Project Report on,

Sunville Properties App



Submitted by :

Shubh Joshi , Hemant Gosavi , Shubham Sangani

Under the Guidance of:

Prof. Junaid Khateeb (Director, Khateeb Institute of Technical Education)

Submitted to :

SUNVILLE PROPERTIES PVT LTD

Date of Submission: 10th July 2020.

Certificate Of Completion

This is to certify that , Mr \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ has successfully implemented an application \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

The Application has been accepted as a completed project as it meets all the requirements specified.

12TH July, 2020

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Khateeb Institute of Technical Education)

Acknowledgements

I would like to express my sincere gratitude to thank my professors and supervisors & for providing their proper and invaluable guidance, comments and suggestions throughout the course of the project. I would specially thank for constantly motivating me to work harder . Also I would like to thank for his assistance for the code & for his help during the preparation of the sample, for providing me an overview of the entire project.

Table of Contents

[1. Introduction 5](#_Toc387826896)

[2. Section 1 *(System Requirements)* 6](#_Toc387826896)

[3. Section 2 *(Technology Used)* 1](#_Toc387826897)

[*2.1* *Abbreviations and Acronyms (Heading 2)* 1](#_Toc387826898)

[*2.2* *Units* 2](#_Toc387826899)

[*2.3* *Equations* 2](#_Toc387826900)

[3. Section 3 3](#_Toc387826901)

[*3.1* *Contents* 3](#_Toc387826902)

[*3.2* *Figures and Table* 3](#_Toc387826903)

[4. Section 4 3](#_Toc387826901)

[5. Section 5 3](#_Toc387826901)

[6. Section 6 3](#_Toc387826901)

[7. Section 7 3](#_Toc387826901)

[8. Conclusion 4](#_Toc387826905)

[Major Contributions 5](#_Toc387826906)

[References 6](#_Toc387826907)

INTRODUCTION OF THE PROJECT

**1.1 OBJECTIVE**

Sunville Properties is a Colorado based property consultancy firm. They have appointed their agents across Major Cities around the world. They have sub- Coompanies which take care the buisness in different countries and are placed in the countries from where they operate from. The Company currently has been using multiple forms of data storage and want to streamline their working using an application, which can help them seamlessly navigate via different forms of storage. Also the company seeks some insights into the current data and also going further in future. So it has requested specific modules to be introduced in the system.

**1.2 SYSTEM REQUIREMENT SPECIFICATIONS**

The system is designed in the following manner

1) A Visual Interface to add the data in each of their tables. A login authentication for data modification

2) An order look up /the search feature based on the criteria: a) Order number b) Order Date c) Customer code which can be used either one or all of them together at a time.

3) Generation of a report that highlights the balance amounts for all orders in descending order. Mentions the name and code of the agent handling the order. The information is updated in the database .

4) Information of the country with maximum number of registered customer and the collective payment amount and outstanding amount for all these customers collectively.

The Requirements of the company:

1) On selection of the year, system should get the following

a) The total property area sold vs total property are leased in Sq-M only.

b) Of the years 2017,2018,2019-the year which got maximum leased area in CA and WS countries.

c) The Agent codes of all the agents who got deals in ‘OWNED’ categories across the years.

d) The agent who got the maximum deals in leased form, for the city Chilliwack

e) The best performer of the years 2017,2018,2019 based on the area leased and owned

f) The amount of property area sold for the month of july for all the years.

g) A time series analysis report of the orders received.

TECHNOLOGY USED

**2.1. Programming Language**

* Python (3.x)

**2.2. Libraries Used**

* tkinter
* pandas
* matplotlib
* pymysql
* re (Regular Expression)
* collections
* datetime

PYTHON

Python is a widely used general-purpose, high level programming language. It was mainly developed for emphasis on code readability. Its syntax allows programmers to express concepts in fewer lines of code.

Python is a programming language that lets you work you quickly an integrate systems more efficiently.

Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including procedural, object oriented and functional programming.

Tkinter

**tkinter** is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. It is used in this project in order to create the front-end ui.

Pandas

**pandas** is a Python package providing fast, flexible, and expressive data structures designed to make working with structured (tabular, multidimensional, potentially heterogeneous) and time series data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real world data analysis in Python. Here it is used to extract data from the excel file provided by the company and store it in dataframes accordingly.

Matplotlib

**Matplotlib** is one of the most popular Python packages used for data visualization. It is a cross-platform library for making 2D plots from data in arrays. It provides an object-oriented API that helps in embedding plots in applications using Python GUI toolkits like Tkinter, PyQt ,WxPython , etc. Here it is used to produce graphs of different types from the data extracted into dataframes by pandas, and then embed it in the Tkinter GUI.

Pymysql

**PyMySQL is** a pure-Python MySQL client library, based on PEP 249. Most public APIs are compatible with mysqlclient **PyMySQL** works with **MySQL** 5.5+ and MariaDB 5.5+. **MySQL** is a leading open source database management system. It is used to connect our code to the MySQL database provided by the company, so we can fetch, modify or analyze data.

Re

This module provides regular expression matching operations similar to those found in Perl.

Collections

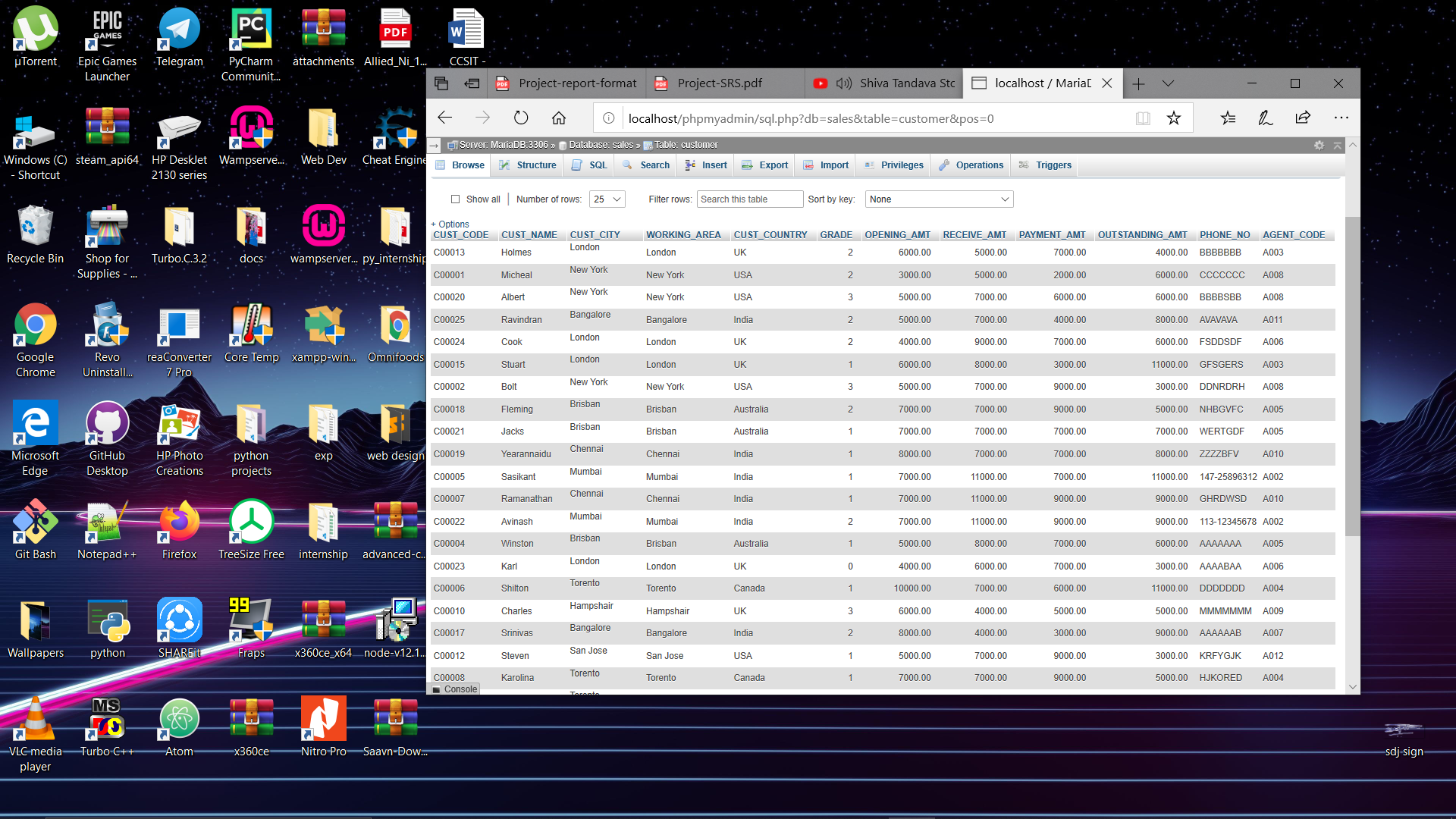
The [**collections**](https://pymotw.com/2/collections/index.html#module-collections) module includes container data types beyond the built-in types **list**, **dict**, and **tuple**. A **Counter** is a container that keeps track of how many times equivalent values are added. It can be used to implement the same algorithms for which bag or multiset data structures are commonly used in other languages. Counter is used here at various instances to count data i.e the no. of deals of an agent, no. of deals in a particular city/month/year etc.

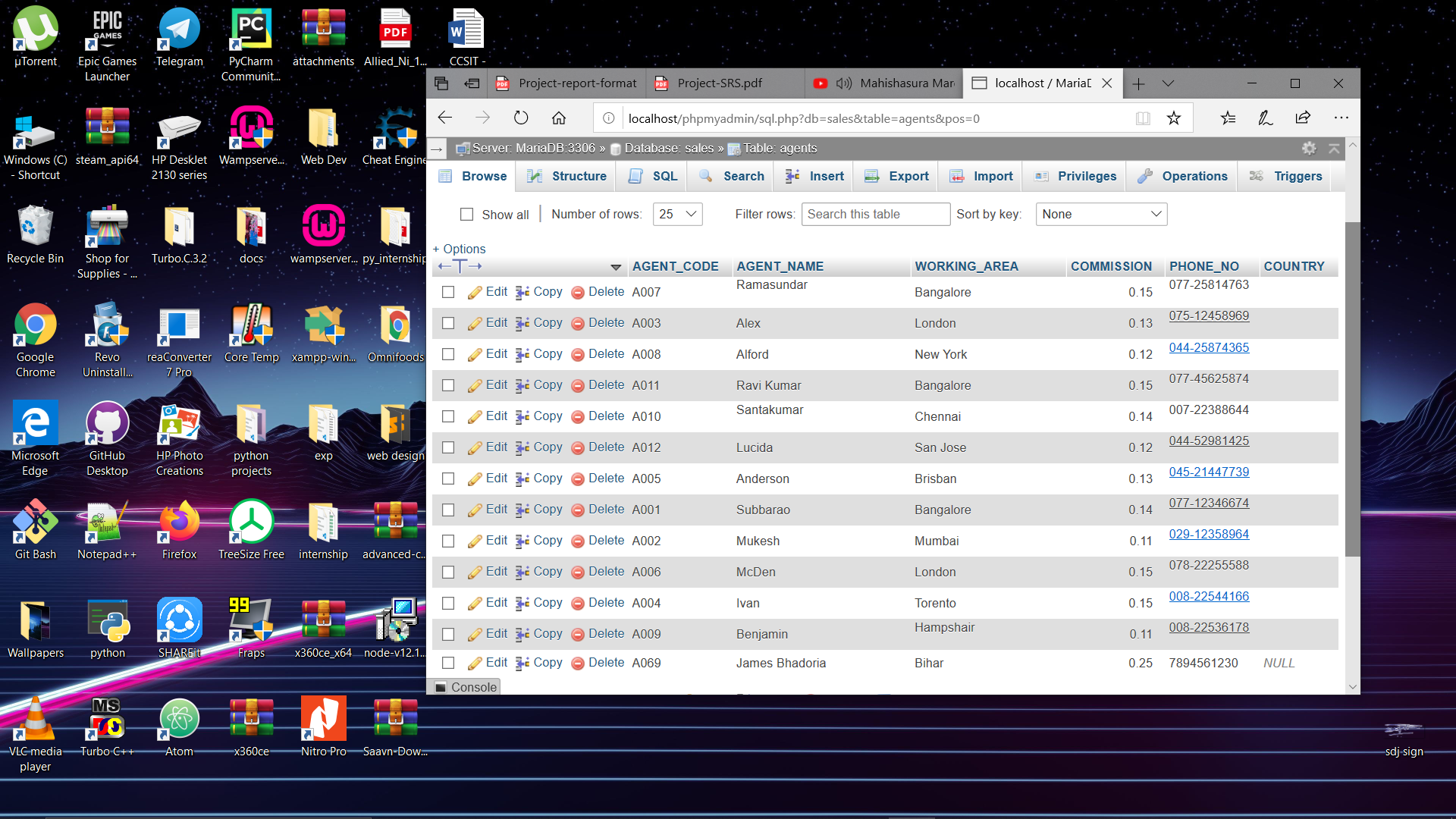
Datetime

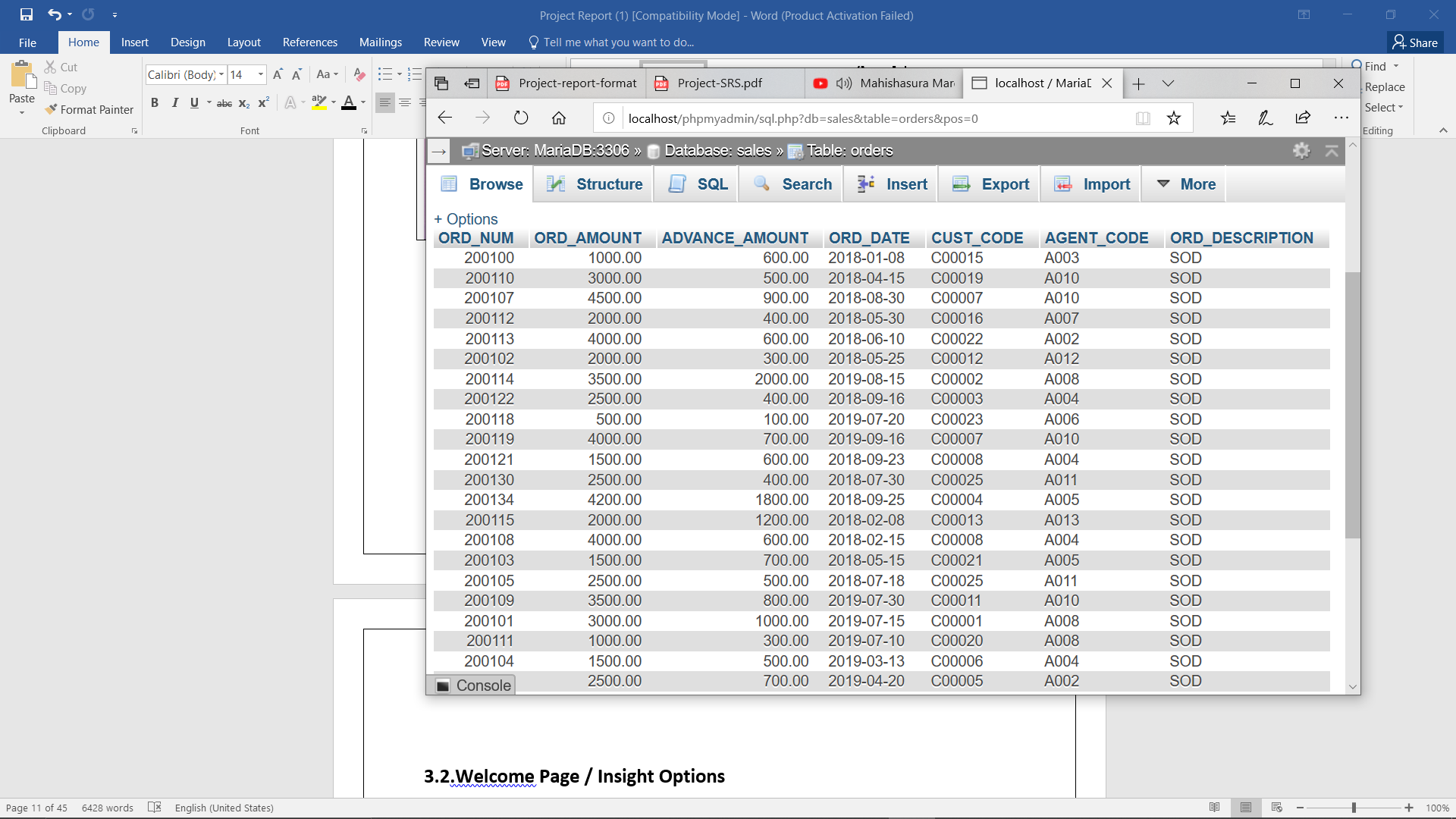
A date in Python is not a data type of its own, but we can import a module named **datetime** to work with dates as date objects. Here it is used to add order dates to the database .

