11/20/2020

PYTHON PROJECT

PROPERTY BUDGET ANALYSER

Group Members: -

Dearsh Oberoi - BT19CSE014

Harpreet Dogra - BT19CSE022

Shivashish Shukla - BT19CSE019

Vaibhav Chaurasia - BT19CSE010

Ajey Kumar Bind - BT19CSE015

**INDEX**

Under the Guidance of – Dr. Mayuri Digalwar

Group Members:

Dearsh Oberoi BT19CSE014

Harpreet Dogra BT19CSE22

Ajay Kumar Bind BT19CSE15

Vaibhav Chaurasia Bt19CSE010

Shivashish Shukla BT19CSE19

**INTRODUCTION**

Even though we all come from different places and backgrounds, the only thing that bring all the students together is IIITN and we are commonly called IIITians. Institutes like Indian Institute of Information Technology, Nagpur have students from all over India. We like together in hostel but the locations of our homes where we grew and studied till last year varies greatly. Some of us come from metropolitan cities while some from small towns.

When we were given this project of ITW1 we thought why not make a project based on real life example. As we know that there exists huge bungalow like houses in our towns and villages but it is not possible in big cities like Mumbai, Delhi, Bengaluru, Pune etc. In metropolitan cities there exits concept of flats in high rise buildings. For a middle-class person, it is not possible to purchase a bungalow in these cities. And searching a perfect house which suits your needs and your pocket is a difficult task. We have witnessed the city develop and the housing prices rise depends on the availability of amenities in a particular region. And this motivated us of collecting data related to this topic, grouping it together, cleaning it and at last analyze the data set. Our goal was to create an app for a person searching a flat in a city like Mumbai to get a brief idea of pricing of houses in different areas of that city with the amenities he required.



This project basically based on Python Programming Language.

**Python Programming Language**

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python supports modules and packages, which encourages program modularity and code reuse. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance.



As mentioned above this project is about analyzing the data collected and presenting it according to required specifications. We made this project as a general program which can be used with any data sets of any locations for finding the general cost of a flat at a location with required amenities and area. For this project we used Pandas for arranging, sorting, and segregating the data according to the needs and then Matplotlib for presenting that data in Graphical methods. A user needs to specify the money he can spend in range format and opt all the amenities. As per mentioned requirement the program will generally give a rough idea how much a flat will cost in particular location or show the trend of changing house area according to price and location. Many graphical representations such as pie chart, bar graph and dot plot has been used to display various data relations.

**CLEANING DATA**

Our team decided to work on a real – world dataset on property related data of flats on sale in Mumbai. While going through the huge CSV file, we noticed that the data had some inconsistencies, like some of the locations were misspelt, some cells had data missing and some data lacked uniformity (inconsistent capitalization) etc.

Then in the amenities section, we had 3 types of entries:

* **1:** If the amenity is provided.
* **0:** If that amenity is not provided.
* **9:** If no information is known.

We had to remove all the rows having the value 9.

An algorithm would differentiate between the correct entries and the wrong entries and give us wrong results. Thus, we had to remove all these anomalies to make our final result as accurate as possible.

First, we replaced all the cells having 9 with **np.nan** using the inbuilt **df.replace()** method.



Then, we got rid of rows with missing values using the **df.dropna()** inbuilt method in pandas.



This method accepts the following arguments:

* **axis:** Can be ‘0’ or ‘1’. ‘0’ means row is to be deleted and ‘1’ means column.
* **how:** Can be ‘any’ or ‘all’. ‘any’ deletes even if one entry is missing while ‘all’ deletes only if all entries are missing.
* **thresh:** Thresh is used to set the minimum number of entries for which the relevant subset of data will be deleted.
* **inplace:** If True, this makes the changes in the dataset itself.

Then we got rid of inconsistent capitalization and typos by selecting the relevant subset of data using **regular expressions** and the inbuilt **df.loc[]** method in pandas.



In the above example, we fixed inconsistencies with the Location named ‘Kharghar’. The df**.contains()** method accepts the following arguments:

* **regular expression**: This is the regular expression by which the data frame was refined by matching relevant entries from ‘Location’ column. **‘kh.\*ar’** matches all entries that begin with ‘kh’, are followed by any character any number of times and end with ‘ar’.
* **flags:** By setting flags to **re.I** we are making sure that Case is ignored while matching data entries.
* **regex:** This must be set to True to use regular expressions instead of normal string data.

