Advanced Database System Lab

Assignment no. 9

PRN: 2020BTECS00005

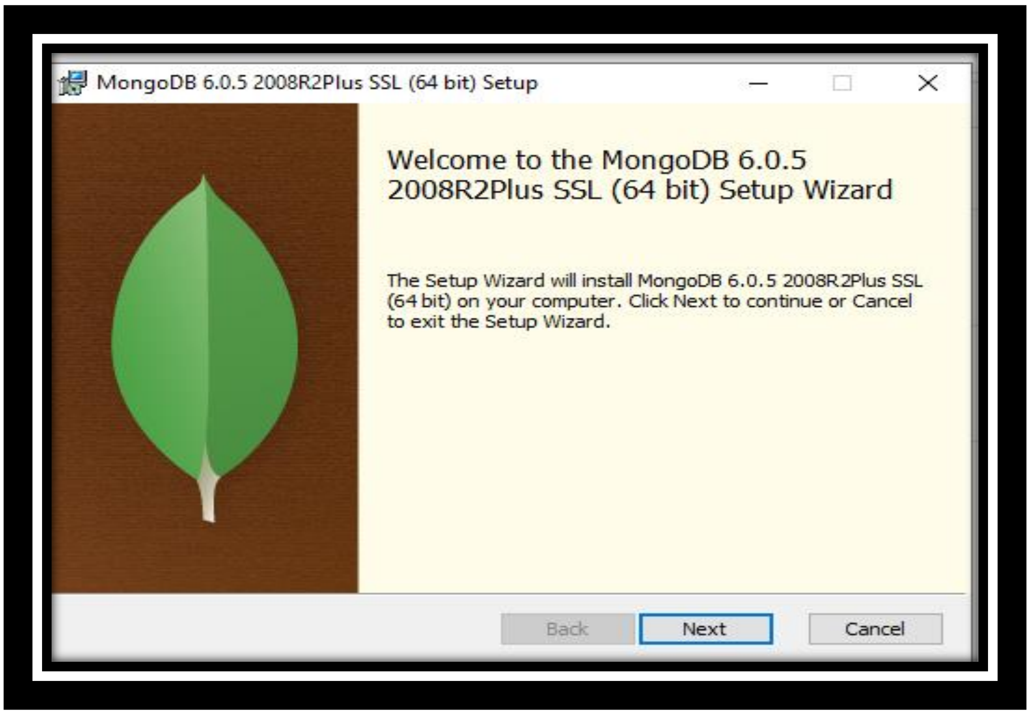
Name: Sanket Shivaji Jadhav

- **Title:** Install & deploy Cloud Databases on Windows.
- **Aim:** Install & deploy MongoDB & CassandraDB on windows. Create a python GUI for CRUD operations.
- **Installation:**

1. MongoDB:

Step 1. Download MongoDB Community Server here.

