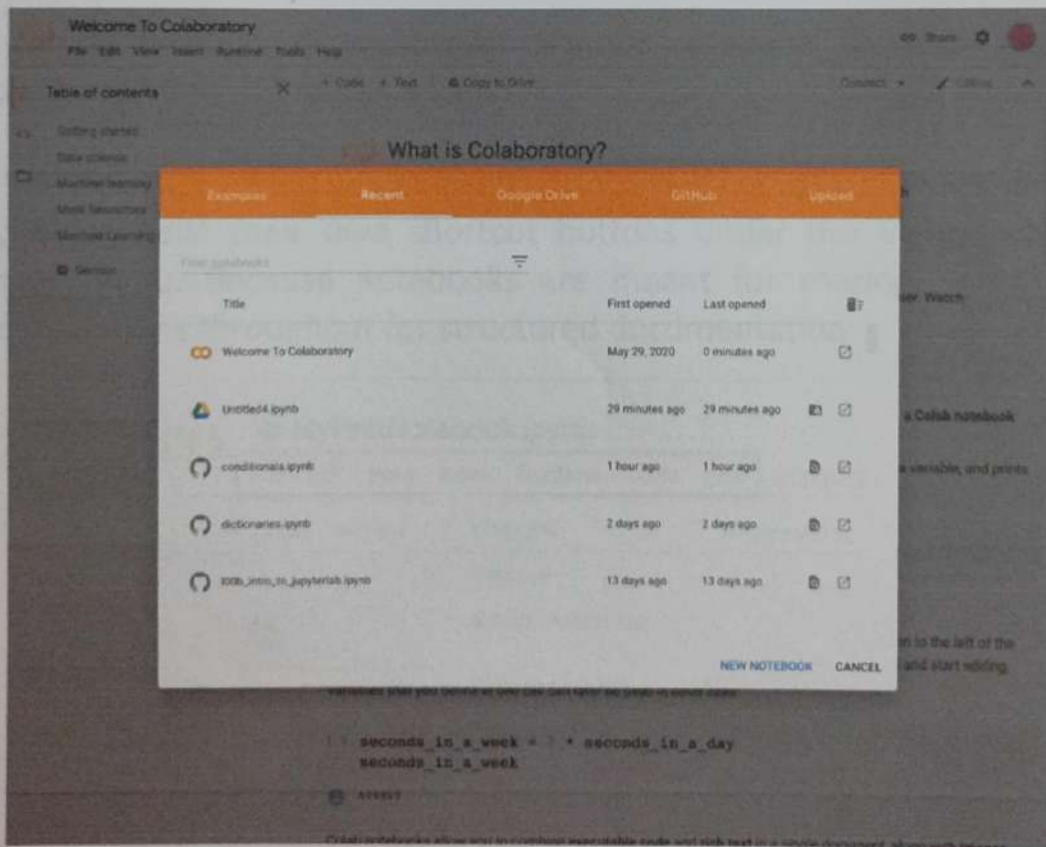
METHODOLOGY

1. First step to open the google colab account.

How to use Colaboratory

To use Colaboratory, you must have a Google account.

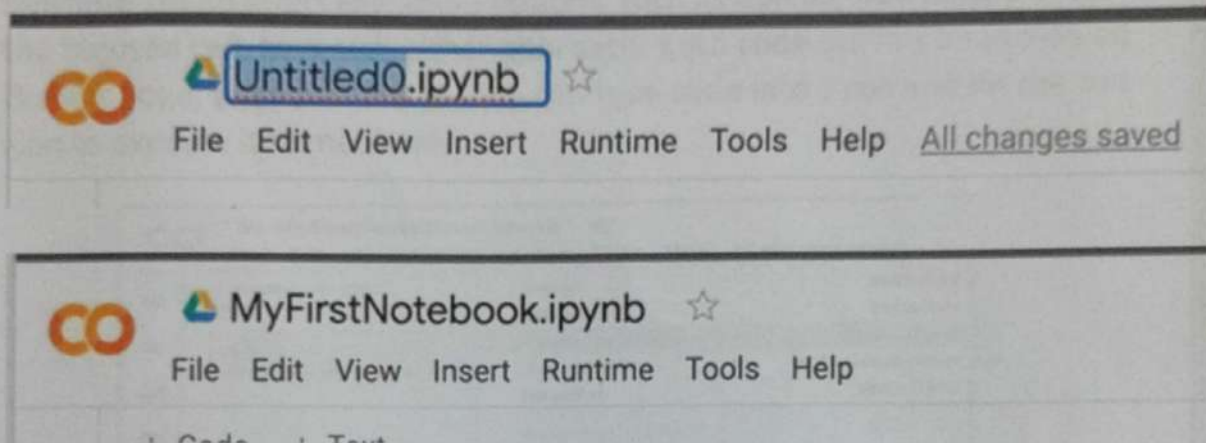
On first we to visit the google colab website - <https://colab.research.google.com/>, to create an account and the we will see a **Welcome To Colaboratory** notebook with links to video introductions and basic information on how to use Colab.