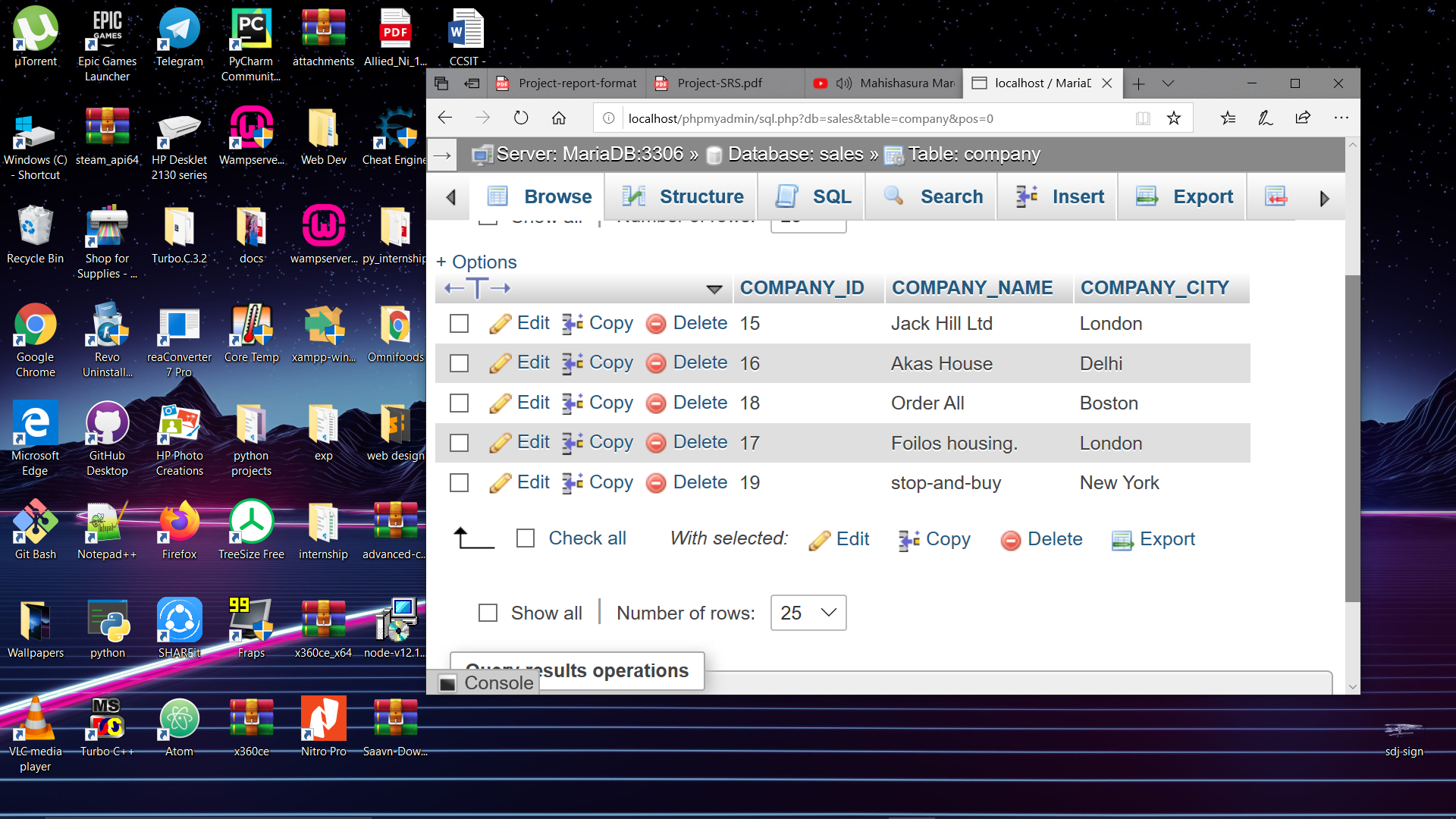
Data Provided by Company

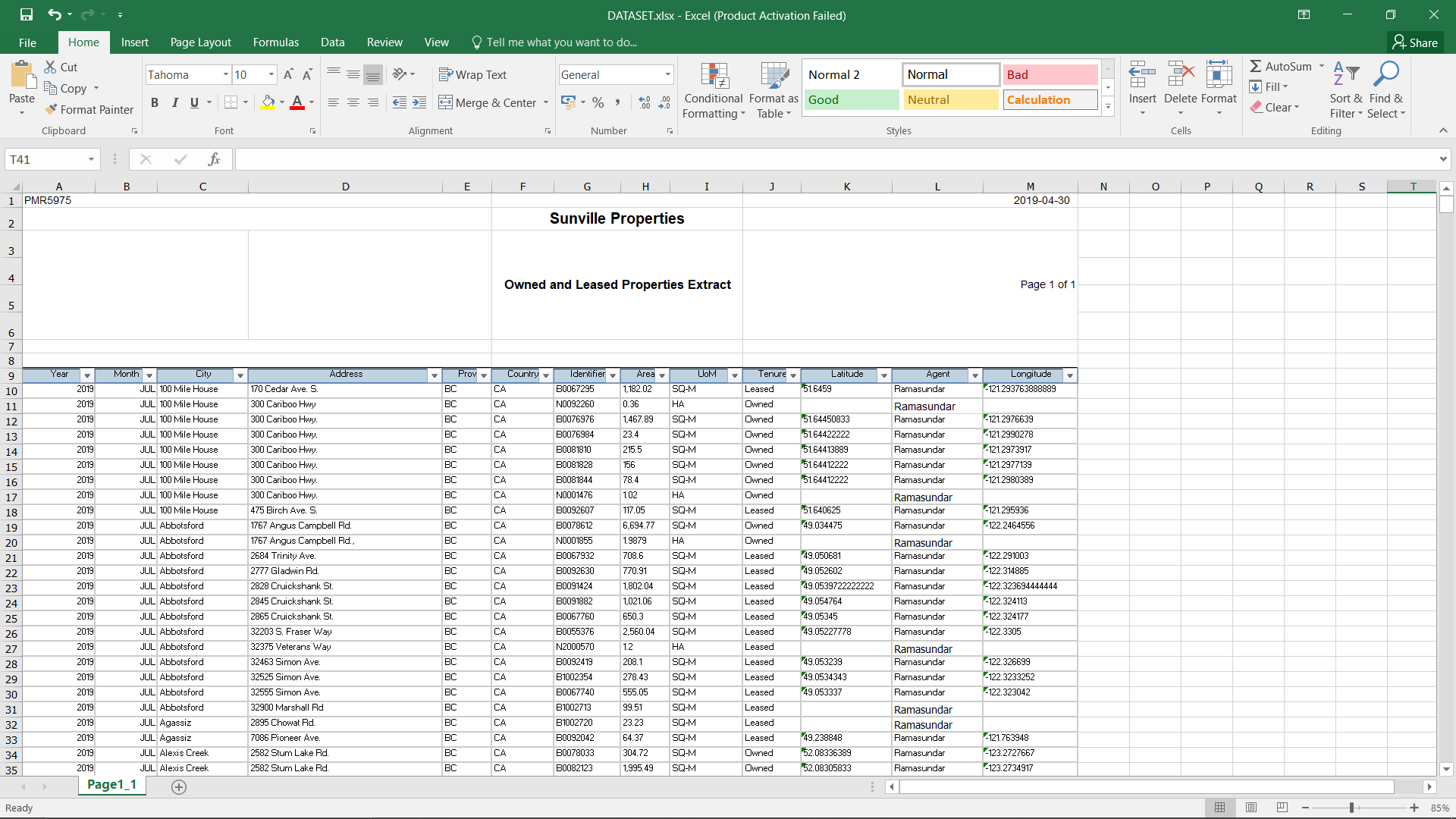
3.1. Snapshots of database/dataset











**3.2.DATBASE SOFTWARE USED – “MySQL” #we used MariaDB so change this**

**MySQL**  is an open-source  relational database management system (RDBMS).  "SQL", the abbreviation for Structured Query Language. A relational database organizes data into one or more data tables in which data types may be related to each other; these relations help structure the data. SQL is a language programmers use to create, modify and extract data from the relational database, as well as control user access to the database. In addition to relational databases and SQL, an RDBMS like MySQL works with an operating system to implement a relational database in a computer's storage system, manages users, allows for network access and facilitates testing database integrity and creation of backups.

MySQL has stand-alone clients that allow users to interact directly with a MySQL database using SQL, but more often MySQL is used with other programs to implement applications that need relational database capability. MySQL is a component of the LAMP web application softwar stack,. MySQL is used by many database-driven web applications, including Drupal, Joomla, phpBB, and WordPress. MySQL is also used by many popular websites, including Facebook,  Flickr, Media ,Wiki , Twitter and YouTube

SNAPSHOTS

**3.2.Login / Register**

|  |  |
| --- | --- |
| C:\Users\shubh\Desktop\New folder\login.png | C:\Users\shubh\Desktop\New folder\reg.png |

**Code for login:**

global username\_verify, password\_verify #username and password acquired from entry field

connection = pymysql.connect(host="localhost", user="root", passwd="",database="sales") # database connection

cursor = connection.cursor()

select\_query = "SELECT \* FROM company\_users where email = '" + username\_verify.get() + "' AND password = '" + password\_verify.get() + "';"

cursor.execute(select\_query) # executing the queries

user\_info = cursor.fetchall()

connection.commit().

connection.close()

**Code for registration:**

connection = pymysql.connect(host="localhost", user="root", passwd="", database="sales") # database connection

cursor = connection.cursor()

insert\_query = "INSERT INTO company\_users (fullname, email, password, company\_id, company\_name, company\_city) VALUES('"+ fullname.get() + "', '"+ email.get() + "', '"+ password.get() + "', '"+ company\_id.get() + "', '"+ company\_name.get() + "', '"+ company\_city.get() + "' );" # queries for inserting values

cursor.execute(insert\_query)

connection.commit()

connection.close()

**3.2.Welcome Page / Insight Options**

|  |  |
| --- | --- |
| C:\Users\shubh\Desktop\New folder\welcome.png | C:\Users\shubh\Desktop\New folder\insight menu.png |

**Code for Welcome Screen :**

# Just Buttons mapped to different functions

Button(screen2, image = lookup , border = 0, command = search\_order)

Button(screen2, image = new\_order , border = 0, command= update\_order)

Button(screen2, image = new\_agent , border = 0, command= update\_agent)

Button(screen2, image = new\_customer , border = 0, command= update\_customer)

Button(screen2, image = new\_company , border = 0, command= update\_company)

Button(screen2, image = balance\_button , border = 0, command = balance\_report)

Button(screen2, image = cumulative\_button , border = 0,command = cumulative\_data)

Button(screen2, image = insights\_button , border = 0,command=insight\_options)

**Code for Insight Options:**

# Just Buttons mapped to different functions

Button(insight\_menu, image = i1, border = 0 ,command=insight\_1)

Button(insight\_menu, image = i2, border = 0 ,command=insight\_2)

Button(insight\_menu, image = i3, border = 0 ,command=insight\_3)

Button(insight\_menu, image = i4, border = 0 ,command=insight\_4)

Button(insight\_menu, image = i5, border = 0 ,command=insight\_5)

Button(insight\_menu, image = i6, border = 0 ,command=insight\_6)

Button(insight\_menu, image = i7, border = 0 ,command=insight\_7)

**3.2.Search**

|  |  |
| --- | --- |
| C:\Users\shubh\Desktop\New folder\search.png |  |

**Code for search :**

global screen3 ,order\_year,order\_month,order\_day,order\_code,customer\_code

orderDateObj = datetime.strptime(order\_year.get() + "-" + order\_month.get() + "-" + order\_day.get(), '%Y-%m-%d')

connection = pymysql.connect(host="localhost", user="root", passwd="", database="sales") # database connection

cursor = connection.cursor()

select\_query = "SELECT \* FROM orders where ORD\_NUM = '" + order\_code.get() + "' OR ORD\_DATE = '" + orderDateObj.strftime("%Y-%m-%d") + "' OR CUST\_CODE = '" + customer\_code.get() + "';" # queries for retrieving values

cursor.execute(select\_query) # executing the queries

order\_info = cursor.fetchall()

connection.commit() # commiting the connection then closing it.

connection.close()

**3.2.Data Modification (Adding New Data to Database)**

|  |  |
| --- | --- |
| C:\Users\shubh\Desktop\New folder\new_ord.png | C:\Users\shubh\Desktop\New folder\new_cust.png |
| C:\Users\shubh\Desktop\New folder\new_agent.png | C:\Users\shubh\Desktop\New folder\new_comp.png |