In the end, we replaced all the relevant cells with a uniform data ‘Kharghar’.



**PANDAS LIBRARY**

Pandas is a software library used with Python programming language for data manipulation and analysis. It provides data structure and operations for manipulating numerical tables and time series.

There are many features of this library. Some of these are:

1. DataFrame object for data manipulation with integrated indexing.

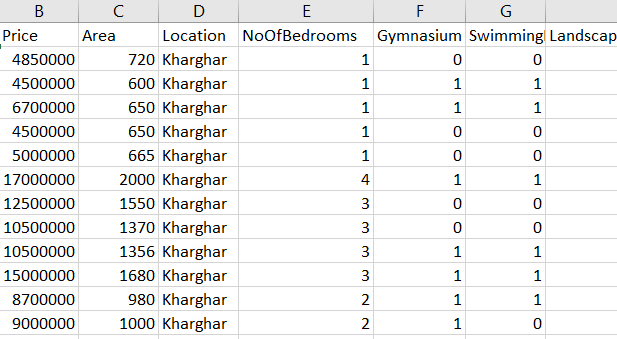
2. Tools for reading and writing data between in-memory data structures and different file formats.

3. Data alignment and integrated handling of missing data.

4. Reshaping and pivoting of data sets.

5. Label-based slicing, fancy indexing, and sub setting of large data sets.

6. Data structure column insertion and deletion.



This is a small part of the dataset just to give the reader an idea of what we are going to do.

We declared a Boolean variable for each of the amenities and set them all as False. Then we took input from user as to whether he wants a particular amenity or not. If a user wanted a particular amenity, we set the variable corresponding to it as True. Then we took input from user the budget range and the number of bedrooms (BHK) he wants.

We started to work with pandas by loading our CSV file into a data frame variable.

For each of the amenities, if the variable corresponding to it is True, we refined our data frame by selecting all those rows which had the entry corresponding to the column of that amenity as 1.

We did this for all the amenities.



We further refined our data frame by choosing rows whose entries in the ‘Price’ column are within the budget range.

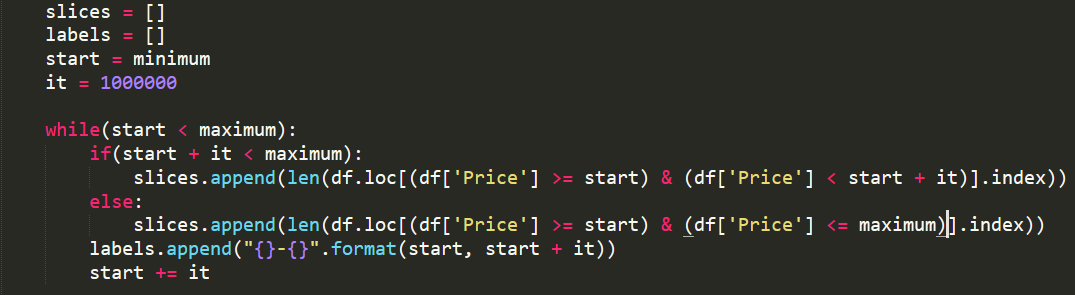


Finally, we further narrowed down our data frame by selecting all those rows which had the number of bedrooms as desired by the user.



Now, our data frame is ready to be processed into graphs.

Pandas was used once again for creating list to be passed in the function for creating pie graphs. Here, we grouped the number of flats appearing in a certain interval.



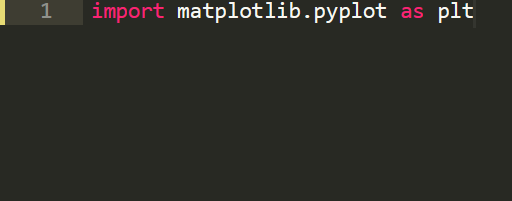
So, in the end we can say that PANDAS library was used in cleaning and refining our data frame and is extremely useful.