Create a workbook

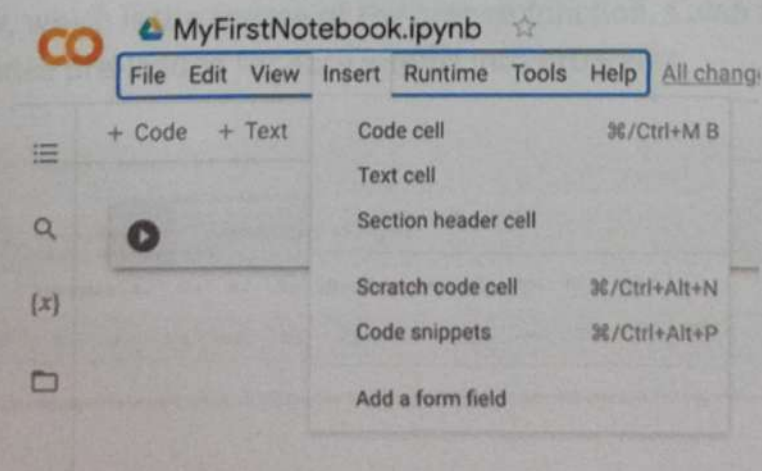
From the **File** menu, click **New Notebook** to create a workbook.

If you are not yet logged in to a Google account, the system will prompt you to log in. The notebook will by default have a generic name; click on the filename field to rename it.



The file type, IPYNB, is short for "IPython notebook" because IPython was the forerunner of Jupyter Notebook.

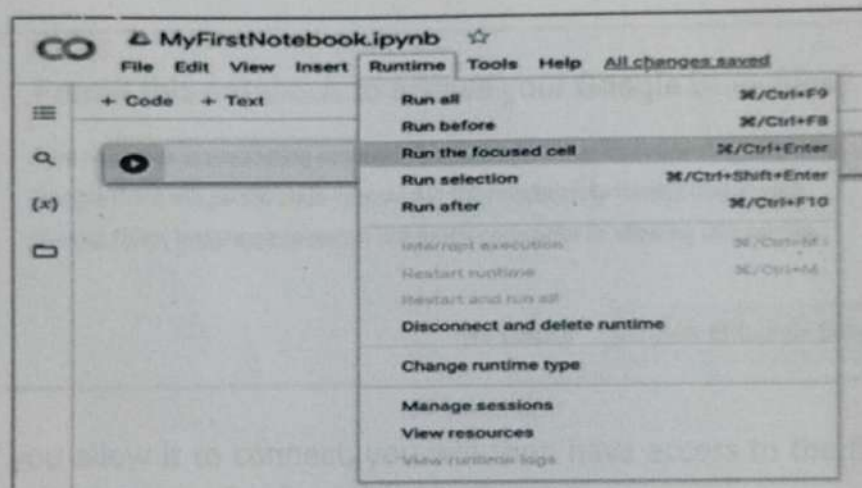
The interface allows you to insert various kinds of cells, mainly text and code, which have their own shortcut buttons under the menu bar via the **Insert** menu. Because notebooks are meant for sharing, there are accommodations throughout for structured documentation.



Code, debug, repeat

You can insert Python code to execute in a code cell. The code can be entirely standalone or imported from various Python libraries.