**Code for :-**

**New Order:**

connection = pymysql.connect(host="localhost", user="root", passwd="", database="sales")

cursor = connection.cursor()

insert\_query = "INSERT INTO orders (ORD\_NUM, ORD\_AMOUNT, ADVANCE\_AMOUNT, ORD\_DATE, CUST\_CODE, AGENT\_CODE, ORD\_DESCRIPTION) VALUES('"+ ord\_no.get() + "', '"+ ord\_amt.get() + "', '"+ adv\_amt.get() + "', '" + orderDateObj\_1.strftime("%Y-%m-%d") + "', '"+ cust\_code.get() + "', '" + agt\_code.get() + "', '"+ ord\_des.get() +"' );" # fullname.get() + "', '"+ email.get() + "', '"+ password.get() + "', '"+ company\_id.get() + "', '"+ company\_name.get() + "', '"+ company\_city.get() + "' );"

cursor.execute(insert\_query)

connection.commit()

connection.close()

**New Agent:**

connection = pymysql.connect(host="localhost", user="root", passwd="", database="sales") # database connection

cursor = connection.cursor()

insert\_query = "INSERT INTO agents (AGENT\_CODE, AGENT\_NAME, WORKING\_AREA, COMMISSION, PHONE\_NO) VALUES('"+ agt\_code\_1.get() + "', '"+ agt\_name.get() + "', '"+ work\_area.get() + "', '" + comm + "', '" + agt\_phone.get() + "' );" # fullname.get() + "', '"+ email.get() + "', '"+ password.get() + "', '"+ company\_id.get() + "', '"+ company\_name.get() + "', '"+ company\_city.get() + "' );"

cursor.execute(insert\_query)

connection.commit()

connection.close()

**New Customer:**

connection = pymysql.connect(host="localhost", user="root", passwd="", database="sales")

cursor = connection.cursor()

insert\_query = "INSERT INTO customer (CUST\_CODE, CUST\_NAME, CUST\_CITY, WORKING\_AREA, CUST\_COUNTRY, GRADE, OPENING\_AMT, RECEIVE\_AMT, PAYMENT\_AMT, OUTSTANDING\_AMT, PHONE\_NO, AGENT\_CODE) VALUES('" + cust\_code\_2.get() + "', '"+ cust\_name.get() + "', '"+ cust\_city.get() + "', '"+ cust\_work\_area.get() + "', '"+ cust\_country.get() + "', '"+ grade.get() + "', '"+ opn\_amt.get() + "', '"+ rec\_amt.get() + "', '"+ pay\_amt.get() + "', '"+ out\_amt.get() + "', '"+ cust\_phone.get() + "', '"+ agt\_code\_2.get()+ "' );"

cursor.execute(insert\_query)

connection.commit()

connection.close()

**New Company:**

connection = pymysql.connect(host="localhost", user="root", passwd="", database="sales") # database connection

cursor = connection.cursor()

insert\_query = "INSERT INTO company (COMPANY\_ID, COMPANY\_NAME, COMPANY\_CITY) VALUES('" + company\_id\_1.get() + "', '"+ company\_name\_1.get() + "', '"+ company\_city\_1.get() + "' );"

cursor.execute(insert\_query) # executing the queries

connection.commit() # commiting the connection then closing it.

connection.close()

**3.2.Reports**

|  |  |
| --- | --- |
| **C:\Users\shubh\Desktop\New folder\BAR.png** | **C:\Users\shubh\Desktop\New folder\CR.png** |

**Code for Balance Amount Report :**

connection = pymysql.connect(host="localhost", user="root", passwd="", database="sales")

cursor = connection.cursor()

select\_query = "SELECT \* ,(ORD\_AMOUNT-ADVANCE\_AMOUNT) AS BALANCE\_AMT FROM orders LEFT JOIN agents ON orders.AGENT\_CODE=agents.AGENT\_CODE ORDER BY `BALANCE\_AMT` DESC ;"

cursor.execute(select\_query)

bal = cursor.fetchall()

alpha = [0,13,5,8] *#column numbers for order number, Balance amount, agent code ,agent name ,* *respectfully*

k = 0

new\_li = [[0 for x in range(4)] for y in range(len(bal))]

for i in range(len(bal)):

for j in alpha:

new\_li[i][k] = bal[i][j]

k+=1

if k == 4:

k=k-4

connection.commit()

connection.close()

for item in new\_li: *# tv is a treeview*

tv.insert('', 'end', values = item)

**Code for Cumulative report:**

connection = pymysql.connect(host="localhost", user="root", passwd="", database="sales")

cursor = connection.cursor()

select\_query = "SELECT SUM(`PAYMENT\_AMT`) FROM `customer`

cursor.execute(select\_query)

cum\_payment\_amt = cursor.fetchall()

select\_query = "SELECT SUM(`OUTSTANDING\_AMT`) FROM `customer`

cursor.execute(select\_query)

cum\_outstanding\_amt = cursor.fetchall()

select\_query = "SELECT `CUST\_COUNTRY` FROM `customer`"

cursor.execute(select\_query)

c\_name = cursor.fetchall()

connection.commit()

connection.close()

Label(screen5\_2, text= cum\_payment\_amt[0][0],width='20', font=("Roboto ", 13 , 'italic'),fg='black', bg='white',anchor=E).place(relx=.9, rely=.3,anchor='e')

Label(screen5\_2, text= cum\_outstanding\_amt[0][0] ,width='20', font=("Roboto ", 13 , 'italic'), fg='black', bg='white',anchor=E).place(relx=.9, rely=.4,anchor='e')

for i in range(len(c\_name)):

country\_names.append(c\_name[i][0])

c=Counter(country\_names).most\_common(1)

Label(screen5\_2, text= c[0][0],width='20', font=("Roboto ", 13 , 'italic'), fg='black', bg='white',anchor=E).place(relx=.9, rely=.6,anchor='e')

Label(screen5\_2, text= c[0][1] ,width='20', font=("Roboto ", 13 , 'italic'),fg='black', bg='white',anchor=E).place(relx=.9, rely=.7,anchor='e')

**3.2.Insights**

|  |  |
| --- | --- |
|  |  |
| C:\Users\shubh\AppData\Local\Microsoft\Windows\INetCache\Content.Word\i3.png | C:\Users\shubh\Desktop\New folder\i6.png |
|  | C:\Users\shubh\Desktop\New folder\i4.png |
| C:\Users\shubh\Desktop\New folder\i7.png |

**Code for** :

**Insight 1:**

owned= df[(df['Tenure']=='Owned') & (df['UoM']=='SQ-M')]

sumo=list(owned.groupby(['Year'])['Area'].sum())

leased= df[(df['Tenure']=='Leased') & (df['UoM']=='SQ-M')]

suml=list(leased.groupby(['Year'])['Area'].sum())

years=sorted(df['Year'].unique())

figure1 = plt.Figure(figsize=(7,5), dpi=85)

ax1 = figure1.add\_subplot(111)

bar1 = FigureCanvasTkAgg(figure1, insight\_1\_screen)

bar1.draw()

bar1.get\_tk\_widget().place(relx=.5,rely=.5,anchor='center')

df1 = pd.DataFrame({'Sold': sumo,'Leased': suml}, index=years)

df1.plot(kind='bar', ax=ax1,title='Total Property Area Sold vs Total Property Area Leased (in SQ-M only)')

**Insight 2:**

leased\_sqm = df[(df['Tenure']=='Leased') & (df['Country'] == countryselect.get())& (df['Year'] != 2020)& (df['UoM']=='SQ-M')]

leased\_ha = df[(df['Tenure']=='Leased') & (df['Country'] == countryselect.get())& (df['Year'] != 2020) & (df['UoM']=='HA')]

leased\_sqm\_sum=list(leased\_sqm.groupby(['Year'])['Area'].sum())

leased\_ha\_sum=list(leased\_ha.groupby(['Year'])['Area'].sum())

for i in range(len(leased\_ha\_sum)):

leased\_sqm\_sum[i] = leased\_sqm\_sum[i]+ (leased\_ha\_sum[i]\*10000)

years=sorted(leased\_sqm['Year'].unique())

figure1 = plt.Figure(figsize=(7,5), dpi=80)

ax1 = figure1.add\_subplot(111)

bar1 = FigureCanvasTkAgg(figure1, insight\_2\_screen)

bar1.draw()

bar1.get\_tk\_widget().place(relx=.5,rely=.5,anchor='center')

yearselect = 'CA'

df1 = pd.DataFrame({'Total Leased Area ': leased\_sqm\_sum}, index=years)

df1.plot(kind='pie', ax=ax1,autopct='%1.2f%%',startangle=0,subplots=True,legend=False,title='Total Leased Area in '+countryselect.get()+' (represented in SQ-M)')

**Insight 3:**

agent,count,code,temp=[],[],[],[]

df\_temp = df[(df['Tenure']=='Owned')]

a = Counter(df\_temp['Agent']).most\_common(50)

for i in a:

agent.append(i[0])

count.append(i[1])

connection = pymysql.connect(host="localhost", user="root", passwd="", database="sales") # database connection

cursor = connection.cursor()

for i in agent:

select\_query = "SELECT \* FROM agents where AGENT\_NAME = '" + i + "';" # queries for retrieving values

cursor.execute(select\_query) # executing the queries

info = cursor.fetchall()

if info:

temp.append(info[0][0])

i=i+' ( '+info[0][0]+')'

code.append(i)

while(len(count)!=len(code)):

count.pop()

connection.commit() # commiting the connection then closing it.

connection.close()

df1 = pd.DataFrame({'Agent Code': temp,'OWNED Deals': count})

figure1 = plt.Figure(figsize=(8,6.5), dpi=70)

ax1 = figure1.add\_subplot(111)

bar1 = FigureCanvasTkAgg(figure1,insight\_3\_screen)

bar1.draw()

bar1.get\_tk\_widget().place(relx=.5,rely=.475,anchor='center')

df1 = df1[['Agent Code','OWNED Deals']].groupby('Agent Code').sum()

df1.plot(title='Codes of Agents with OWNED Deals across the years 2017-2020',kind='bar', ax=ax1,subplots=True, legend = True)

**Insight 4:**

agents,count = [],[]

chill = df[(df['City']=='Chilliwack') & (df['Tenure']=='Leased')]

chill\_agents = Counter(chill['Agent']).most\_common(50)

for i in range(len(chill\_agents)):

agents.append(chill\_agents[i][0])

count.append(chill\_agents[i][1])

figure1 = plt.Figure(figsize=(7,5), dpi=85)

ax1 = figure1.add\_subplot(111)

bar1 = FigureCanvasTkAgg(figure1, insight\_4\_screen)

bar1.draw()

bar1.get\_tk\_widget().place(relx=.5,rely=.5,anchor='center')

df1 = pd.DataFrame({'No of Leased Deals in Chilliwack ': count}, index=agents)

df1.plot(kind='bar', ax=ax1,title='Agent with maximum leased deals in Chilliwack')

**Insight 5:**

agent1,area1,agent2,area2=[],[],[],[]

x1 = df[(df['Tenure']=='Owned') & (df['Year'] == int(yearselect.get())) ]

y1 = df[(df['Tenure']=='Leased') & (df['Year'] == int(yearselect.get())) ]

for a,row in x1.iterrows():

if row['UoM'] == 'HA':

row['Area'] = row['Area']\*10000

area1.append(row['Area'])

agent1.append(row['Agent'])

for a,row in y1.iterrows():

if row['UoM'] == 'HA':

row['Area'] = row['Area']\*10000

area2.append(row['Area'])

agent2.append(row['Agent'])

x2 = pd.DataFrame({'Agent': agent1,'Area': area1})

y2 = pd.DataFrame({'Agent': agent2,'Area': area2})

x3=list(x2.groupby(['Agent'])['Area'].sum())

y3=list(y2.groupby(['Agent'])['Area'].sum())

if yearselect.get()=='2019':

y3.insert(-3,0)

agent=sorted(x2['Agent'].unique())

figure1 = plt.Figure(figsize=(8,5), dpi=70)

ax1 = figure1.add\_subplot(111)

bar1 = FigureCanvasTkAgg(figure1, insight\_5\_screen)

bar1.draw()

bar1.get\_tk\_widget().place(relx=.5,rely=.5,anchor='center')

df1 = pd.DataFrame({'Owned Area': x3,'Leased Area':y3}, index=agent)

df1.plot(kind='bar',ax=ax1,title='Performance of Agents in Year '+yearselect.get())

**Insight 6:**

x = df[(df['Month']=='JUL') & (df['Tenure']=='Owned') & (df['UoM']=='SQ-M')]

y = df[(df['Month']=='JUL') & (df['Tenure']=='Owned') & (df['UoM']=='HA')]

sumsqm=list(x.groupby(['Year'])['Area'].sum())

sumha=list(y.groupby(['Year'])['Area'].sum())

years = sorted(x['Year'].unique())

for i in range(len(sumha)):

sumsqm[i] = sumsqm[i]+ (sumha[i]\*10000)

figure1 = plt.Figure(figsize=(7,5), dpi=85)

ax1 = figure1.add\_subplot(111)

bar1 = FigureCanvasTkAgg(figure1, insight\_6\_screen)

bar1.draw()

bar1.get\_tk\_widget().place(relx=.5,rely=.5,anchor='center')

df1 = pd.DataFrame({'Total Area (represented in SQ-M)': sumsqm}, index=years)

df1.plot(kind='bar', ax=ax1, title='Total Area Sold in the month of July across the Years (represented in SQ-M)')

**Insight 7:**

connection = pymysql.connect(host="localhost", user="root", passwd="", database="sales") # database connection

cursor = connection.cursor()

select\_query = "SELECT ORD\_AMOUNT,ORD\_DATE FROM orders ORDER BY orders.ORD\_DATE ASC ;" # queries for retrieving values

cursor.execute(select\_query) # executing the queries

info = cursor.fetchall()

amt,dt = [],[]

for i in info:

amt.append(float(i[0]))

dt.append(i[1])

connection.commit() # commiting the connection then closing it.

connection.close()

figure1 = plt.Figure(figsize=(12,5), dpi=60)

ax1 = figure1.add\_subplot(111)

bar1 = FigureCanvasTkAgg(figure1, insight\_7\_screen)

bar1.draw()

bar1.get\_tk\_widget().place(relx=.5,rely=.5,anchor='center')

df7 = pd.DataFrame({'Order Amount ': amt}, index=dt)

df7.plot(ax=ax1 , title='Time series analysis of all orders in year ' + str(info[0][1].year)+'-'+ str(info[-1][1].year))

**TESTING**

In this project, we have tried to make the code as fail-proof as possible , right from the start, i.e at Login Page , user-entered email and password are crosschecked in the database to see if an account with that credentials exists.

select\_query = "SELECT \* FROM company\_users where email = '" + username\_verify.get() + "' AND password = '" + password\_verify.get() + "';"

cursor.execute(select\_query) # executing the queries

user\_info = cursor.fetchall()

if user\_info is NULL , an error message is shown to the user to add correct data.