**MATPLOTLIB**



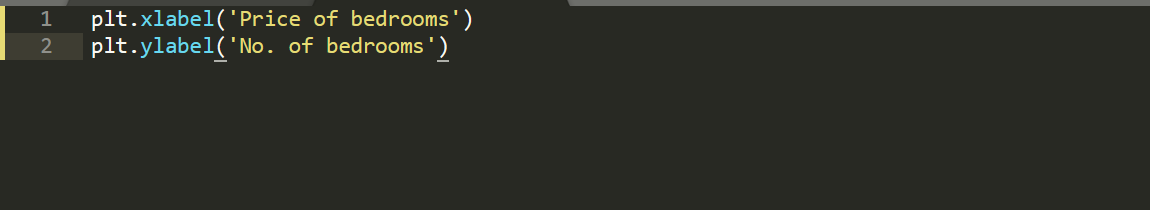
We can make graphs in Python using Matplotlib library. **Matplotlib** is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter.

So, to start using Matplotlib, we need to import Matplotlib and hence create the environment to perform visualizations. Go to the terminal window and type pip install matplotlib and press enter. It will download the required packages within seconds and then you are good to go. We can easily import library by writing the following line of code.

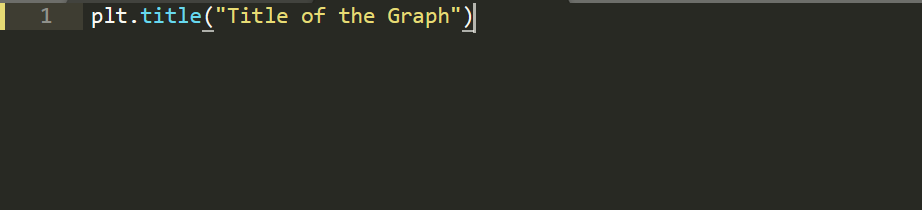


Now we can perform all operations using keyword plt. So, various operations we can use using plt are mentioned below

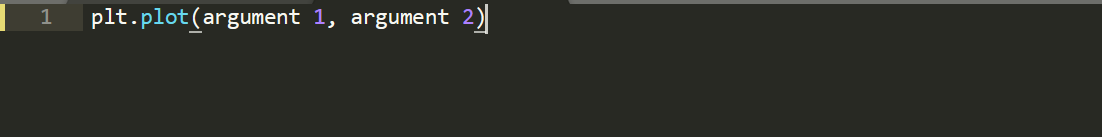
* We can add labels using function**plt.xlabel()** for x-axis and**plt.ylabel()**for y-axis.



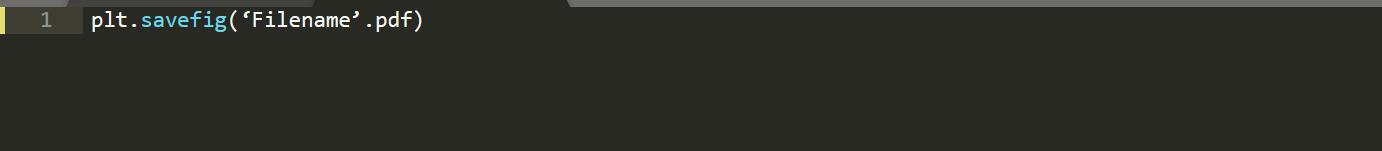
* We can add a title/heading to the plot using**plt.title()** function.



* **plt.plot()** method creates a graph between two variables entered as arguments.



* **plt.show()** is also a function used for plotting the graph and it does not take any arguments and it also does not give extra information about its current version and all.
* We can also save our graph with a very simple instruction plt**.savefig(‘Filename’.pdf) ,** it can be **png**, ***.*jpeg** and ***.*pdf**format.

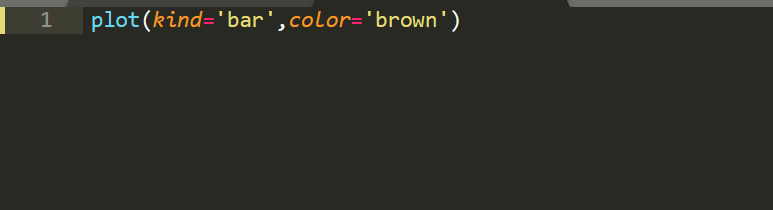


**Various Types of Graphs**

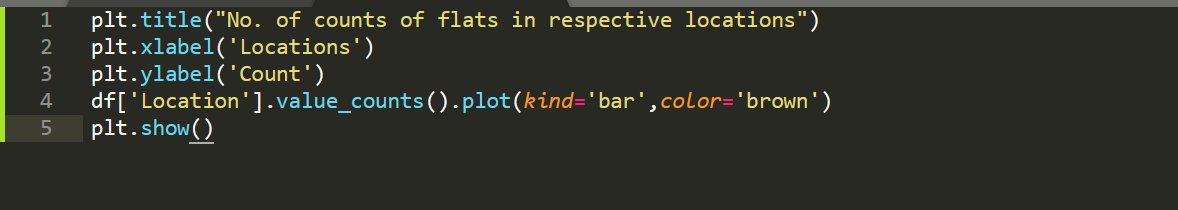
There are numerous types of plots available in Matplotlib, each has its own usage with certain specific data. Proper selection of plots is very essential and this needs to be understood before moving forward with the creation of plots. The most commonly used plots are:

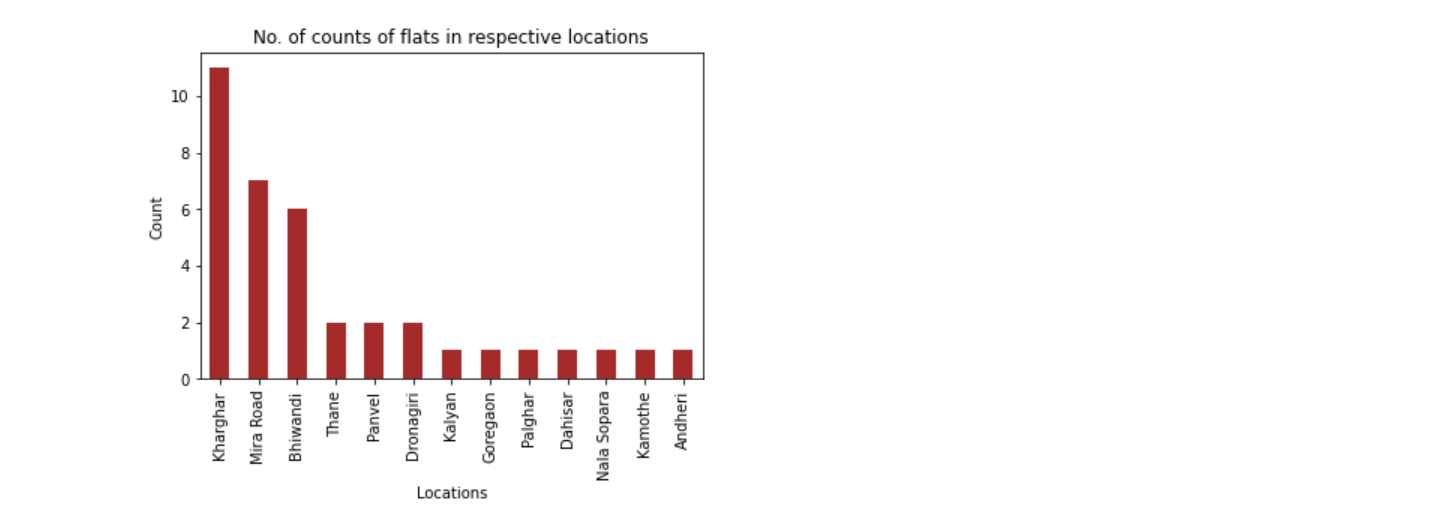
* Bar Plots.
* Pie Plots and Scatter Plots.