A notebook can be treated as a rolling log of work, with earlier code snippets being no longer executed in favor of later ones, or treated as an evolving set of code blocks intended for ongoing execution. The **Runtime** menu offers execution options, such as **Run all**, **Run before** or **Run the focused cell**, to match either approach. Each code cell has a run icon on the left edge, as shown above. You can type code into a cell and hit the run icon to execute it immediately.



If the code generates an error, the error output will appear beneath the cell. Correcting the problem and hitting run again replaces the error info with program output. The first line of code, in its own cell, imports the NumPy library, which is the source of the arange function. Colab has many common libraries pre-loaded for easy import into programs.

```
[2] import numpy as np

[5] test_array = np.arange(0,111,3)
test_array[:11]

array([ 0,  3,  6,  9, 12, 15, 18, 21, 24, 27, 30])
```

Incorporating data into the notebook

After getting comfortable with the interface and using it for initial test coding, you must eventually provide the code with data to analyze or otherwise manipulate. Colab can mount a user's Google Drive to the VM hosting their notebook using a code cell.

```
from google.colab import drive
drive.mount('/my_drive')
```

Once you hit run, Google will ask for permission to mount the drive.

Permit this notebook to access your Google Drive files?

This notebook is requesting access to your Google Drive files. Granting access to Google Drive will permit code executed in the notebook to modify files in your Google Drive. Make sure to review notebook code prior to allowing this access.

No thanks

Connect to Google Drive

If you allow it to connect, you will then have access to the files in your Google Drive via the `/my_drive` path.

Save and share

By default, Colab puts notebooks in a **Colab Notebooks** folder under **My Drive** in Google Drive.

My Drive > Colab Notebooks ▾

Type ▾

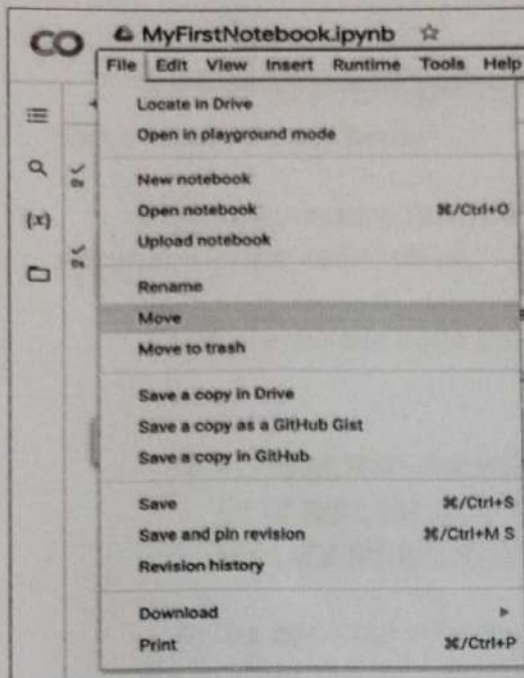
People ▾

Modified ▾

[\(Send feedback to Google\)](#)