At the Registration Page, user enters their Name , Email-ID, Password ,Company Id/Name/City , the validations are set in such away that name should only contain alphabets (i.e no numbers/special characters), email- id should be valid, and the company info should match to a registered company in the database (i.e ID/Name/City) should correspond to each other And all Fields should be filled. Validations :-

if fullname.get() and email.get() and password.get() and repassword.get() and company\_id.get() and company\_name.get() and company\_city.get(): # checking for all empty values in entry field

if company\_id.get() == "--ID--" or company\_name.get() == "--Name--" or company\_city.get() == "--City--":

Label(screen1, text="⚠ Please Select Company ID / Name / City", fg="white", font=("Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

else:

if tnc.get(): # checking for acceptance of agreement

if re.match("^.+@(\[?)[a-zA-Z0-9-.]+.([a-zA-Z]{2,3}|[0-9]{1,3})(]?)$", email.get()): # validating the email

if password.get() == repassword.get(): # checking both password match or not

if all(x.isalpha() or x.isspace() for x in fullname.get()) and (len(fullname.get()) > 0):

# if u enter in this block everything is fine just enter the values in database

connection = pymysql.connect(host="localhost", user="root", passwd="", database="sales") # database connection

cursor = connection.cursor()

select\_query = "SELECT \* FROM company where COMPANY\_ID = '" + company\_id.get() + "' AND COMPANY\_NAME = '" + company\_name.get() + "' AND COMPANY\_CITY = '" + company\_city.get() + "';" # queries for retrieving values

cursor.execute(select\_query) # executing the queries

company\_info = cursor.fetchall()

connection.commit() # commiting the connection then closing it.

connection.close()

if company\_info:

# Only then, new user entry is added in the database . Otherwise the user is told to enter valid data .

At the search page, the user is expected to enter either Order Number or Order Date or Customer Code , in order to lookup an order . Code:-

orderDateObj = datetime.strptime(order\_year.get() + "-" + order\_month.get() + "-" + order\_day.get(), '%Y-%m-%d')

connection = pymysql.connect(host="localhost", user="root", passwd="", database="sales") # database connection

cursor = connection.cursor()

select\_query = "SELECT \* FROM orders where ORD\_NUM = '" + order\_code.get() + "' OR ORD\_DATE = '" + orderDateObj.strftime("%Y-%m-%d") + "' OR CUST\_CODE = '" + customer\_code.get() + "';" # queries for retrieving values

cursor.execute(select\_query) # executing the queries

order\_info = cursor.fetchall()

connection.commit() # commiting the connection then closing it.

connection.close() # closing the connection of the database

if order\_info:

# Only then the order data is shown . Otherwise the user is told to enter valid data .

When adding new order , the Order number should be 6 numbers only , all the amounts should be digits only, and Customer Code and Agent code should be valid (i.e It should be an existing Customer and Agent in the database). Validations:-

if ord\_no.get() and adv\_amt.get() and ord\_amt.get() and cust\_code.get() and agt\_code.get() and ord\_des.get() and order\_year\_1.get() and order\_month\_1.get() and order\_day\_1.get():

orderDateObj\_1 = datetime.strptime(order\_year\_1.get() + "-" + order\_month\_1.get() + "-" + order\_day\_1.get(), '%Y-%m-%d')

if len(ord\_no.get()) == 6 and ord\_no.get().isdigit():

pass

else:

Label(screen4\_1, text="⚠ Order number should have 6 digits ", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

if ord\_amt.get() and ord\_amt.get().isdigit():

if adv\_amt.get() and adv\_amt.get().isdigit():

if order\_year\_1.get() and order\_month\_1.get() and order\_day\_1.get():

if len(cust\_code.get()) == 6 and cust\_code.get().isalnum():

connection = pymysql.connect(host="localhost", user="root", passwd="", database="sales") # database connection

cursor = connection.cursor()

select\_query = "SELECT \* FROM customer where CUST\_CODE = '" + cust\_code.get() + "';" # queries for retrieving values

cursor.execute(select\_query) # executing the queries

cust\_code\_info = cursor.fetchall()

connection.commit() # commiting the connection then closing it.

connection.close()

if cust\_code\_info:

pass

else:

Label(screen4\_1, text="⚠ Customer Code not found in the Database, Enter Valid Customer Code", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

if len(agt\_code.get()) == 4 and agt\_code.get().isalnum():

connection = pymysql.connect(host="localhost", user="root", passwd="", database="sales") # database connection

cursor = connection.cursor()

select\_query = "SELECT \* FROM agents where AGENT\_CODE = '" + agt\_code.get() + "';" # queries for retrieving values

cursor.execute(select\_query) # executing the queries

agent\_code\_info = cursor.fetchall()

connection.commit() # commiting the connection then closing it.

connection.close()

if agent\_code\_info:

pass

else:

Label(screen4\_1, text="⚠ Agent Code not found in the Database, Enter Valid Agent Code", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

if all(x.isalpha() or x.isspace() for x in ord\_des.get()) and (len(ord\_des.get()) > 0):

# Only then the order data is added . Otherwise the user is told to enter valid data

When adding new agent , the Agent Code should be 4 characters (alphanumberic) , the phone no. should be 10 digits only, Name and working area should not have numbers or special characters,All fields should be filled. Validations:-

if agt\_code\_1.get() and agt\_name.get() and work\_area.get() and commission.get() and agt\_phone.get():

if len(agt\_code\_1.get()) == 4 and agt\_code\_1.get().isalnum():

if all(x.isalpha() or x.isspace() for x in agt\_name.get()) and (len(agt\_name.get()) > 0):

if all(x.isalpha() or x.isspace() for x in work\_area.get()) and (len(work\_area.get()) > 0):

if commission.get().isdigit():

if int(commission.get()) > 0 and int(commission.get()) < 99:

comm = '0.'+ commission.get()

pass

else:

Label(screen4\_2, text="⚠ Commission Percentage should be in the range of 0 - 99", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

if len(agt\_phone.get()) == 10 and agt\_phone.get().isdigit():

# Only then the agent data is added . Otherwise the user is told to enter valid data .

When adding new customer, all names and areas should not contain numbers or special characters , all amounts and phone no. should be numbers only , agent code should be valid.All fields should be filled. Validations:-

if cust\_code\_2 and cust\_name and cust\_city and cust\_work\_area and cust\_country and grade and opn\_amt and rec\_amt and pay\_amt and out\_amt and cust\_phone and agt\_code\_2:

if len(cust\_code\_2.get()) == 6 and cust\_code\_2.get().isalnum():

if all(x.isalpha() or x.isspace() for x in cust\_name.get()) and (len(cust\_name.get()) > 0):

if all(x.isalpha() or x.isspace() for x in cust\_city.get()) and (len(cust\_city.get()) > 0):

if all(x.isalpha() or x.isspace() for x in cust\_work\_area.get()) and (len(cust\_work\_area.get()) > 0):

if all(x.isalpha() or x.isspace() for x in cust\_country.get()) and (len(cust\_country.get()) > 0):

if grade.get().isdigit():

if int(grade.get()) > 0 and int(grade.get()) < 4:

pass

else:

Label(screen4\_3, text="⚠ Commission should be in the range of 1 - 3", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

if opn\_amt.get().isdigit():

if rec\_amt.get().isdigit():

if pay\_amt.get().isdigit():

if out\_amt.get().isdigit():

if len(cust\_phone.get()) == 10 and cust\_phone.get().isdigit():

if len(agt\_code\_2.get()) == 4 and agt\_code\_2.get().isalnum():

connection = pymysql.connect(host="localhost", user="root", passwd="", database="sales") # database connection

cursor = connection.cursor()

select\_query = "SELECT \* FROM agents where AGENT\_CODE = '" + agt\_code\_2.get() + "';" # queries for retrieving values

cursor.execute(select\_query) # executing the queries

agent\_code\_info = cursor.fetchall()

connection.commit() # commiting the connection then closing it.

connection.close()

if agent\_code\_info:

pass

else:

Label(screen4\_3, text="⚠ Agent Code not found in the Database, Enter Valid Agent Code", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

# Only then the customer data is added . Otherwise the user is told to enter valid data .

When adding new company, all names and areas should not contain numbers or special characters , ID should be no. in range 1-99 only. All fields should be filled. Validations:-

if company\_id\_1.get() and company\_name\_1.get() and company\_city\_1.get():

if company\_id\_1.get().isdigit() and int(company\_id\_1.get())>0 and int(company\_id\_1.get())<99:

if all(x.isalpha() or x.isspace() for x in company\_name\_1.get()) and (len(company\_name\_1.get()) > 0):

if all(x.isalpha() or x.isspace() for x in company\_city\_1.get()) and (len(company\_city\_1.get()) > 0):

# Only then the company data is added . Otherwise the user is told to enter valid data .

SOURCE CODE

from tkinter import \*

from tkinter import messagebox

from ttkthemes import themed\_tk as tk

from tkinter import ttk

import re, pymysql

from datetime import datetime

import pandas as pd

from collections import Counter

import matplotlib.pyplot as plt

from matplotlib.backends.backend\_tkagg import (FigureCanvasTkAgg, NavigationToolbar2Tk)

from matplotlib.figure import Figure

df1=pd.read\_excel('DATASET.xlsx',header=None).iloc[8:-1,:]

x=list(df1.iloc[0])

df1=df1.iloc[1:]

df1.to\_excel('NEW\_DATASET.xlsx',index=False,header=x)

df=pd.read\_excel('NEW\_DATASET.xlsx')

def adjustWindow(window):

w = 600 # width for the window size

h = 600 # height for the window size

ws = screen.winfo\_screenwidth() # width of the screen

hs = screen.winfo\_screenheight() # height of the screen

x = (ws/2) - (w/2) # calculate x and y coordinates for the Tk window

y = (hs/2) - (h/2)

window.geometry('%dx%d+%d+%d' % (w, h, x, y)) # set the dimensions of the screen and where it is placed

window.resizable(False, False) # disabling the resize option for the window

def register():

global screen1, fullname, email, password, repassword, gender, tnc, company\_id, company\_name, company\_city # making all entry field variable global

fullname, email, password, repassword, tnc, company\_id, company\_name, company\_city = StringVar(),StringVar(),StringVar(),StringVar(),IntVar(),StringVar(),StringVar(),StringVar()

screen1 = Toplevel(screen)

screen1.title("Registeration")

adjustWindow(screen1) # configuring the window

image2,submit\_button,proceed\_button= PhotoImage(file="register\_final\_1.png"),PhotoImage(file="submit\_button\_3.png"),PhotoImage(file="proceed\_to\_login\_3.png")

label\_for\_image= Label(screen1, image=image2)

label\_for\_image.pack()#lace(x=0, y=0, relwidth=1, relheight=1)

label\_text,labelx,labely = ["Full Name ","E-mail ID ","Password ","Re-Password ","Company Info : "],[.343,.342,.3428,.360,.3],[.305,.385,.465,.545,.645]

for t,x,y in zip(label\_text,labelx,labely):

Label(screen1, text= t,font=("Roboto lt", 9, 'italic'), bg='black', fg='white').place(relx=x, rely=y, anchor="center")

ttk.Entry(screen1, textvar=fullname , width='39').place(relx=.5, rely=.34, anchor="center")

ttk.Entry(screen1, textvar=email , width='39').place(relx=.5, rely=.42, anchor="center")

ttk.Entry(screen1, textvar=password, show="\*" , width='39').place(relx=.5, rely=.50, anchor="center")

ttk.Entry(screen1, textvar=repassword, show="\*" , width='39').place(relx=.5, rely=.58, anchor="center")

id\_list,name\_list,city\_list = ['ID'], ['Name'],['City']

connection = pymysql.connect(host="localhost", user="root", passwd="", database="sales") # database connection

cursor = connection.cursor()

select\_query = "SELECT \* FROM company;"

cursor.execute(select\_query) # executing the queries

company\_info = cursor.fetchall()

for i in range(len(company\_info)):

id\_list.append(company\_info[i][0])

name\_list.append(company\_info[i][1])

city\_list.append(company\_info[i][2])

connection.commit() # commiting the connection then closing it.

connection.close()

droplist = ttk.OptionMenu(screen1, company\_id, \*id\_list)

droplist.config(width=2)

company\_id.set('ID')

droplist.place(relx=.465, rely=.65, anchor="center")

droplist = ttk.OptionMenu(screen1, company\_name, \*name\_list)

droplist.config(width=6)

company\_name.set('Name')

droplist.place(relx=.6, rely=.65, anchor="center")

droplist = ttk.OptionMenu(screen1, company\_city, \*city\_list)

droplist.config(width=5)

company\_city.set('City')

droplist.place(relx=.75, rely=.65, anchor="center")

ttk.Checkbutton(screen1, text="I accept all terms and conditions ", variable=tnc).place(relx=.5, rely=.735, anchor="center")

Button(screen1, image = submit\_button , border = 0,command=register\_user).place(relx=.5, rely=.85, anchor="center")

Button(screen1, image = proceed\_button , border = 0, command=screen1.destroy).place(relx=.5, rely=.90, anchor="center")

screen1.mainloop()

def register\_user():

if fullname.get() and email.get() and password.get() and repassword.get() and company\_id.get() and company\_name.get() and company\_city.get(): # checking for all empty values in entry field

if company\_id.get() == "--ID--" or company\_name.get() == "--Name--" or company\_city.get() == "--City--":

Label(screen1, text="⚠ Please Select Company ID / Name / City", fg="white", font=("Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

else:

if tnc.get(): # checking for acceptance of agreement

if re.match("^.+@(\[?)[a-zA-Z0-9-.]+.([a-zA-Z]{2,3}|[0-9]{1,3})(]?)$", email.get()): # validating the email

if password.get() == repassword.get(): # checking both password match or not

if all(x.isalpha() or x.isspace() for x in fullname.get()) and (len(fullname.get()) > 0):

# if u enter in this block everything is fine just enter the values in database

connection = pymysql.connect(host="localhost", user="root", passwd="", database="sales") # database connection

cursor = connection.cursor()

select\_query = "SELECT \* FROM company where COMPANY\_ID = '" + company\_id.get() + "' AND COMPANY\_NAME = '" + company\_name.get() + "' AND COMPANY\_CITY = '" + company\_city.get() + "';" # queries for retrieving values

cursor.execute(select\_query) # executing the queries

company\_info = cursor.fetchall()

connection.commit() # commiting the connection then closing it.

connection.close()

if company\_info:

print(company\_info)

connection = pymysql.connect(host="localhost", user="root", passwd="", database="sales") # database connection

cursor = connection.cursor()

insert\_query = "INSERT INTO company\_users (fullname, email, password, company\_id, company\_name, company\_city) VALUES('"+ fullname.get() + "', '"+ email.get() + "', '"+ password.get() + "', '"+ company\_id.get() + "', '"+ company\_name.get() + "', '"+ company\_city.get() + "' );" # queries for inserting values

cursor.execute(insert\_query) # executing the queries

connection.commit() # commiting the connection then closing it.