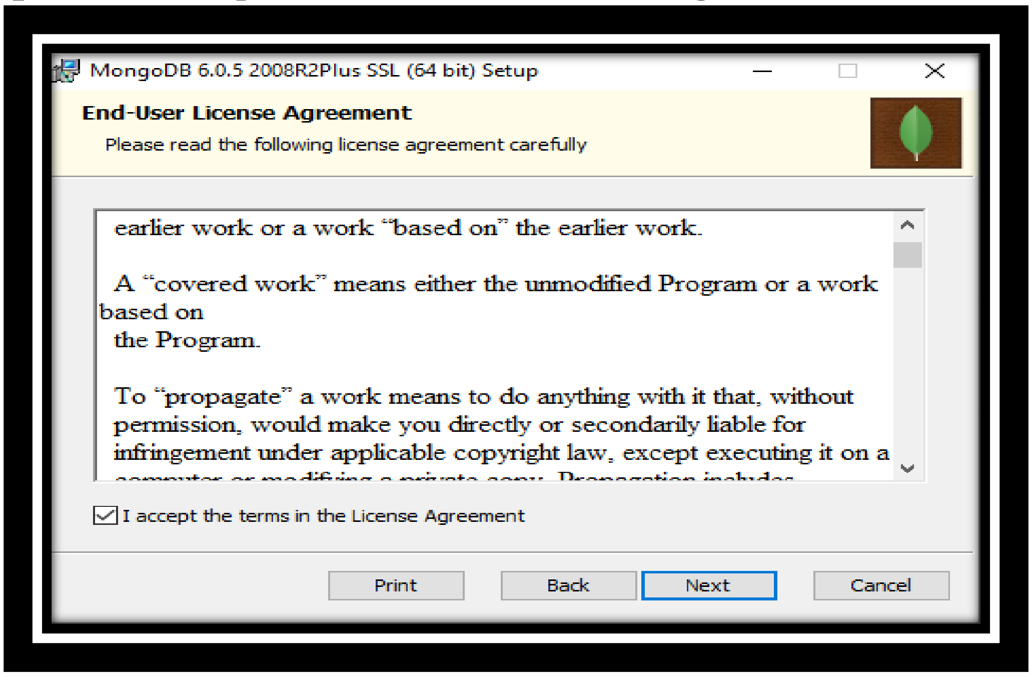
Step 2. Once download is complete open the msi file. Click Next in the start up screen.



Welcome to the MongoDB 6.0.5 2008R2Plus SSL (64 bit) Setup Wizard

The Setup Wizard will install MongoDB 6.0.5 2008R2Plus SSL (64 bit) on your computer. Click Next to continue or Cancel to exit the Setup Wizard.

Step 3. Accept the End-User License Agreement.



MongoDB 6.0.5 2008R2Plus SSL (64 bit) Setup

End-User License Agreement

Please read the following license agreement carefully

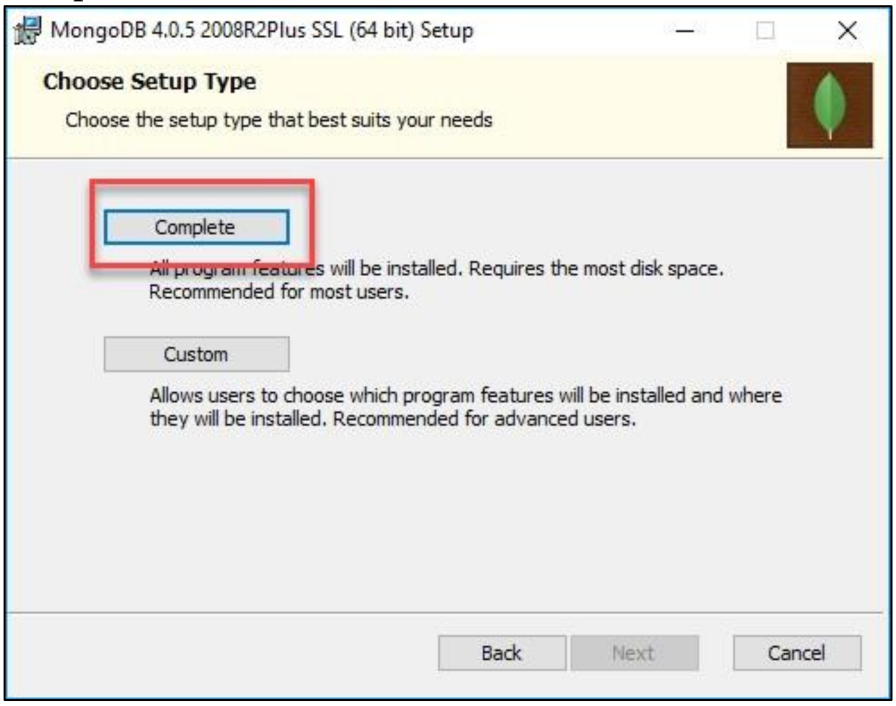
earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on
the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a

☑ I accept the terms in the License Agreement

Step 4. Click on the "complete" button to install all of the components.