Name

MyFirstNotebook.ipynb



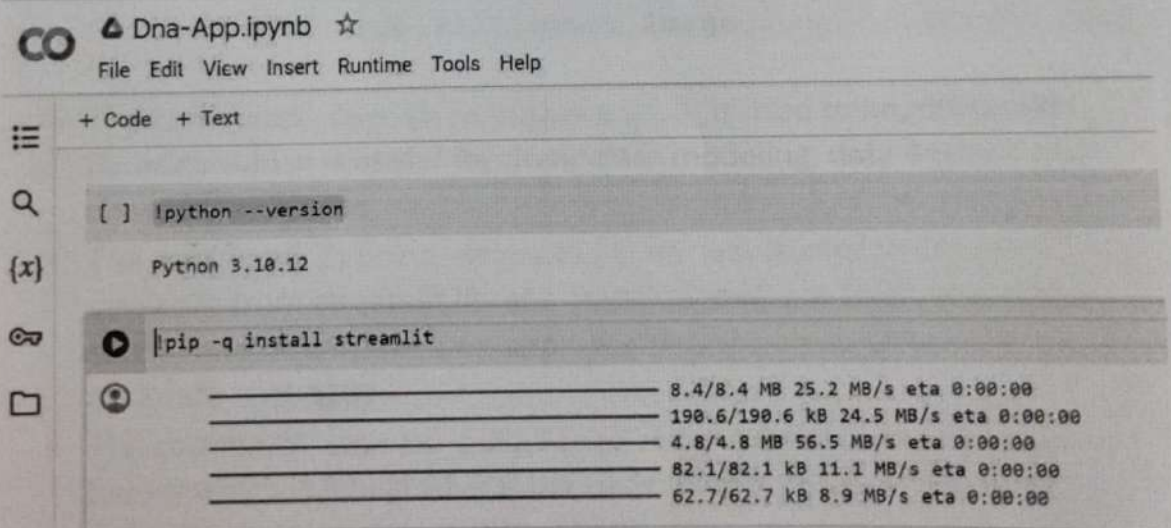
The **File** menu enables notebooks to be saved as named revisions in the version history, relocated using **Move**, or saved as a copy in Drive or GitHub. It also allows you to download and upload notebooks. Tools based on Jupyter provide broad compatibilities, so you can create notebooks in one place and then upload and use them in another.

You can use the **Share** button in the upper right to grant other Google users access to the notebook and to copy links.

2. Now the second step is to download required libraries in the google colab which are not preinstalled. i.e Streamlit.

For confirmation we will check the python version present on google colab. By importing dash command in code tab - `!python --version`

The result is as follows :



The Python version present is Python 3.10.12, and the notebook is named as Dna-App.ipynb.

Now to Download the Streamlit Library, we have import pip command in the code tab i.e, `!pip -q install streamlit`

When we run the code the library will get successfully installed.

3. In these step, we will construct the main framework of our app i.e to type the code by importing required libraries/packages that are required for our project from the google colab.

In the code tab – Firstly we have to create the environment i.e .py file to store the all imported libraries and modules (codes) which will act as main framework to build our app.

These is done by typing the line in the code tab :-

```
%%writefile dna-app.py
```

Now the second step is to import required libraries for our app to work in the same code tab :-

```
import pandas as pd
import streamlit as st
import altair as alt
from PIL import Image
```

- The command `import pandas as pd` is used to import pandas libraries which is useful for doing data modeling, data analysis and data manipulation.
- The command `import streamlit as st` is used to import streamlit from streamlit library. Streamlit helps to transform Python scripts into interactive web apps. Build dashboards, generate reports, or create chat apps.
- The command `import altair as alt` is used to import altair library which is declarative visualization library for python, used for data visualization or to display charts.

- The Command `from PIL import Image`, the PIL stands for Python Image Library is used to import the image module from the PIL library to load the image.

After importing all the libraries and setting a particular environment, now we can write a code in the same code tab :

```
image = Image.open('dna-logo.jpg')

st.image(image, use_column_width=True)

st.write("""
# DNA Nucleotide Count Web App

This app counts the nucleotide composition of query DNA!

***
""")

st.header('Enter DNA sequence')

sequence_input = ">DNA Query
\nGAACACGTGGAGGCAAACAGGAAGGTGAAGAAGAAGCTTATCCTATCAGGACGGAAG
GTCCTGTGCTCGGG\nATCTTCCAGACGTCGCGACTCTAAATTGCCCCCTCTGAGGTC
AAGGAACACAAGATGGTTTTGGAAATGC\nTGAACCCGATACATTATAACATCACCAG
CATCGTGCCTGAAGCCATGCCTGCTGCCACCATGCCAGTCCT"

sequence = st.text_area("Sequence input", sequence_input,
height=250)
sequence = sequence.splitlines()
sequence = sequence[1:]
sequence = ''.join(sequence)

st.write("""
***
""")

st.header('INPUT (DNA Query)')
sequence

st.header('OUTPUT (DNA Nucleotide Count)')
```

```

st.subheader('1. Print dictionary')
def DNA_nucleotide_count(seq):
    d = dict([
        ('A', seq.count('A')),
        ('T', seq.count('T')),
        ('G', seq.count('G')),
        ('C', seq.count('C'))
    ])
    return d

X = DNA_nucleotide_count(sequence)

X

st.subheader('2. Print text')
st.write('There are ' + str(X['A']) + ' adenine (A)')
st.write('There are ' + str(X['T']) + ' thymine (T)')
st.write('There are ' + str(X['G']) + ' guanine (G)')
st.write('There are ' + str(X['C']) + ' cytosine (C)')

st.subheader('3. Display DataFrame')
df = pd.DataFrame.from_dict(X, orient='index')
df = df.rename({0: 'count'}, axis='columns')
df.reset_index(inplace=True)
df = df.rename(columns = {'index': 'nucleotide'})
st.write(df)

st.subheader('4. Display Bar chart')
p = alt.Chart(df).mark_bar().encode(
    x='nucleotide',
    y='count'
)
p = p.properties(
    width=alt.Step(80) # controls width of bar.
)
st.write(p)