connection.close() # closing the connection of the database

Label(screen1, text="Registration Success ! You May Proceed To Login ", bg="white", font=("Roboto", 11), width='3000', anchor=W, fg='green').place(x=0, y=570) # printing successful registration message

else:

Label(screen1, text="⚠ Please Select Valid Company ID / Name / City", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

else:

Label(screen1, text="⚠ Name cannot contain Digits or Special Characters . Spaces are allowed." , fg="red", font=("Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

else:

Label(screen1, text="⚠ Password does not match", fg="red", font=(

"Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

else:

Label(screen1, text="⚠ Please enter valid email id", fg="red", font=(

"Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

else:

Label(screen1, text="⚠ Please accept the agreement", fg="red",

font=("Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

else:

Label(screen1, text="⚠ Please fill all the details ", fg="red",font=("Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

def login\_verify():

global userID

connection = pymysql.connect(host="localhost", user="root", passwd="", database="sales") # database connection

cursor = connection.cursor()

select\_query = "SELECT \* FROM company\_users where email = '" + username\_verify.get() + "' AND password = '" + password\_verify.get() + "';" # queries for retrieving values

cursor.execute(select\_query) # executing the queries

user\_info = cursor.fetchall()

connection.commit() # commiting the connection then closing it.

connection.close() # closing the connection of the database

if user\_info:

messagebox.showinfo("Congratulation", "Login Succesfull") # displaying message for successful login

userID = user\_info[0][0]

welcome\_page(user\_info) # opening welcome window

else:

messagebox.showerror("Error", "Invalid Username or Password")

def welcome\_page(user\_info):

global screen2

screen2 = Toplevel(screen)

screen2.title("Welcome")

adjustWindow(screen2) # configuring the window

image1, lookup,new\_order,new\_agent,new\_customer,new\_company,balance\_button,cumulative\_button,back,insights\_button, = PhotoImage(file="bg\_1.png"), PhotoImage(file="lookup.png"), PhotoImage(file="new\_order.png"),PhotoImage(file="new\_agent.png"),PhotoImage(file="new\_customer.png"),PhotoImage(file="new\_company.png"),PhotoImage(file="balance.png"),PhotoImage(file="cumulative\_report.png"),PhotoImage(file="back.png"),PhotoImage(file="insights.png")

label\_for\_image= Label(screen2, image=image1)

label\_for\_image.place(x=0, y=0, relwidth=1, relheight=1)

Label(screen2, text="Welcome " + user\_info[0][1], width='500', height="1", font=("Ariel", 20,), fg='white', bg='#43315E', anchor =W).pack()

Label(screen2, text="Search ",width='500', font=("Roboto lt", 10 , 'italic'), bg='black', fg='white',anchor=W).place(relx=0, rely=.085)

Button(screen2, image = lookup , border = 0, command = search\_order).place(relx=.5, rely=.16, anchor="center")

Label(screen2, text="Add New Data To Database ",width='500', font=("Roboto lt", 10 , 'italic'), bg='black', fg='white',anchor=W).place(relx=0, rely=.2)

Button(screen2, image = new\_order , border = 0, command= update\_order).place(relx=.5, rely=.275, anchor="center")

Button(screen2, image = new\_agent , border = 0, command= update\_agent).place(relx=.5, rely=.320, anchor="center")

Button(screen2, image = new\_customer , border = 0, command= update\_customer).place(relx=.5, rely=.365, anchor="center")

Button(screen2, image = new\_company , border = 0, command= update\_company).place(relx=.5, rely=.410, anchor="center")

Label(screen2, text="Report ",width='500', font=("Roboto lt", 10 , 'italic'), bg='black', fg='white',anchor=W).place(relx=0, rely=.455)

Button(screen2, image = balance\_button , border = 0, command = balance\_report).place(relx=.5, rely=.530, anchor="center")

Button(screen2, image = cumulative\_button , border = 0,command = cumulative\_data).place(relx=.5, rely=.575, anchor="center")

Label(screen2, text="Insights ",width='500', font=("Roboto lt", 10 , 'italic'), bg='black', fg='white',anchor=W).place(relx=0, rely=.615)

Button(screen2, image = insights\_button , border = 0,command=insight\_options).place(relx=.5, rely=.685, anchor="center")

Button(screen2, image = back, border = 0, command=screen2.destroy).place(relx=.5, rely=.95, anchor="center")

screen2.mainloop()

def search\_order():

global screen3 ,order\_year,order\_month,order\_day,order\_code,customer\_code

order\_code = StringVar()

customer\_code = StringVar()

screen3 = Toplevel(screen)

screen3.title("Search Order")

adjustWindow(screen3) # configuring the window

back,search\_button,image1 = PhotoImage(file="back.png"),PhotoImage(file="search.png"),PhotoImage(file="bg\_1.png")

label\_for\_image= Label(screen3, image=image1)

label\_for\_image.place(x=0, y=0, relwidth=1, relheight=1)

Label(screen3, text="Search Order ", width='500', height="1", font=("Ariel", 20,), fg='white', bg='#43315E', anchor =W).pack()

order\_year,order\_month,order\_day = StringVar(),StringVar(),StringVar()

rx,ry,t,v = [1950,1,1],[2021,13,32],[.4,.5,.6],[order\_year,order\_month,order\_day]

for (i,j,k,l) in zip(rx,ry,t,v):

choices = list(range(i,j))

ttk.Combobox(screen3 , width=5, values = choices ,textvariable = l ).place(relx=k, rely=.135, anchor="center")

order\_year.set("2000")

order\_month.set("01")

order\_day.set("01")

label\_text,rx1,ry1=[],[],[]

Label(screen3, text="Order Number ", font=("Roboto lt", 10 , 'italic'), bg='black', fg='white').place(relx=.1, rely=.09, anchor="center")

ttk.Entry(screen3, textvar=order\_code , width='15').place(relx=.115, rely=.130, anchor="center")

Label(screen3, text="Order Date ", font=("Roboto lt", 10 , 'italic'), bg='black', fg='white').place(relx=.51, rely=.09, anchor="center")

Label(screen3, text="Customer Code ", font=("Roboto lt", 10 , 'italic'), bg='black', fg='white').place(relx=.895, rely=.09, anchor="center")

ttk.Entry(screen3, textvar=customer\_code , width='15').place(relx=.89, rely=.130, anchor="center")

Button(screen3, image = search\_button, border = 0, command = fetch\_order).place(relx=.5, rely=.2, anchor="center")

Label(screen3, text="Search Result",width='500', font=("Roboto lt", 10 , 'italic'), bg='black', fg='white',anchor=W).place(relx=0, rely=.275,anchor='w')

Label(screen3, text="Order Number : ",width='50', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.35,anchor='center')

Label(screen3, text="Advance Amount : ",width='50', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.42,anchor='center')

Label(screen3, text="Order ID : ",width='50', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.49,anchor='center')

Label(screen3, text="Order Date : ",width='50', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.56,anchor='center')

Label(screen3, text="Customer Code : ",width='50', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.63,anchor='center')

Label(screen3, text="Agent Code : ",width='50', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.70,anchor='center')

Label(screen3, text="Order Description : ",width='50', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.77,anchor='center')

Button(screen3, image = back, border = 0, command=screen3.destroy).place(relx=.5, rely=.95, anchor="center")

screen3.mainloop()

def fetch\_order():

global screen3 ,order\_year,order\_month,order\_day,order\_code,customer\_code

orderDateObj = datetime.strptime(order\_year.get() + "-" + order\_month.get() + "-" + order\_day.get(), '%Y-%m-%d')

connection = pymysql.connect(host="localhost", user="root", passwd="", database="sales") # database connection

cursor = connection.cursor()

select\_query = "SELECT \* FROM orders where ORD\_NUM = '" + order\_code.get() + "' OR ORD\_DATE = '" + orderDateObj.strftime("%Y-%m-%d") + "' OR CUST\_CODE = '" + customer\_code.get() + "';" # queries for retrieving values

cursor.execute(select\_query) # executing the queries

order\_info = cursor.fetchall()

connection.commit() # commiting the connection then closing it.

connection.close() # closing the connection of the database

if order\_info:

Label(screen3, text=order\_info[0][0],width='10', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=E).place(relx=.8, rely=.35,anchor='e')

Label(screen3, text=order\_info[0][1],width='10', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=E).place(relx=.8, rely=.42,anchor='e')

Label(screen3, text=order\_info[0][2],width='10', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=E).place(relx=.8, rely=.49,anchor='e')

Label(screen3, text=order\_info[0][3],width='10', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=E).place(relx=.8, rely=.56,anchor='e')

Label(screen3, text=order\_info[0][4],width='10', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=E).place(relx=.8, rely=.63,anchor='e')

Label(screen3, text=order\_info[0][5],width='10', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=E).place(relx=.8, rely=.70,anchor='e')

Label(screen3, text=order\_info[0][6],width='10', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=E).place(relx=.8, rely=.77,anchor='e')

Label(screen3, text=" Record Found ", width='100', height="1", font=("Ariel", 20,), fg='green', bg='white').place(relx=.5, rely=.85,anchor='center')

else:

Label(screen3, text="Order Number : ",width='50', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.35,anchor='center')

Label(screen3, text="Advance Amount : ",width='50', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.42,anchor='center')

Label(screen3, text="Order ID : ",width='50', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.49,anchor='center')

Label(screen3, text="Order Date : ",width='50', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.56,anchor='center')

Label(screen3, text="Customer Code : ",width='50', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.63,anchor='center')

Label(screen3, text="Agent Code : ",width='50', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.70,anchor='center')

Label(screen3, text="Order Description : ",width='50', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.77,anchor='center')

Label(screen3, text=" Record Not Found ", width='100', height="1", font=("Ariel", 20,), fg='red', bg='black').place(relx=.5, rely=.85,anchor='center')

def update\_order():

global screen4\_1, ord\_no , adv\_amt , ord\_amt , cust\_code , agt\_code , ord\_des , order\_year\_1 ,order\_month\_1 , order\_day\_1

ord\_no , adv\_amt , ord\_amt , order\_year\_1 , order\_month\_1 , order\_day\_1 , cust\_code , agt\_code , ord\_des = StringVar(),StringVar(),StringVar(),StringVar(),StringVar(),StringVar(),StringVar(),StringVar(),StringVar()

screen4\_1 = Toplevel(screen)

screen4\_1.title("Update Order")

adjustWindow(screen4\_1) # configuring the window

back = PhotoImage(file="back.png")

add\_button = PhotoImage(file="add.png")

image1= PhotoImage(file="bg\_1.png")

label\_for\_image= Label(screen4\_1, image=image1)

label\_for\_image.place(x=0, y=0, relwidth=1, relheight=1)

Label(screen4\_1, text="Add New Order ", width='500', height="1", font=("Ariel", 20,), fg='white', bg='#43315E', anchor =W).pack()

Label(screen4\_1, text="Order Number : ",width='50', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.25,anchor='center')

ttk.Entry(screen4\_1, textvar= ord\_no , width='40').place(relx=.66, rely=.25, anchor="center")

Label(screen4\_1, text="Order Amount : ",width='50', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.32,anchor='center')

ttk.Entry(screen4\_1, textvar= ord\_amt , width='40').place(relx=.66, rely=.32, anchor="center")

Label(screen4\_1, text="Advanced Amount : ",width='50', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.39,anchor='center')

ttk.Entry(screen4\_1, textvar= adv\_amt , width='40').place(relx=.66, rely=.39, anchor="center")

Label(screen4\_1, text="Order Date : ",width='50', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.46,anchor='center')

choices = list(range(1950,2051))

ttk.Combobox(screen4\_1 , width=5, values = choices ,textvariable = order\_year\_1 ).place(relx=.55, rely=.46, anchor="center")

choices = list(range(1,13))

ttk.Combobox(screen4\_1 , width=5, values = choices ,textvariable = order\_month\_1 ).place(relx=.65, rely=.46, anchor="center")

choices = list(range(1,32))

ttk.Combobox(screen4\_1 , width=5, values = choices ,textvariable = order\_day\_1 ).place(relx=.75, rely=.46, anchor="center")

order\_year\_1.set("2020")

order\_month\_1.set("1")

order\_day\_1.set("1")

Label(screen4\_1, text="Customer Code : ",width='50', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.53,anchor='center')

ttk.Entry(screen4\_1, textvar= cust\_code , width='40').place(relx=.66, rely=.53, anchor="center")

Label(screen4\_1, text="Agent Code : ",width='50', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.60,anchor='center')

ttk.Entry(screen4\_1, textvar= agt\_code , width='40').place(relx=.66, rely=.60, anchor="center")

Label(screen4\_1, text="Order Description : ",width='50', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.67,anchor='center')

ttk.Entry(screen4\_1, textvar= ord\_des , width='40').place(relx=.66, rely=.67, anchor="center")

Button(screen4\_1, image = add\_button, border = 0, command = update\_order\_db).place(relx=.5, rely=.8, anchor="center")

Button(screen4\_1, image = back, border = 0, command=screen4\_1.destroy).place(relx=.5, rely=.9, anchor="center")

Label(screen4\_1, text=" Status", fg="white", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

screen4\_1.mainloop()

def update\_order\_db():

if ord\_no.get() and adv\_amt.get() and ord\_amt.get() and cust\_code.get() and agt\_code.get() and ord\_des.get() and order\_year\_1.get() and order\_month\_1.get() and order\_day\_1.get():

orderDateObj\_1 = datetime.strptime(order\_year\_1.get() + "-" + order\_month\_1.get() + "-" + order\_day\_1.get(), '%Y-%m-%d')

if len(ord\_no.get()) == 6 and ord\_no.get().isdigit():

pass

else:

Label(screen4\_1, text="⚠ Order number should have 6 digits ", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

if ord\_amt.get() and ord\_amt.get().isdigit():

if adv\_amt.get() and adv\_amt.get().isdigit():

if order\_year\_1.get() and order\_month\_1.get() and order\_day\_1.get():

if len(cust\_code.get()) == 6 and cust\_code.get().isalnum():

connection = pymysql.connect(host="localhost", user="root", passwd="", database="sales") # database connection

cursor = connection.cursor()

select\_query = "SELECT \* FROM customer where CUST\_CODE = '" + cust\_code.get() + "';" # queries for retrieving values

cursor.execute(select\_query) # executing the queries

cust\_code\_info = cursor.fetchall()

connection.commit() # commiting the connection then closing it.

connection.close()

if cust\_code\_info:

pass

else:

Label(screen4\_1, text="⚠ Customer Code not found in the Database, Enter Valid Customer Code", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

if len(agt\_code.get()) == 4 and agt\_code.get().isalnum():

connection = pymysql.connect(host="localhost", user="root", passwd="", database="sales") # database connection

cursor = connection.cursor()

select\_query = "SELECT \* FROM agents where AGENT\_CODE = '" + agt\_code.get() + "';" # queries for retrieving values

cursor.execute(select\_query) # executing the queries

agent\_code\_info = cursor.fetchall()

connection.commit() # commiting the connection then closing it.