**BAR GRAPH**

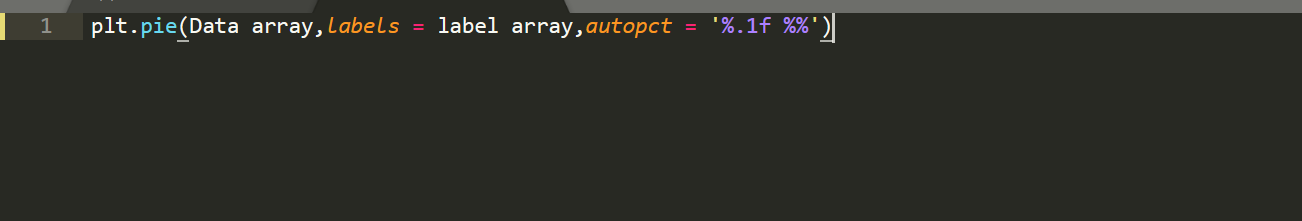


**OUR BAR GRAPH USED IN PROGRAM CODE**:

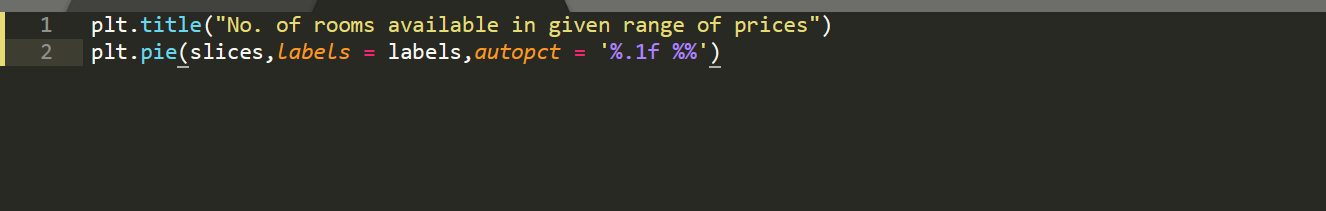




**PIE GRAPH**

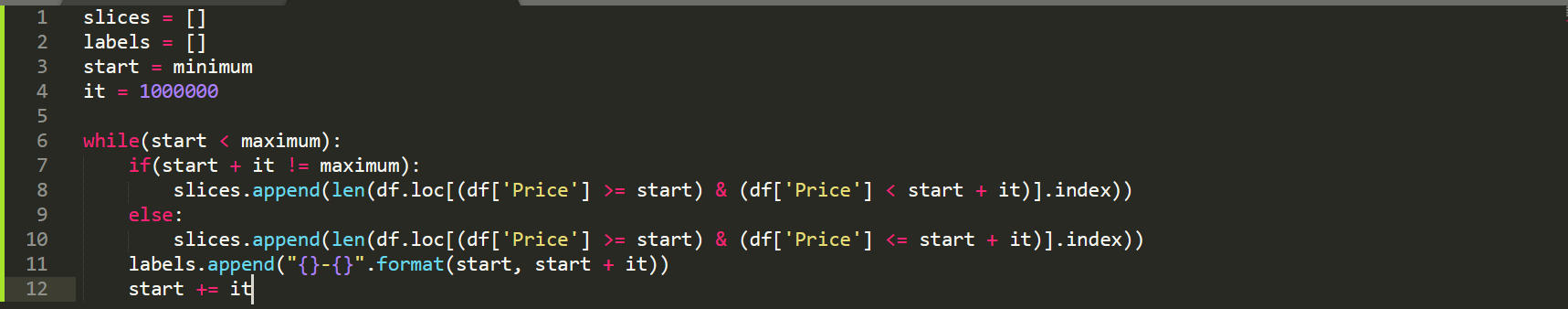


**OUR PIE GRAPH USED IN PROGRAM CODE**:

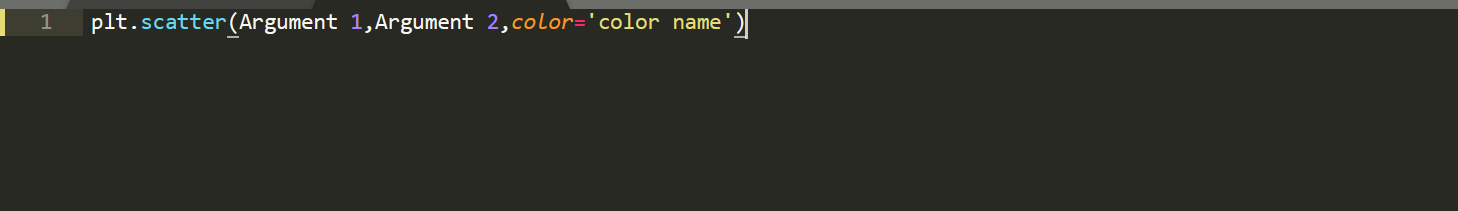




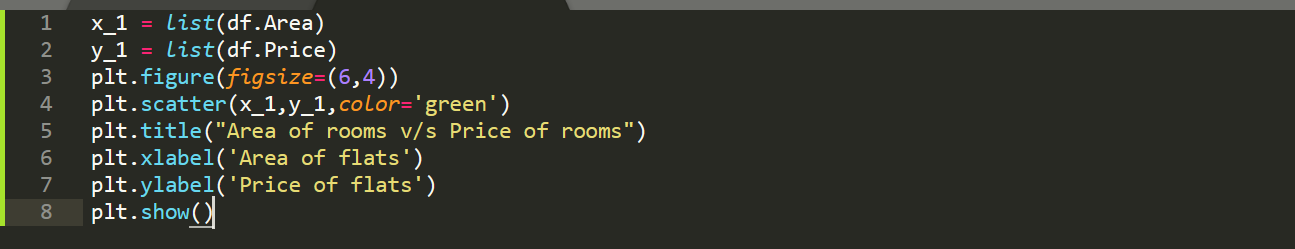
Slice array and label array are calculated by using following snippet of code