```

*The above all code is used specifically to draw a web app with a page leading with image at starting i.e image = Image.open('dna-logo.jpg')

```

st.image(image, use_column_width=True)
*and then we have a intro page which states the app description i.e
st.write("""

```

1-

```
# DNA Nucleotide Count Web App
```

```
This app counts the nucleotide composition of query DNA!
```

```
***
```

```
""")
```

*The the next we have to add dna sequence input bar i.e

```
st.header('Enter DNA sequence')
```

```
sequence_input = ">DNA Query
```

```
\nGAACACGTGGAGGCAAACAGGAAGGTGAAGAAGAAGTATCCTATCAGGACGGAAG
GTCCTGTGCTCGGG\nATCTTCCAGACGTCGCGACTCTAAATTGCCCCCTCTGAGGTC
AAGGAACACAAGATGGTTTTGGAAATGC\nTGAACCCGATACATTATAACATCACCAG
CATCGTGCCTGAAGCCATGCCTGCTGCCACCATGCCAGTCCT"
```

```
sequence = st.text_area("Sequence input", sequence_input,
height=250)
```

```
sequence = sequence.splitlines()
```

```
sequence = sequence[1:]
```

```
sequence = ''.join(sequence)
```

```
st.write("""
```

```
***
```

```
""")
```

*The next code is about the DNA input Query i.e st.header('INPUT

```
(DNA Query)')
```

```
sequence
```

*Next is the print directory which is the actual output of our sequence, i.e

```
A,T,G,C count - st.header('OUTPUT (DNA Nucleotide Count)')
```

```
st.subheader('1. Print dictionary')
```

```
def DNA_nucleotide_count(seq):
```

```
    d = dict([
```

```
        ('A', seq.count('A')),
```

```
        ('T', seq.count('T')),
```

```
        ('G', seq.count('G')),
```

```
        ('C', seq.count('C'))
```

```
    ])
```

```
    return d
```

```
X = DNA_nucleotide_count(sequence)
```

```
X
```

```
1-
```

*Next is the Print text –

```
st.subheader('2. Print text')
st.write('There are ' + str(X['A']) + ' adenine (A)')
st.write('There are ' + str(X['T']) + ' thymine (T)')
st.write('There are ' + str(X['G']) + ' guanine (G)')
st.write('There are ' + str(X['C']) + ' cytosine (C)')
```

*The display data frame, which is in the table form –

```
st.subheader('3. Display DataFrame')
df = pd.DataFrame.from_dict(X, orient='index')
df = df.rename({0: 'count'}, axis='columns')
df.reset_index(inplace=True)
df = df.rename(columns = {'index':'nucleotide'})
st.write(df)
```

*The display bar chart, which display the results in th graph format –

```
st.subheader('4. Display Bar chart')
p = alt.Chart(df).mark_bar().encode(
    x='nucleotide',
    y='count'
)
p = p.properties(
    width=alt.Step(80) # controls width of bar.
)
st.write(p)
```

*After writing the above code in one code tab, we have to run that code tab, by which the .py file i.e [dna-app.py](#) will get generated and saved in the files menu.