connection.close()

if agent\_code\_info:

pass

else:

Label(screen4\_1, text="⚠ Agent Code not found in the Database, Enter Valid Agent Code", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

if all(x.isalpha() or x.isspace() for x in ord\_des.get()) and (len(ord\_des.get()) > 0):

connection = pymysql.connect(host="localhost", user="root", passwd="", database="sales") # database connection

cursor = connection.cursor()

insert\_query = "INSERT INTO orders (ORD\_NUM, ORD\_AMOUNT, ADVANCE\_AMOUNT, ORD\_DATE, CUST\_CODE, AGENT\_CODE, ORD\_DESCRIPTION) VALUES('"+ ord\_no.get() + "', '"+ ord\_amt.get() + "', '"+ adv\_amt.get() + "', '" + orderDateObj\_1.strftime("%Y-%m-%d") + "', '"+ cust\_code.get() + "', '" + agt\_code.get() + "', '"+ ord\_des.get() +"' );" # fullname.get() + "', '"+ email.get() + "', '"+ password.get() + "', '"+ company\_id.get() + "', '"+ company\_name.get() + "', '"+ company\_city.get() + "' );" # queries for inserting values

cursor.execute(insert\_query) # executing the queries

connection.commit() # commiting the connection then closing it.

connection.close() # closing the connection of the database

Label(screen4\_1, text="New Order Added Successfully ", bg="white", font=("Roboto", 11), width='3000', anchor=W, fg='green').place(x=0, y=570) # printing successful registration message

else:

Label(screen4\_1, text="⚠ Order Description should not have numbers or special characters", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

else:

Label(screen4\_1, text="⚠ Agent Code should have 4 alphanumeric characters ", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

else:

Label(screen4\_1, text="⚠ Custumer Code should have 6 alphanumeric characters ", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

else:

Label(screen4\_1, text="⚠ Please Enter Order Date ", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

else:

Label(screen4\_1, text="⚠ Advanced Amount should have numbers only ", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

else:

Label(screen4\_1, text="⚠ Order Amount should have numbers only", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

else:

Label(screen4\_1, text="⚠ Please Fill All The Details ", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

def update\_agent():

global screen4\_2 , agt\_code\_1 , agt\_name , work\_area , commission , agt\_phone

agt\_code\_1 , agt\_name , work\_area , commission , agt\_phone = StringVar(),StringVar(),StringVar(),StringVar(),StringVar()

screen4\_2 = Toplevel(screen)

screen4\_2.title("Update Agent")

adjustWindow(screen4\_2) # configuring the window

back = PhotoImage(file="back.png")

add\_button = PhotoImage(file="add.png")

image1= PhotoImage(file="bg\_1.png")

label\_for\_image= Label(screen4\_2, image=image1)

label\_for\_image.place(x=0, y=0, relwidth=1, relheight=1)

Label(screen4\_2, text="Add New Agent ", width='500', height="1", font=("Ariel", 20,), fg='white', bg='#43315E', anchor =W).pack()

Label(screen4\_2, text="Agent Code : ",width='50', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.35,anchor='center')

ttk.Entry(screen4\_2, textvar= agt\_code\_1 , width='40').place(relx=.66, rely=.35, anchor="center")

Label(screen4\_2, text="Agent Name : ",width='50', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.42,anchor='center')

ttk.Entry(screen4\_2, textvar= agt\_name , width='40').place(relx=.66, rely=.42, anchor="center")

Label(screen4\_2, text="Working Area : ",width='50', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.49,anchor='center')

ttk.Entry(screen4\_2, textvar= work\_area , width='40').place(relx=.66, rely=.49, anchor="center")

Label(screen4\_2, text="Commission : ",width='50', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.56,anchor='center')

ttk.Entry(screen4\_2, textvar= commission , width='40').place(relx=.66, rely=.56, anchor="center")

Label(screen4\_2, text="Phone number : ",width='50', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.63,anchor='center')

ttk.Entry(screen4\_2, textvar= agt\_phone , width='40').place(relx=.66, rely=.63, anchor="center")

Button(screen4\_2, image = add\_button, border = 0, command = update\_agent\_db).place(relx=.5, rely=.8, anchor="center")

Button(screen4\_2, image = back, border = 0, command=screen4\_2.destroy).place(relx=.5, rely=.9, anchor="center")

Label(screen4\_2, text=" Status", fg="white", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

screen4\_2.mainloop()

def update\_agent\_db():

if agt\_code\_1.get() and agt\_name.get() and work\_area.get() and commission.get() and agt\_phone.get():

if len(agt\_code\_1.get()) == 4 and agt\_code\_1.get().isalnum():

if all(x.isalpha() or x.isspace() for x in agt\_name.get()) and (len(agt\_name.get()) > 0):

if all(x.isalpha() or x.isspace() for x in work\_area.get()) and (len(work\_area.get()) > 0):

if commission.get().isdigit():

if int(commission.get()) > 0 and int(commission.get()) < 99:

comm = '0.'+ commission.get()

pass

else:

Label(screen4\_2, text="⚠ Commission Percentage should be in the range of 0 - 99", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

if len(agt\_phone.get()) == 10 and agt\_phone.get().isdigit():

connection = pymysql.connect(host="localhost", user="root", passwd="", database="sales") # database connection

cursor = connection.cursor()

insert\_query = "INSERT INTO agents (AGENT\_CODE, AGENT\_NAME, WORKING\_AREA, COMMISSION, PHONE\_NO) VALUES('"+ agt\_code\_1.get() + "', '"+ agt\_name.get() + "', '"+ work\_area.get() + "', '" + comm + "', '" + agt\_phone.get() + "' );" # fullname.get() + "', '"+ email.get() + "', '"+ password.get() + "', '"+ company\_id.get() + "', '"+ company\_name.get() + "', '"+ company\_city.get() + "' );" # queries for inserting values

cursor.execute(insert\_query) # executing the queries

connection.commit() # commiting the connection then closing it.

connection.close() # closing the connection of the database

Label(screen4\_2, text="New Agent Added Successfully ", bg="white", font=("Roboto", 11), width='3000', anchor=W, fg='green').place(x=0, y=570) # printing successful registration message

pass

else:

Label(screen4\_2, text="⚠ Agent Phone Number should have 10 Numbers", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

else:

Label(screen4\_2, text="⚠ Commission should have numbers only", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

else:

Label(screen4\_2, text="⚠ Name of the area cannot have numbers or special characters", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

else:

Label(screen4\_2, text="⚠ Name of the agent cannot have numbers or special characters", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

else:

Label(screen4\_2, text="⚠ Agent Code should have 4 alphanumeric characters", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

else:

Label(screen4\_2, text="⚠ Please Fill All The Details", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

def update\_customer():

global screen4\_3 , cust\_code\_2, cust\_name, cust\_city, cust\_work\_area, cust\_country, grade, opn\_amt, rec\_amt, pay\_amt, out\_amt, cust\_phone , agt\_code\_2

cust\_code\_2, cust\_name, cust\_city, cust\_work\_area, cust\_country, grade, opn\_amt, rec\_amt, pay\_amt, out\_amt, cust\_phone , agt\_code\_2 = StringVar(),StringVar(),StringVar(),StringVar(),StringVar(),StringVar(),StringVar(),StringVar(),StringVar(),StringVar(),StringVar(),StringVar()

screen4\_3 = Toplevel(screen)

screen4\_3.title("Update Customer")

adjustWindow(screen4\_3) # configuring the window

back = PhotoImage(file="back.png")

add\_button = PhotoImage(file="add.png")

image1= PhotoImage(file="bg\_1.png")

label\_for\_image= Label(screen4\_3, image=image1)

label\_for\_image.place(x=0, y=0, relwidth=1, relheight=1)

Label(screen4\_3, text="Add New Customer ", width='500', height="1", font=("Ariel", 20,), fg='white', bg='#43315E', anchor =W).pack()

Label(screen4\_3, text="Customer Code : ",width='50', font=("Roboto ", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.15,anchor='center')

ttk.Entry(screen4\_3, textvar= cust\_code\_2 , width='40').place(relx=.7, rely=.15, anchor="center")

Label(screen4\_3, text="Customer Name : ",width='50', font=("Roboto ", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.20,anchor='center')

ttk.Entry(screen4\_3, textvar= cust\_name , width='40').place(relx=.7, rely=.20, anchor="center")

Label(screen4\_3, text="Customer City : ",width='50', font=("Roboto ", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.25,anchor='center')

ttk.Entry(screen4\_3, textvar= cust\_city , width='40').place(relx=.7, rely=.25, anchor="center")

Label(screen4\_3, text="Working Area : ",width='50', font=("Roboto ", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.30,anchor='center')

ttk.Entry(screen4\_3, textvar= cust\_work\_area , width='40').place(relx=.7, rely=.30, anchor="center")

Label(screen4\_3, text="Customer Country : ",width='50', font=("Roboto ", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.35,anchor='center')

ttk.Entry(screen4\_3, textvar= cust\_country , width='40').place(relx=.7, rely=.35, anchor="center")

Label(screen4\_3, text="Grade : ",width='50', font=("Roboto ", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.40,anchor='center')

ttk.Entry(screen4\_3, textvar= grade , width='40').place(relx=.7, rely=.40, anchor="center")

Label(screen4\_3, text="Opening Amount : ",width='50', font=("Roboto ", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.45,anchor='center')

ttk.Entry(screen4\_3, textvar= opn\_amt , width='40').place(relx=.7, rely=.45, anchor="center")

Label(screen4\_3, text="Recieved Amount : ",width='50', font=("Roboto ", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.50,anchor='center')

ttk.Entry(screen4\_3, textvar= rec\_amt , width='40').place(relx=.7, rely=.50, anchor="center")

Label(screen4\_3, text="Payment Amount : ",width='50', font=("Roboto ", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.55,anchor='center')

ttk.Entry(screen4\_3, textvar= pay\_amt , width='40').place(relx=.7, rely=.55, anchor="center")

Label(screen4\_3, text="Outstanding Amount : ",width='50', font=("Roboto ", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.60,anchor='center')

ttk.Entry(screen4\_3, textvar= out\_amt , width='40').place(relx=.7, rely=.60, anchor="center")

Label(screen4\_3, text="Phone Number : ",width='50', font=("Roboto ", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.65,anchor='center')

ttk.Entry(screen4\_3, textvar= cust\_phone , width='40').place(relx=.7, rely=.65, anchor="center")

Label(screen4\_3, text="Agent Code : ",width='50', font=("Roboto ", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.70,anchor='center')

ttk.Entry(screen4\_3, textvar= agt\_code\_2 , width='40').place(relx=.7, rely=.70, anchor="center")

Button(screen4\_3, image = add\_button, border = 0, command = update\_customer\_db).place(relx=.5, rely=.8, anchor="center")

Button(screen4\_3, image = back, border = 0, command=screen4\_3.destroy).place(relx=.5, rely=.9, anchor="center")

Label(screen4\_3, text=" Status ", fg="white", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

screen4\_3.mainloop()

def update\_customer\_db():

if cust\_code\_2 and cust\_name and cust\_city and cust\_work\_area and cust\_country and grade and opn\_amt and rec\_amt and pay\_amt and out\_amt and cust\_phone and agt\_code\_2:

if len(cust\_code\_2.get()) == 6 and cust\_code\_2.get().isalnum():

if all(x.isalpha() or x.isspace() for x in cust\_name.get()) and (len(cust\_name.get()) > 0):

if all(x.isalpha() or x.isspace() for x in cust\_city.get()) and (len(cust\_city.get()) > 0):

if all(x.isalpha() or x.isspace() for x in cust\_work\_area.get()) and (len(cust\_work\_area.get()) > 0):

if all(x.isalpha() or x.isspace() for x in cust\_country.get()) and (len(cust\_country.get()) > 0):

if grade.get().isdigit():

if int(grade.get()) > 0 and int(grade.get()) < 4:

pass

else:

Label(screen4\_3, text="⚠ Commission should be in the range of 1 - 3", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

if opn\_amt.get().isdigit():

if rec\_amt.get().isdigit():

if pay\_amt.get().isdigit():

if out\_amt.get().isdigit():

if len(cust\_phone.get()) == 10 and cust\_phone.get().isdigit():

if len(agt\_code\_2.get()) == 4 and agt\_code\_2.get().isalnum():

connection = pymysql.connect(host="localhost", user="root", passwd="", database="sales") # database connection

cursor = connection.cursor()

select\_query = "SELECT \* FROM agents where AGENT\_CODE = '" + agt\_code\_2.get() + "';" # queries for retrieving values

cursor.execute(select\_query) # executing the queries

agent\_code\_info = cursor.fetchall()

connection.commit() # commiting the connection then closing it.

connection.close()

if agent\_code\_info:

pass

else:

Label(screen4\_3, text="⚠ Agent Code not found in the Database, Enter Valid Agent Code", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

connection = pymysql.connect(host="localhost", user="root", passwd="", database="sales") # database connection

cursor = connection.cursor()

insert\_query = "INSERT INTO customer (CUST\_CODE, CUST\_NAME, CUST\_CITY, WORKING\_AREA, CUST\_COUNTRY, GRADE, OPENING\_AMT, RECEIVE\_AMT, PAYMENT\_AMT, OUTSTANDING\_AMT, PHONE\_NO, AGENT\_CODE) VALUES('" + cust\_code\_2.get() + "', '"+ cust\_name.get() + "', '"+ cust\_city.get() + "', '"+ cust\_work\_area.get() + "', '"+ cust\_country.get() + "', '"+ grade.get() + "', '"+ opn\_amt.get() + "', '"+ rec\_amt.get() + "', '"+ pay\_amt.get() + "', '"+ out\_amt.get() + "', '"+ cust\_phone.get() + "', '"+ agt\_code\_2.get()+ "' );"

cursor.execute(insert\_query) # executing the queries

connection.commit() # commiting the connection then closing it.

connection.close() # closing the connection of the database

Label(screen4\_3, text="New Customer Added Successfully ", bg="white", font=("Roboto", 11), width='3000', anchor=W, fg='green').place(x=0, y=570) # printing successful registration message

else:

Label(screen4\_3, text="⚠ Agent Code should have 4 alphanumeric characters", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

else:

Label(screen4\_3, text="⚠ Customer Phone Number should have 10 numbers", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

else:

Label(screen4\_3, text="⚠ Outstanding Amount should not have alphabets or special characters", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

else:

Label(screen4\_3, text="⚠ Payment Amount should not have alphabets or special characters", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

else:

Label(screen4\_3, text="⚠ Recieved Amount should not have alphabets or special characters", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

else:

Label(screen4\_3, text="⚠ Opening Amount should not have alphabets or special characters", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

else:

Label(screen4\_3, text="⚠ Grade should not have alphabets or special characters", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

else:

Label(screen4\_3, text="⚠ Country Name should not have numbers or special characters", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

else:

Label(screen4\_3, text="⚠ Work Area Name should not have numbers or special characters", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

else:

Label(screen4\_3, text="⚠ City Name should not have numbers or special characters", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

else:

Label(screen4\_3, text="⚠ Customer Name should not have numbers or special characters", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

else:

Label(screen4\_3, text="⚠ Customer Code should have 6 alphanumeric characters", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

else:

Label(screen4\_3, text="⚠ Please fill all the details", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

def update\_company():

global screen4\_4 , company\_id\_1 , company\_name\_1 , company\_city\_1

company\_id\_1 , company\_name\_1 , company\_city\_1 = StringVar(),StringVar(),StringVar()

screen4\_4 = Toplevel(screen)

screen4\_4.title("Update Company")

adjustWindow(screen4\_4) # configuring the window

back = PhotoImage(file="back.png")

add\_button = PhotoImage(file="add.png")

image1= PhotoImage(file="bg\_1.png")

label\_for\_image= Label(screen4\_4, image=image1)

label\_for\_image.place(x=0, y=0, relwidth=1, relheight=1)

Label(screen4\_4, text="Add New Company ", width='500', height="1", font=("Ariel", 20,), fg='white', bg='#43315E', anchor =W).pack()

Label(screen4\_4, text="Company ID : ",width='50', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.35,anchor='center')

ttk.Entry(screen4\_4, textvar= company\_id\_1 , width='40').place(relx=.66, rely=.35, anchor="center")

Label(screen4\_4, text="Company Name : ",width='50', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.42,anchor='center')

ttk.Entry(screen4\_4, textvar= company\_name\_1 , width='40').place(relx=.66, rely=.42, anchor="center")

Label(screen4\_4, text="City : ",width='50', font=("Roboto lt", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.49,anchor='center')

ttk.Entry(screen4\_4, textvar= company\_city\_1 , width='40').place(relx=.66, rely=.49, anchor="center")

Button(screen4\_4, image = add\_button, border = 0, command = update\_company\_db).place(relx=.5, rely=.8, anchor="center")

Button(screen4\_4, image = back, border = 0, command=screen4\_4.destroy).place(relx=.5, rely=.9, anchor="center")

Label(screen4\_4, text=" Status", fg="white", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

screen4\_4.mainloop()

def update\_company\_db():

if company\_id\_1.get() and company\_name\_1.get() and company\_city\_1.get():

if company\_id\_1.get().isdigit() and int(company\_id\_1.get())>0 and int(company\_id\_1.get())<99:

if all(x.isalpha() or x.isspace() for x in company\_name\_1.get()) and (len(company\_name\_1.get()) > 0):

if all(x.isalpha() or x.isspace() for x in company\_city\_1.get()) and (len(company\_city\_1.get()) > 0):

connection = pymysql.connect(host="localhost", user="root", passwd="", database="sales") # database connection

cursor = connection.cursor()

insert\_query = "INSERT INTO company (COMPANY\_ID, COMPANY\_NAME, COMPANY\_CITY) VALUES('" + company\_id\_1.get() + "', '"+ company\_name\_1.get() + "', '"+ company\_city\_1.get() + "' );"

cursor.execute(insert\_query) # executing the queries

connection.commit() # commiting the connection then closing it.

connection.close() # closing the connection of the database

Label(screen4\_4, text="New Company Added Successfully ", bg="white", font=("Roboto", 11), width='3000', anchor=W, fg='green').place(x=0, y=570)

pass

else:

Label(screen4\_4, text=" ⚠ Company Name should not have numbers or special characters", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

else:

Label(screen4\_4, text=" ⚠ City Name should not have numbers or special characters", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

else:

Label(screen4\_4, text=" ⚠ Company ID should numbers in range 1-99", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

else:

Label(screen4\_4, text=" ⚠ Please Fill All The Details", fg="red", font=( "Roboto", 11), width='3000', anchor=W, bg='black').place(x=0, y=570)

return

def balance\_report():

global screen5\_1

bal = []

screen5\_1 = Toplevel(screen)

screen5\_1.title("Balance Amount Report")

adjustWindow(screen5\_1) # configuring the window

back = PhotoImage(file="back.png")

image1= PhotoImage(file="bg\_1.png")

label\_for\_image= Label(screen5\_1, image=image1)

label\_for\_image.place(x=0, y=0, relwidth=1, relheight=1)

Label(screen5\_1, text="Balance Amount Report ", width='500', height="1", font=("Ariel", 20,), fg='white', bg='#43315E', anchor =W).pack()

connection = pymysql.connect(host="localhost", user="root", passwd="", database="sales") # database connection

cursor = connection.cursor()

select\_query = "SELECT \* ,(ORD\_AMOUNT-ADVANCE\_AMOUNT) AS BALANCE\_AMT FROM orders LEFT JOIN agents ON orders.AGENT\_CODE=agents.AGENT\_CODE ORDER BY `BALANCE\_AMT` DESC ;"

cursor.execute(select\_query) # executing the queries

bal = cursor.fetchall()

alpha = [0,13,5,8]

k = 0

new\_li = [[0 for x in range(4)] for y in range(len(bal))]

for i in range(len(bal)):

for j in alpha:

new\_li[i][k] = bal[i][j]

k+=1

if k == 4:

k=k-4

connection.commit() # commiting the connection then closing it.

connection.close() # closing the connection of the database

tv = ttk.Treeview(screen5\_1, columns=(1,2,3,4), show = "headings", height = '20')

tv.place(relx=.5, rely=.5, anchor="center")

verscrlbar = ttk.Scrollbar(screen5\_1,

orient ="vertical",

command = tv.yview)

verscrlbar.place(relx=.9, rely=.175, height=425)

tv.configure(yscrollcommand = verscrlbar.set)

tv.column(1, width=130, minwidth=130, stretch=NO)

tv.column(2, width=130, minwidth=130, stretch=NO)

tv.column(3, width=80, minwidth=80)

tv.column(4, width=150, minwidth=150, stretch=NO)

tv.heading(1,text="Order Number",anchor = W)

tv.heading(2, text="Balance Amount",anchor = W)

tv.heading(3, text="Agent Code",anchor = W)

tv.heading(4, text="Agent Name",anchor = W)

for item in new\_li:

tv.insert('', 'end', values = item)

Button(screen5\_1, image = back, border = 0, command=screen5\_1.destroy).place(relx=.5, rely=.95, anchor="center")

screen5\_1.mainloop()

def cumulative\_data():

global screen5\_2

country\_names = []

screen5\_2 = Toplevel(screen)

screen5\_2.title("Cumulative Data Report")

adjustWindow(screen5\_2) # configuring the window

back = PhotoImage(file="back.png")

image1= PhotoImage(file="bg\_1.png")

label\_for\_image= Label(screen5\_2, image=image1)

label\_for\_image.place(x=0, y=0, relwidth=1, relheight=1)

Label(screen5\_2, text="Cumulative Data Report ", width='500', height="1", font=("Ariel", 20,), fg='white', bg='#43315E', anchor =W).pack()

Label(screen5\_2, text="Cumulative Data of All Customers",width='500', font=("Roboto lt", 10 , 'italic'), bg='black', fg='white',anchor=W).place(relx=0, rely=.2,anchor='w')

Label(screen5\_2, text="Payment Amount : ",width='50', font=("Roboto ", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.3,anchor='center')

Label(screen5\_2, text="Outstanding Amount : ",width='50', font=("Roboto ", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.4,anchor='center')

Label(screen5\_2, text="Country with Maximum number of Customers",width='500', font=("Roboto ", 10 , 'italic'), bg='black', fg='white',anchor=W).place(relx=0, rely=.5,anchor='w')

Label(screen5\_2, text="Country Name : ",width='50', font=("Roboto ", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.6,anchor='center')

Label(screen5\_2, text="Number of Customers : ",width='50', font=("Roboto ", 13 , 'italic'), bg='black', fg='white',anchor=W).place(relx=.5, rely=.7,anchor='center')

connection = pymysql.connect(host="localhost", user="root", passwd="", database="sales") # database connection

cursor = connection.cursor()

select\_query = "SELECT SUM(`PAYMENT\_AMT`) FROM `customer` "# queries for retrieving values

cursor.execute(select\_query) # executing the queries

cum\_payment\_amt = cursor.fetchall()

select\_query = "SELECT SUM(`OUTSTANDING\_AMT`) FROM `customer` "# queries for retrieving values

cursor.execute(select\_query)

cum\_outstanding\_amt = cursor.fetchall()

select\_query = "SELECT `CUST\_COUNTRY` FROM `customer`"

cursor.execute(select\_query)

c\_name = cursor.fetchall()

connection.commit() # commiting the connection then closing it.

connection.close()

Label(screen5\_2, text= cum\_payment\_amt[0][0],width='20', font=("Roboto ", 13 , 'italic'),fg='black', bg='white',anchor=E).place(relx=.9, rely=.3,anchor='e')

Label(screen5\_2, text= cum\_outstanding\_amt[0][0] ,width='20', font=("Roboto ", 13 , 'italic'), fg='black', bg='white',anchor=E).place(relx=.9, rely=.4,anchor='e')

for i in range(len(c\_name)):

country\_names.append(c\_name[i][0])

c=Counter(country\_names).most\_common(1)

Label(screen5\_2, text= c[0][0],width='20', font=("Roboto ", 13 , 'italic'), fg='black', bg='white',anchor=E).place(relx=.9, rely=.6,anchor='e')

Label(screen5\_2, text= c[0][1] ,width='20', font=("Roboto ", 13 , 'italic'),fg='black', bg='white',anchor=E).place(relx=.9, rely=.7,anchor='e')

Button(screen5\_2, image = back, border = 0, command=screen5\_2.destroy).place(relx=.5, rely=.95, anchor="center")

screen5\_2.mainloop()

def insight\_options():

global insight\_menu

insight\_menu = Toplevel(screen)

insight\_menu.title("Insights Options")

adjustWindow(insight\_menu) # configuring the window

back,show = PhotoImage(file="back.png"),PhotoImage(file="show.png")

i1,i2,i3,i4,i5,i6,i7 = PhotoImage(file="i\_1.png"),PhotoImage(file="i\_2.png"),PhotoImage(file="i\_3.png"),PhotoImage(file="i\_4.png"),PhotoImage(file="i\_5.png"),PhotoImage(file="i\_6.png"),PhotoImage(file="i\_7.png")

image1= PhotoImage(file="bg\_1.png")

label\_for\_image= Label(insight\_menu, image=image1)

label\_for\_image.place(x=0, y=0, relwidth=1, relheight=1)

Label(insight\_menu, text="Insights", width='500', height="1", font=("Ariel", 20,), fg='white', bg='#43315E', anchor =W).pack()

Button(insight\_menu, image = i1, border = 0 ,command=insight\_1).place(relx=.5, rely=.2, anchor="center")

Button(insight\_menu, image = i2, border = 0 ,command=insight\_2).place(relx=.5, rely=.3, anchor="center")

Button(insight\_menu, image = i3, border = 0 ,command=insight\_3).place(relx=.5, rely=.4, anchor="center")

Button(insight\_menu, image = i4, border = 0 ,command=insight\_4).place(relx=.5, rely=.5, anchor="center")

Button(insight\_menu, image = i5, border = 0 ,command=insight\_5).place(relx=.5, rely=.6, anchor="center")

Button(insight\_menu, image = i6, border = 0 ,command=insight\_6).place(relx=.5, rely=.7, anchor="center")

Button(insight\_menu, image = i7, border = 0 ,command=insight\_7).place(relx=.5, rely=.8, anchor="center")

Button(insight\_menu, image = back, border = 0, command=insight\_menu.destroy).place(relx=.5, rely=.95, anchor="center")

insight\_menu.mainloop()

def insight\_1():

global insight\_1\_screen

insight\_1\_screen = Toplevel(screen)

insight\_1\_screen.title("Total Property Area Sold vs Leased ")

adjustWindow(insight\_1\_screen) # configuring the window

back = PhotoImage(file="back.png")

image1= PhotoImage(file="bg\_1.png")

label\_for\_image= Label(insight\_1\_screen, image=image1)

label\_for\_image.place(x=0, y=0, relwidth=1, relheight=1)

Label(insight\_1\_screen, text="Total Property Area Sold vs Leased ", width='500', height="1", font=("Ariel", 20,), fg='white', bg='#43315E', anchor =W).pack()

owned= df[(df['Tenure']=='Owned') & (df['UoM']=='SQ-M')]

sumo=list(owned.groupby(['Year'])['Area'].sum())

leased= df[(df['Tenure']=='Leased') & (df['UoM']=='SQ-M')]

suml=list(leased.groupby(['Year'])['Area'].sum())

years=sorted(df['Year'].unique())

figure1 = plt.Figure(figsize=(7,5), dpi=85)

ax1 = figure1.add\_subplot(111)

bar1 = FigureCanvasTkAgg(figure1, insight\_1\_screen)

bar1.draw()

bar1.get\_tk\_widget().place(relx=.5,rely=.5,anchor='center')

df1 = pd.DataFrame({'Sold': sumo,'Leased': suml}, index=years)

df1.plot(kind='bar', ax=ax1,title='Total Property Area Sold vs Total Property Area Leased (in SQ-M only)')#,autopct='%1.2f%%',startangle=0,subplots=True,legend=False)

toolbarFrame = Frame(master=insight\_1\_screen)

toolbarFrame.place(relx=.25,rely=.85)

toolbar = NavigationToolbar2Tk(bar1, toolbarFrame)

toolbar.update()

Button(insight\_1\_screen, image = back, border = 0, command=insight\_1\_screen.destroy).place(relx=.5, rely=.95, anchor="center")

insight\_1\_screen.mainloop()

def insight\_2():

global insight\_2\_screen,countryselect

insight\_2\_screen = Toplevel(screen)

insight\_2\_screen.title("Year With Maximum Leased Area ")

adjustWindow(insight\_2\_screen) # configuring the window

back,show2 = PhotoImage(file="back.png"),PhotoImage(file="show\_2.png")

image1= PhotoImage(file="bg\_1.png")

label\_for\_image= Label(insight\_2\_screen, image=image1)

label\_for\_image.place(x=0, y=0, relwidth=1, relheight=1)

Label(insight\_2\_screen, text="Year With Maximum Leased Area ", width='500', height="1", font=("Ariel", 20,), fg='white', bg='#43315E', anchor =W).pack()

countryselect=StringVar()

country = ['Country','CA','WS']

droplist = ttk.OptionMenu(insight\_2\_screen, countryselect, \*country)

droplist.config(width=12)

countryselect.set('CA')

droplist.place(relx=.3, rely=.1, anchor="center")

Button(insight\_2\_screen, image = show2, border = 0, command=insight\_2\_countryselect).place(relx=.7, rely=.1, anchor="center")