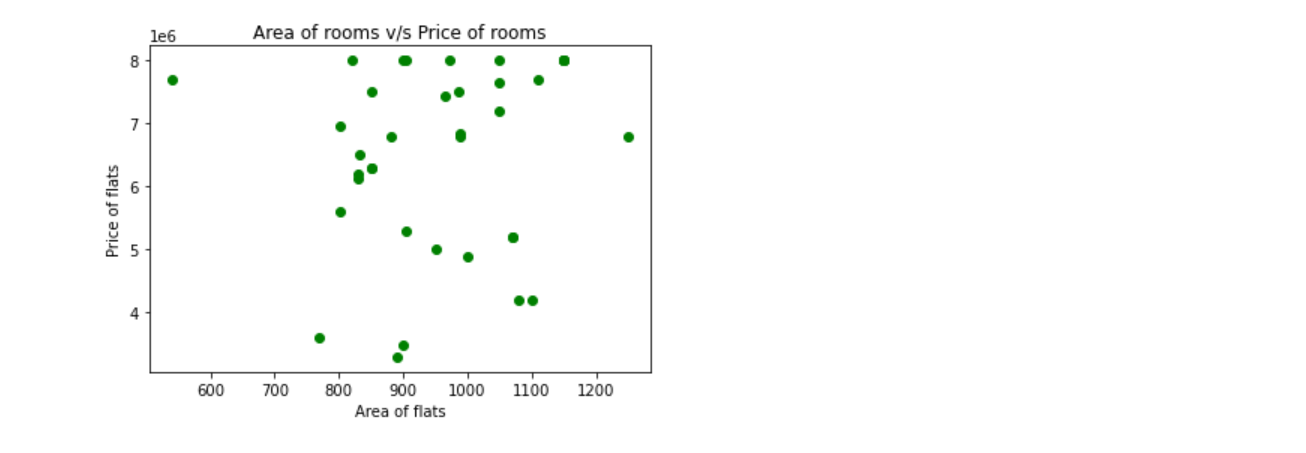


**SCATTER GRAPH**



**OUR SCATTER GRAPH USED IN PROGRAM CODE:**





**GUI**



GUI (graphical user interface) is a form of user interface that allows users to interact with electronic device through graphical icons and audio indicator such as primary notation, instead of text-based interface, typed command labels of text navigation.

Python offers multiple options for developing GUI. It is as standard python interface to the TK GUI toolkit shipped with python. Python with tkinter is the fastest and easiest way to create the GUI applications.



To create a GUI using tkinter in python, firstly we must install tkinter package in our system.

After installing tkinter we can easily import tkinter library by writing following lines of code.



In tkinter there are several options and widgets from which we can create our GUI in a systematic and creative manner.

Here is the list of widgets and options we have used to create our GUI: -

* **Canvas:** It is used to draw shapes, such as lines, ovals, polygons and rectangles, in your application.
* **Frame:** It is used as a container to organize other widgets in it.
* **Label:** It is used to provide a single line caption for other widgets and to give headings and provide text.
* **Checkbutton:** It is used to display multiple options for the user.
* **Entry:** It is used to take input from the user.
* **Button:** It is used to create a button on our GUI.
* **Place:** It is used to place the different widgets on the appropriate position on the app window and to fix geometry of the app window.

We have used different options in the widgets to style and execute our GUI. Here is the list: -

* **bg:** Used to give background colour to the different widgets.
* **text:** used to give text to provide information about the widgets
* **fg:** used to give colour to the text.
* **font:** used to define font type and size for our text.
* **command:** used to give task to perform when the widget is used.

Following options are used in place method to fix geometry of the widgets-:

* **relx:** to give the relative margin of the widget along x-axis of the parent window.
* **rely:** to give the relative margin of the widget along y-axis of the parent window.
* **relheight:** to give the relative height of the widget with respect to parent window.
* **relwidth:** to give the relative width of the widget with respect to parent window.