The screenshot shows the JupyterLab interface. On the left, the 'Files' panel shows a directory structure with 'sample_data' and 'dna-app.py'. A white arrow points to 'dna-app.py'. The main area shows a code cell with the following code:

```
%%writefile dna-app.py
import pandas as pd
import streamlit as st
import altair as alt
from PIL import Image
```

Below the code cell, the output shows the file 'dna-app.py' has been created. The status bar at the bottom indicates '0s completed at 5:11 PM'.

*Now to check whether the written code/dna-app.py work or not, it is checked by the command for streamlit and localtunnel –
`!streamlit run dna-app.py & npx localtunnel --port 8501`

*After successful run of the code we will get the supporting link for the app we created that is temporary displayed until the code run in google colab. The localtunnel link which is provided helps us to display the app prior the app is been hosted publically

The main link is provided along with the Network and the External URL. The External URL is important and copied (except the last four digit and the colon symbol) and entered in the main link which will act as password for our app to run.

Dna-App.ipynb ☆
 File Edit View Insert Runtime Tools Help [All changes saved](#)

+ Code + Text
 Writing dna-app.py

```
!streamlit run dna-app.py & npx localtunnel --port 8501
```

Collecting usage statistics. To deactivate, set browser.gatherUsageStats to False.

You can now view your Streamlit app in your browser.

Network URL: <http://172.28.0.12:8501>
 External URL: <http://35.230.53.255:8501>

npx: installed 22 in 4.629s
 your url is: <https://cute-kings-do.loca.lt>

Main Link

cute-kings-do.loca.lt

YouTube Maps

Please proceed with caution.

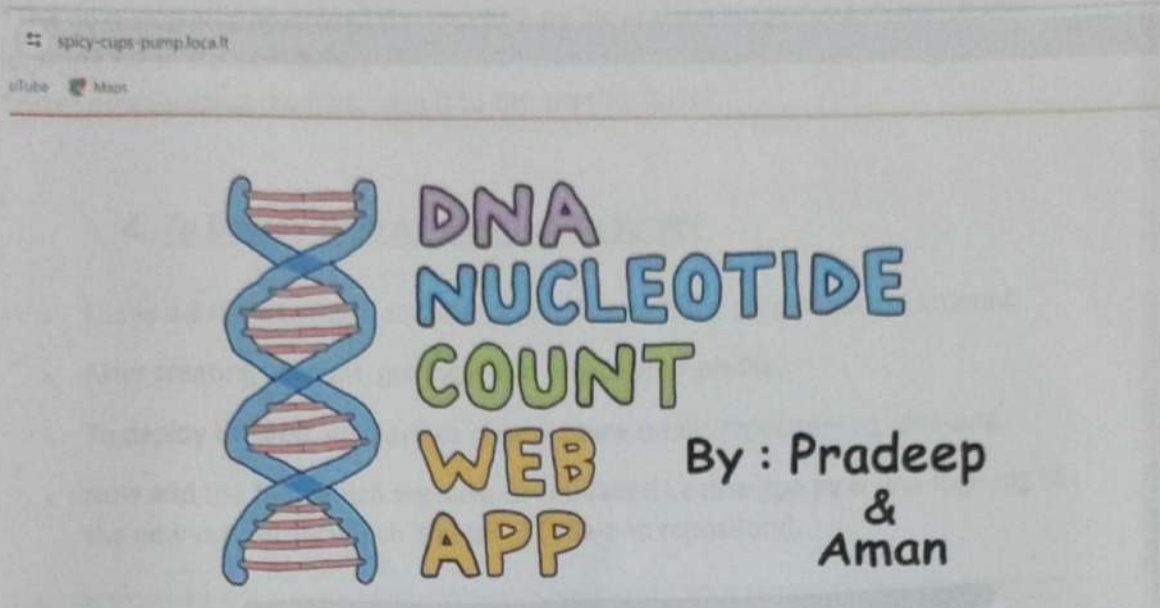
To access the website, please enter the tunnel password below.

If you don't know what it is, please ask whoever you got this link from.

Tunnel Password:

[Click to Submit](#)

*When we enter and submit the password, it is redirected to the app we created.