Button(insight\_2\_screen, image = back, border = 0, command=insight\_2\_screen.destroy).place(relx=.5, rely=.95, anchor="center")

insight\_2\_screen.mainloop()

def insight\_2\_countryselect():

leased\_sqm = df[(df['Tenure']=='Leased') & (df['Country'] == countryselect.get())& (df['Year'] != 2020)& (df['UoM']=='SQ-M')]

leased\_ha = df[(df['Tenure']=='Leased') & (df['Country'] == countryselect.get())& (df['Year'] != 2020) & (df['UoM']=='HA')]

leased\_sqm\_sum=list(leased\_sqm.groupby(['Year'])['Area'].sum())

leased\_ha\_sum=list(leased\_ha.groupby(['Year'])['Area'].sum())

for i in range(len(leased\_ha\_sum)):

leased\_sqm\_sum[i] = leased\_sqm\_sum[i]+ (leased\_ha\_sum[i]\*10000)

years=sorted(leased\_sqm['Year'].unique())

figure1 = plt.Figure(figsize=(7,5), dpi=80)

ax1 = figure1.add\_subplot(111)

bar1 = FigureCanvasTkAgg(figure1, insight\_2\_screen)

bar1.draw()

bar1.get\_tk\_widget().place(relx=.5,rely=.5,anchor='center')

yearselect = 'CA'

df1 = pd.DataFrame({'Total Leased Area ': leased\_sqm\_sum}, index=years)

df1.plot(kind='pie', ax=ax1,autopct='%1.2f%%',startangle=0,subplots=True,legend=False,title='Total Leased Area in '+countryselect.get()+' (represented in SQ-M)')

toolbarFrame = Frame(master=insight\_2\_screen)

toolbarFrame.place(relx=.25,rely=.85)

toolbar = NavigationToolbar2Tk(bar1, toolbarFrame)

toolbar.update()

def insight\_3():

global insight\_3\_screen

insight\_3\_screen = Toplevel(screen)

insight\_3\_screen.title("Codes of Agents with OWNED Deals across the years")

adjustWindow(insight\_3\_screen) # configuring the window

back = PhotoImage(file="back.png")

image1= PhotoImage(file="bg\_1.png")

label\_for\_image= Label(insight\_3\_screen, image=image1)

label\_for\_image.place(x=0, y=0, relwidth=1, relheight=1)

Label(insight\_3\_screen, text="Agents with OWNED Deals across the years", width='500', height="1", font=("Ariel", 20,), fg='white', bg='#43315E', anchor =W).pack()

agent,count,code,temp=[],[],[],[]

df\_temp = df[(df['Tenure']=='Owned')]

a = Counter(df\_temp['Agent']).most\_common(50)

for i in a:

agent.append(i[0])

count.append(i[1])

connection = pymysql.connect(host="localhost", user="root", passwd="", database="sales") # database connection

cursor = connection.cursor()

for i in agent:

select\_query = "SELECT \* FROM agents where AGENT\_NAME = '" + i + "';" # queries for retrieving values

cursor.execute(select\_query) # executing the queries

info = cursor.fetchall()

if info:

temp.append(info[0][0])

i=i+' ( '+info[0][0]+')'

code.append(i)

while(len(count)!=len(code)):

count.pop()

connection.commit() # commiting the connection then closing it.

connection.close()

df1 = pd.DataFrame({'Agent Code': temp,'OWNED Deals': count})

figure1 = plt.Figure(figsize=(8,6.5), dpi=70)

ax1 = figure1.add\_subplot(111)

bar1 = FigureCanvasTkAgg(figure1,insight\_3\_screen)

bar1.draw()

bar1.get\_tk\_widget().place(relx=.5,rely=.475,anchor='center')

df1 = df1[['Agent Code','OWNED Deals']].groupby('Agent Code').sum()

df1.plot(title='Codes of Agents with OWNED Deals across the years 2017-2020',kind='bar', ax=ax1,subplots=True, legend = True)#,autopct='%1.2f%%',startangle=0)

toolbarFrame = Frame(master=insight\_3\_screen)

toolbarFrame.place(relx=.25,rely=.86)

toolbar = NavigationToolbar2Tk(bar1, toolbarFrame)

toolbar.update()

Button(insight\_3\_screen, image = back, border = 0, command=insight\_3\_screen.destroy).place(relx=.5, rely=.95, anchor="center")

insight\_3\_screen.mainloop()

def insight\_4():

global insight\_4\_screen

insight\_4\_screen = Toplevel(screen)

insight\_4\_screen.title("Agent with maximum leased deals in Chilliwack ")

adjustWindow(insight\_4\_screen) # configuring the window

back = PhotoImage(file="back.png")

image1= PhotoImage(file="bg\_1.png")

label\_for\_image= Label(insight\_4\_screen, image=image1)

label\_for\_image.place(x=0, y=0, relwidth=1, relheight=1)

Label(insight\_4\_screen, text="Maximum leased deals in Chilliwack", width='500', height="1", font=("Ariel", 20,), fg='white', bg='#43315E', anchor =W).pack()

agents,count = [],[]

chill = df[(df['City']=='Chilliwack') & (df['Tenure']=='Leased')]

chill\_agents = Counter(chill['Agent']).most\_common(50)

for i in range(len(chill\_agents)):

agents.append(chill\_agents[i][0])

count.append(chill\_agents[i][1])

figure1 = plt.Figure(figsize=(7,5), dpi=85)

ax1 = figure1.add\_subplot(111)

bar1 = FigureCanvasTkAgg(figure1, insight\_4\_screen)

bar1.draw()

bar1.get\_tk\_widget().place(relx=.5,rely=.5,anchor='center')

df1 = pd.DataFrame({'No of Leased Deals in Chilliwack ': count}, index=agents)

df1.plot(kind='bar', ax=ax1,title='Agent with maximum leased deals in Chilliwack')#,autopct='%1.2f%%',startangle=0,subplots=True)

toolbarFrame = Frame(master=insight\_4\_screen)

toolbarFrame.place(relx=.25,rely=.85)

toolbar = NavigationToolbar2Tk(bar1, toolbarFrame)

toolbar.update()

Button(insight\_4\_screen, image = back, border = 0, command=insight\_4\_screen.destroy).place(relx=.5, rely=.95, anchor="center")

insight\_4\_screen.mainloop()

def insight\_5():

global insight\_5\_screen,yearselect

yearselect=StringVar()

year = ['Year','2017','2018','2019']

insight\_5\_screen = Toplevel(screen)

insight\_5\_screen.title("Agent Performance Comparison ")

adjustWindow(insight\_5\_screen) # configuring the window

back,show = PhotoImage(file="back.png"),PhotoImage(file="show.png")

image1= PhotoImage(file="bg\_1.png")

label\_for\_image= Label(insight\_5\_screen, image=image1)

label\_for\_image.place(x=0, y=0, relwidth=1, relheight=1)

Label(insight\_5\_screen, text="Agent Performance Comparison ", width='500', height="1", font=("Ariel", 20,), fg='white', bg='#43315E', anchor =W).pack()

droplist = ttk.OptionMenu(insight\_5\_screen, yearselect, \*year)

droplist.config(width=12)

yearselect.set('2017')

droplist.place(relx=.3, rely=.1, anchor="center")

Button(insight\_5\_screen, image = show, border = 0, command=insight\_5\_yearselect).place(relx=.7, rely=.1, anchor="center")

Button(insight\_5\_screen, image = back, border = 0, command=insight\_5\_screen.destroy).place(relx=.5, rely=.95, anchor="center")

insight\_5\_screen.mainloop()

def insight\_5\_yearselect():

agent1,area1,agent2,area2=[],[],[],[]

x1 = df[(df['Tenure']=='Owned') & (df['Year'] == int(yearselect.get())) ]

y1 = df[(df['Tenure']=='Leased') & (df['Year'] == int(yearselect.get())) ]

for a,row in x1.iterrows():

if row['UoM'] == 'HA':

row['Area'] = row['Area']\*10000

area1.append(row['Area'])

agent1.append(row['Agent'])

for a,row in y1.iterrows():

if row['UoM'] == 'HA':

row['Area'] = row['Area']\*10000

area2.append(row['Area'])

agent2.append(row['Agent'])

x2 = pd.DataFrame({'Agent': agent1,'Area': area1})

y2 = pd.DataFrame({'Agent': agent2,'Area': area2})

x3=list(x2.groupby(['Agent'])['Area'].sum())

y3=list(y2.groupby(['Agent'])['Area'].sum())

if yearselect.get()=='2019':

y3.insert(-3,0)

agent=sorted(x2['Agent'].unique())

figure1 = plt.Figure(figsize=(8,5), dpi=70)

ax1 = figure1.add\_subplot(111)

bar1 = FigureCanvasTkAgg(figure1, insight\_5\_screen)

bar1.draw()

bar1.get\_tk\_widget().place(relx=.5,rely=.5,anchor='center')

df1 = pd.DataFrame({'Owned Area': x3,'Leased Area':y3}, index=agent)

df1.plot(kind='bar',ax=ax1,title='Performance of Agents in Year '+yearselect.get())#,legend=False)

toolbarFrame = Frame(master=insight\_5\_screen)

toolbarFrame.place(relx=.25,rely=.82)

toolbar = NavigationToolbar2Tk(bar1, toolbarFrame)

toolbar.update()

def insight\_6():

global insight\_6\_screen

insight\_6\_screen = Toplevel(screen)

insight\_6\_screen.title("Total area sold in July")

adjustWindow(insight\_6\_screen) # configuring the window

back = PhotoImage(file="back.png")

image1= PhotoImage(file="bg\_1.png")

label\_for\_image= Label(insight\_6\_screen, image=image1)

label\_for\_image.place(x=0, y=0, relwidth=1, relheight=1)

Label(insight\_6\_screen, text="Total area sold in July ", width='500', height="1", font=("Ariel", 20,), fg='white', bg='#43315E', anchor =W).pack()

x = df[(df['Month']=='JUL') & (df['Tenure']=='Owned') & (df['UoM']=='SQ-M')]

y = df[(df['Month']=='JUL') & (df['Tenure']=='Owned') & (df['UoM']=='HA')]

sumsqm=list(x.groupby(['Year'])['Area'].sum())

sumha=list(y.groupby(['Year'])['Area'].sum())

years = sorted(x['Year'].unique())

for i in range(len(sumha)):

sumsqm[i] = sumsqm[i]+ (sumha[i]\*10000)

figure1 = plt.Figure(figsize=(7,5), dpi=85)

ax1 = figure1.add\_subplot(111)

bar1 = FigureCanvasTkAgg(figure1, insight\_6\_screen)

bar1.draw()

bar1.get\_tk\_widget().place(relx=.5,rely=.5,anchor='center')

df1 = pd.DataFrame({'Total Area (represented in SQ-M)': sumsqm}, index=years)

df1.plot(kind='bar', ax=ax1, title='Total Area Sold in the month of July across the Years (represented in SQ-M)')#,autopct='%1.2f%%',startangle=0)

toolbarFrame = Frame(master=insight\_6\_screen)

toolbarFrame.place(relx=.25,rely=.86)

toolbar = NavigationToolbar2Tk(bar1, toolbarFrame)

toolbar.update()

Button(insight\_6\_screen, image = back, border = 0, command=insight\_6\_screen.destroy).place(relx=.5, rely=.95, anchor="center")

insight\_6\_screen.mainloop()

def insight\_7():

global insight\_7\_screen

insight\_7\_screen = Toplevel(screen)

insight\_7\_screen.title("Time Series Analysis ")

adjustWindow(insight\_7\_screen) # configuring the window

back = PhotoImage(file="back.png")

image1= PhotoImage(file="bg\_1.png")

label\_for\_image= Label(insight\_7\_screen, image=image1)

label\_for\_image.place(x=0, y=0, relwidth=1, relheight=1)

Label(insight\_7\_screen, text="Time Series Analysis ", width='500', height="1", font=("Ariel", 20,), fg='white', bg='#43315E', anchor =W).pack()

connection = pymysql.connect(host="localhost", user="root", passwd="", database="sales") # database connection

cursor = connection.cursor()

select\_query = "SELECT ORD\_AMOUNT,ORD\_DATE FROM orders ORDER BY orders.ORD\_DATE ASC ;" # queries for retrieving values

cursor.execute(select\_query) # executing the queries

info = cursor.fetchall()

amt,dt = [],[]

for i in info:

amt.append(float(i[0]))

dt.append(i[1])

connection.commit() # commiting the connection then closing it.

connection.close()

figure1 = plt.Figure(figsize=(12,5), dpi=60)

ax1 = figure1.add\_subplot(111)

bar1 = FigureCanvasTkAgg(figure1, insight\_7\_screen)

bar1.draw()

bar1.get\_tk\_widget().place(relx=.5,rely=.5,anchor='center')

df7 = pd.DataFrame({'Order Amount ': amt}, index=dt)

df7.plot(ax=ax1 , title='Time series analysis of all orders in year ' + str(info[0][1].year)+'-'+ str(info[-1][1].year))#,autopct='%1.2f%%',startangle=0,subplots=True)

toolbarFrame = Frame(master=insight\_7\_screen)

toolbarFrame.place(relx=.25,rely=.755)

toolbar = NavigationToolbar2Tk(bar1, toolbarFrame)

toolbar.update()

Button(insight\_7\_screen, image = back, border = 0, command=insight\_7\_screen.destroy).place(relx=.5, rely=.95, anchor="center")

insight\_7\_screen.mainloop()

def main\_screen():

global screen, username\_verify, password\_verify

screen = tk.ThemedTk()

screen.get\_themes()

screen.set\_theme("arc")# initializing the tkinter window

username\_verify, password\_verify = StringVar(),StringVar()

screen.title("Login") # mentioning title of the window

adjustWindow(screen) # configuring the window

image1, login\_button, register\_button = PhotoImage(file="login\_final\_1.png"),PhotoImage(file="login\_button\_3.png"),PhotoImage(file="register\_button\_3.png")

label\_for\_image= Label(screen, image=image1)

label\_for\_image.place(x=0, y=0, relwidth=1, relheight=1)

Label(screen, text="E-mail ID ", font=("Roboto lt", 8 , 'italic'), bg='black', fg='white').place(relx=.337, rely=.35, anchor="center")

Label(screen, text="Password ", font=("Roboto lt", 8, 'italic'), bg='black', fg='white').place(relx=.3425, rely=.45, anchor="center")

ttk.Entry(screen, textvar=username\_verify , width='39').place(relx=.5, rely=.39, anchor="center")

ttk.Entry(screen, textvar=password\_verify, show="\*" , width='39').place(relx=.5, rely=.49, anchor="center")

Button(screen, image = login\_button , border = 0, command=login\_verify).place(relx=.5, rely=.63, anchor="center")

Button(screen, image = register\_button , border = 0, command=register).place(relx=.5, rely=.73, anchor="center")

screen.mainloop()

main\_screen()

CONCLUSION

The project entitled “ Sunville Properties App “ is developed using Python Tkinter as front end MySQL database in the back end to computerize and ease the process of Sunville Properties ,which can help them seamlessly navigate via different forms of storage and which can also help them to fetch , modify and analyze their data .