We have done the following procedure to make our GUI-:

* We have used canvas widget to initialize our app window and give it initial dimensions.



* We have made a frame and inside it creates a label to give our main heading.



* We have repeat above procedure to make another heading which tells users to mark their requirements.
* We have made a frame and inside it, we initialized eight different check buttons to provide multiple options to users. They can choose which amenity they want in their GUI.
* We have created another frame and inside it made three interfaces to take inputs from users using label and entry widgets.



* We have created another frame and create a final button which will run our main program and give the desired output.

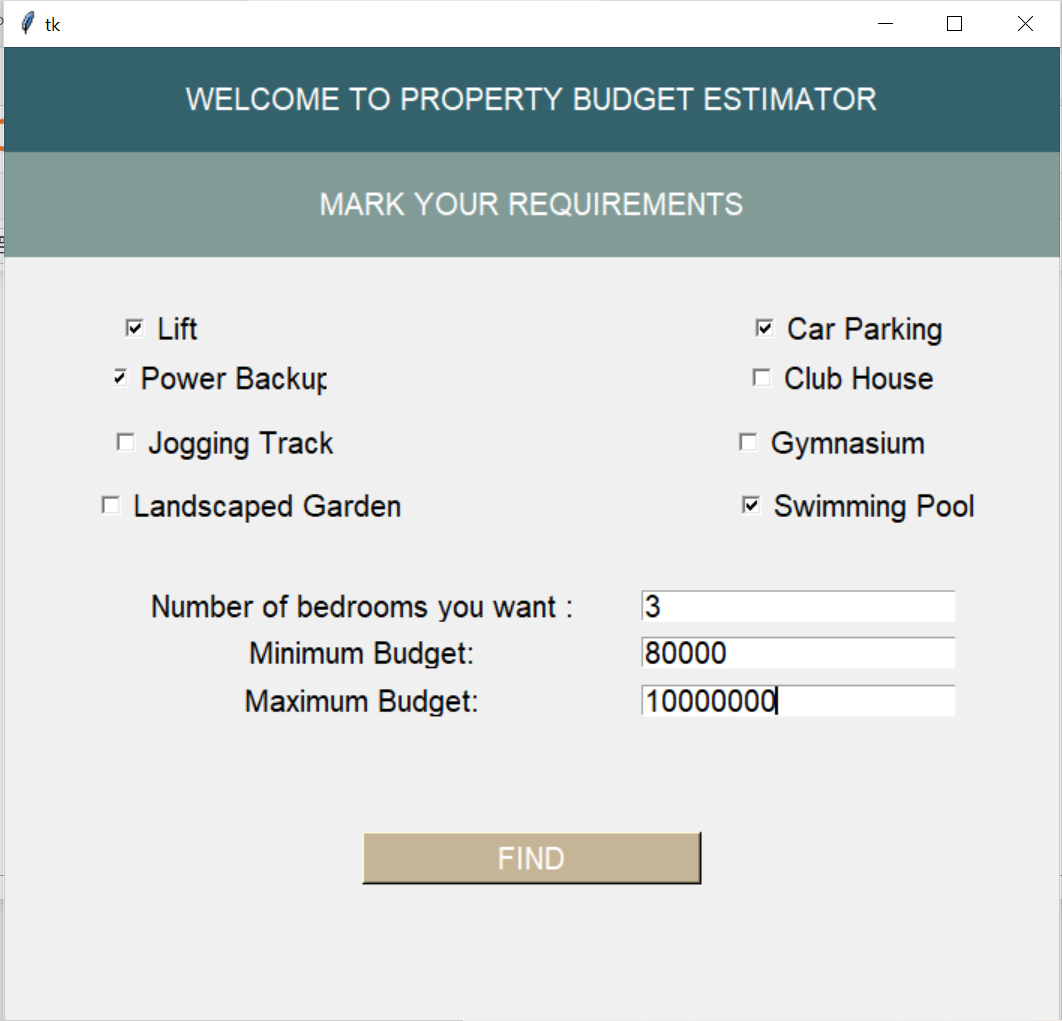


* We have used place method to fix our widgets in appropriate manner using following method-:



**Using above procedure, we have created the following GUI for our application**:

(INPUT IS ALSO SHOWN IN GUI)



(OUTPUT):