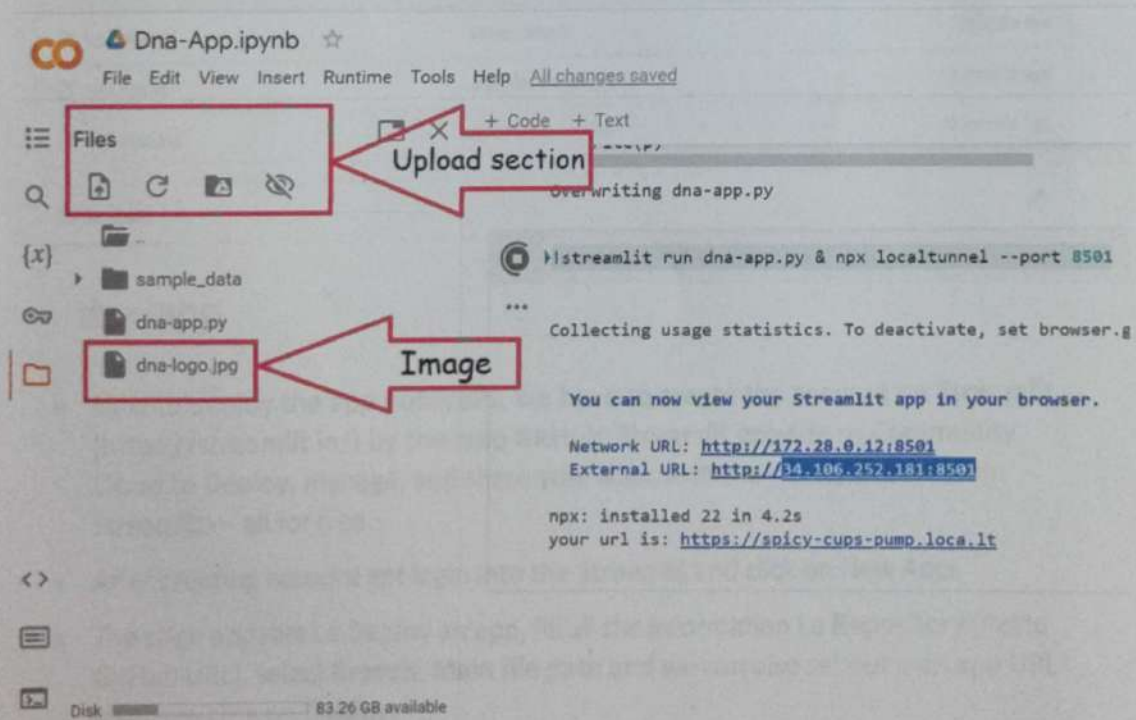


DNA Nucleotide Count Web App

This app counts the nucleotide composition of query DNA!

Thus our app is successfully worked.....

The above image is uploaded in the google colab through the upload section.



*The next step is to download the **dna-app.py** file and the supporting image i.e **dna-logo.jpg** file in the computer.

*After downloading the files, save it to the specific folder.

4. To Deploy Web Application on Server

- Firstly we have to open account on GitHub with the help of google account.
- After creating account, get login and create your profile.
- To deploy the app, we have to create a new public repository eg. dna-app.
- Now add the files which we have downloaded i.e dna-app.py & dna-logo.jpg to the new repository which is created (dna-app repository).

The screenshot shows the GitHub interface for a repository named 'dna-app' by user 'PradeepGanap'. The repository is public and has 1 branch and 0 tags. The commit history shows three commits, all made 2 months ago. The files listed are README.md, dna-app.py, and dna-logo.jpg. The README file is visible below the commit history.

File	Commit Message	Commit Date
README.md	Initial commit	2 months ago
dna-app.py	Add files via upload	2 months ago
dna-logo.jpg	Add files via upload	2 months ago

- Now to deploy the app publically, we have to create the account on **Streamlit** (<https://streamlit.io/>) by the help GitHub, Streamlit provide us Community Cloud to Deploy, manage, and share your apps with the world, directly from Streamlit — all for free.
- After creating account get login into the Streamlit and click on New Apps
- The page appears i.e Deploy an app, fill all the information i.e **Repository** (Paste GitHub URL), select **Branch**, **Main file path** and we can also set our own app URL and then click on **Deploy**.

share streamlit.io/Deploy

YouTube Maps

← Back

Deploy an app

Repository Paste GitHub URL
pradeepganap/dna-app

Branch
main

Main file path
dna-app.py

App URL (Optional)
dna-app-pradeepganap .streamlit.app

Domain is available

Advanced settings...

Deploy

*After the successful deployment of app on streamlit, the app is now publically available to use. (URL : <https://dna-app-pradeepganap.streamlit.app/>)

pradeepganap My apps Explore Create app

Your apps

- bioinfo · main · bioinfo.py
- dna-app · main · dna-app.py** ←

Thus our App is ready to use....

RESULTS

➤ DNA Nucleotide Count Web App Link :

<https://dna-app-pradeepqanap.streamlit.app/>

DNA NUCLEOTIDE COUNT WEB APP By : Pradeep & Aman

DNA Nucleotide Count Web App
This app counts the nucleotide composition of query DNA!

Manage app

DNA Nucleotide Count Web App

This app counts the nucleotide composition of query DNA!

Enter DNA sequence

Sequence input

DNA Query

```

GAGCAGGTGAGGCGAAGCGAAGTGTGAGGAGGATTTCTTCTTLAGGAGGAGGAGTCCCTGTCTTGGG
ATCTTLLAGAGCTTCCGATGATTTGCCCCCTGTGAGGTTAAGGAGACAGATGCTTTTGGAAATGC
GAAACCGATGATTTTATCATGACAGCAAGCTGCTTAAACCTATGCTTCCAGCAAGTGGCTGCT
  
```

INPUT (DNA Query)

Manage app

We can enter any DNA sequence in the **Sequence input** box and press (ctrl + enter) button to get the results.

1-

CONCLUSION

In Conclusion, we have build a DNA nucleotide Count Web Application in google colab and successfully deployed by using streamlit. These App provides A,T,G,C count (with OUTPUT in text, dataframe and in the form of chart) of any DNA sequence when entered into the INPUT (DNA Query).

REFERENCES

- Basic of python by geeksforgeeks ([geeksforgeeks.org/what-is-python/](https://www.geeksforgeeks.org/what-is-python/)) – what is python? , [what-is-the-difference-between-pythons-module-package-and-library?](https://www.geeksforgeeks.org/what-is-the-difference-between-pythons-module-package-and-library/) ([geeksforgeeks.org/what-is-the-difference-between-pythons-module-package-and-library/amp/](https://www.geeksforgeeks.org/what-is-the-difference-between-pythons-module-package-and-library/amp/)), and Python doc/essay/blurb by (python.org/doc/essays/blurb/)
- How to use google colab : (research.google.com/colaboratory/faq.html), google colab topics by scaler (scaler.com/topics/what-is-google-colab/), How to use google colab by techtarget (techtarget.com/searchenterpriseai/tutorial/Why-and-how-to-use-Google-Colab)
- GitHub Account tutorial (docs.github.com/en/get-started/onboarding/getting-started-with-your-github-account)
- Streamlit tutorial by medium blog (medium.com/data-and-beyond/streamlit-d357935b9c) and (docs.streamlit.io/get-started/installation#install-streamlit-on-windows)
- JCharis Jesse (github.com/Jcharis/DataScienceTools)
- How to launch streamlit app from google colab notebook (discuss.streamlit.io/t/how-to-launch-streamlit-app-from-

FUTURE PROSPECTS

- Generally these web application is used for counting the number of all four nucleotides and finding the percentage number of each of them which is to help students, professors and experts in the field of biological sciences to count nucleotide composition of DNA!
- In Future Research, we are developing the web application which is based on the principle of Central Dogma.
- These new version app will contain the various tools such as to count DNA nucleotide count (same as the perious app), The tool which covert DNA sequence into Replicate DNA sequence, It will also consist of the tool to covert DNA sequence into mRNA sequence (i.e Transcription), The will also have the features to covert the generated mRNA sequence into the Amino Acid sequence (i.e Translation mRNA is converted into Amino Acid sequence) and these all results will be in the form of print dictionary, print text, plot frequency, in table format, and in the form of bar chart.
- This program made are specifically for studies made with genetics and organisms in microbiology such as viruses or bacterias.