MongoDB 4.0.5 2008R2Plus SSL (64 bit) Setup

Choose Setup Type

Choose the setup type that best suits your needs

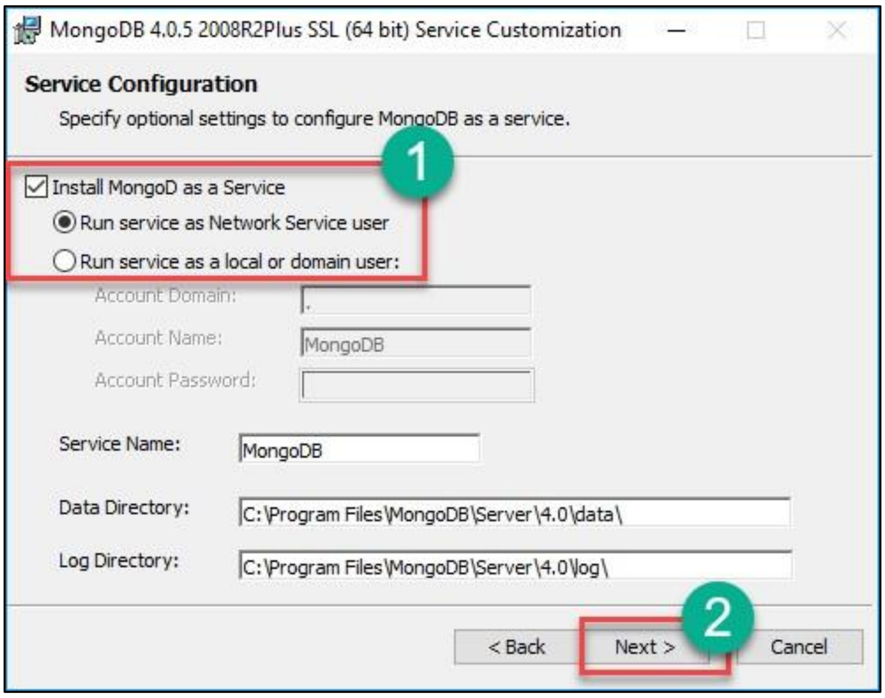
Complete

All program features will be installed. Requires the most disk space. Recommended for most users.

Custom

Allows users to choose which program features will be installed and where they will be installed. Recommended for advanced users.

Step 5. Select "Run service as Network Service user". make a note of the data directory, we'll need this later. Click Next.


MongoDB 4.0.5 2008R2Plus SSL (64 bit) Service Customization

Service Configuration

Specify optional settings to configure MongoDB as a service.

[✓] Install MongoD as a Service

(●) Run service as Network Service user

( ) Run service as a local or domain user:

Account Domain: .

Account Name: MongoDB

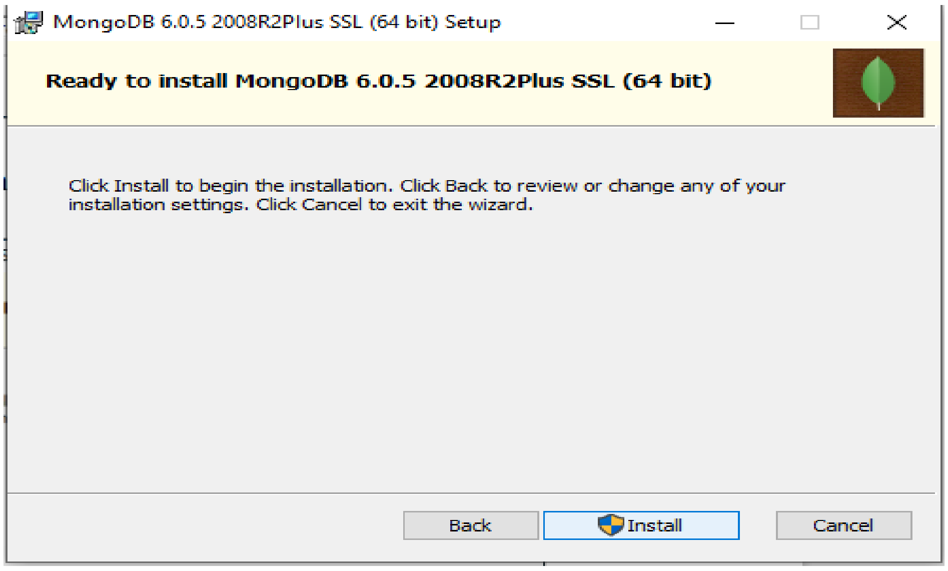
Account Password:

Service Name: MongoDB

Data Directory: C:\Program Files\MongoDB\Server\4.0\data\

Log Directory: C:\Program Files\MongoDB\Server\4.0\log\

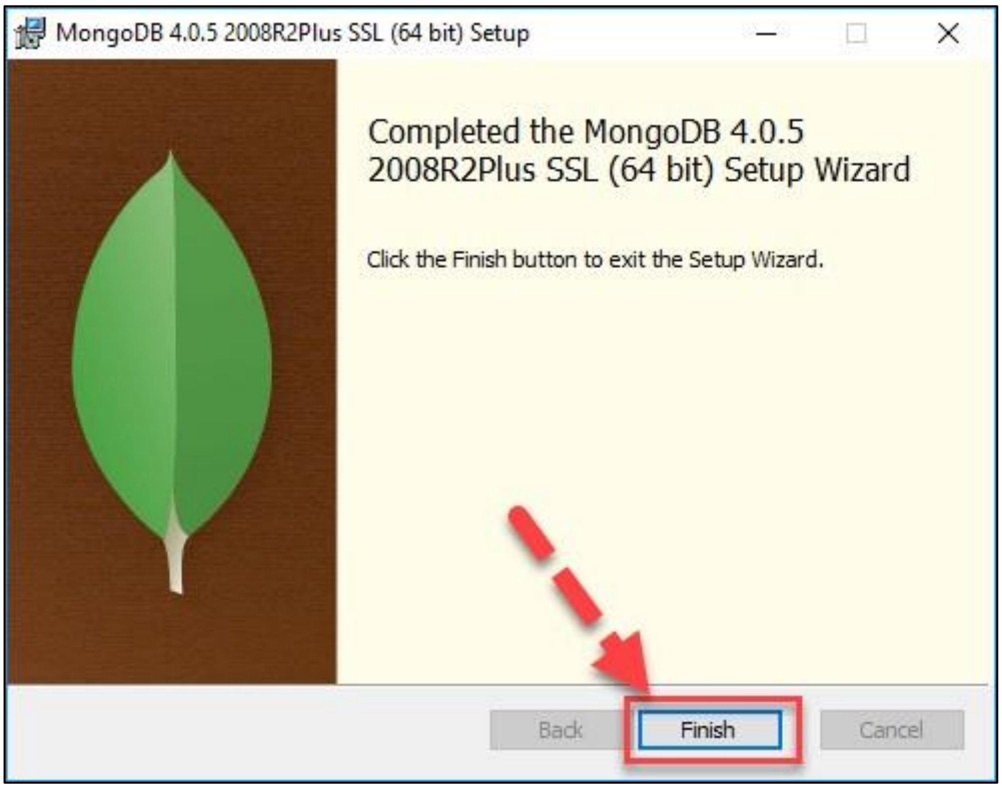
Step 6. Click on the Install button to start the installation.


MongoDB 6.0.5 2008R2Plus SSL (64 bit) Setup

Ready to install MongoDB 6.0.5 2008R2Plus SSL (64 bit)

Click Install to begin the installation. Click Back to review or change any of your installation settings. Click Cancel to exit the wizard.

Step 7. Once complete the installation, Click on the Finish button

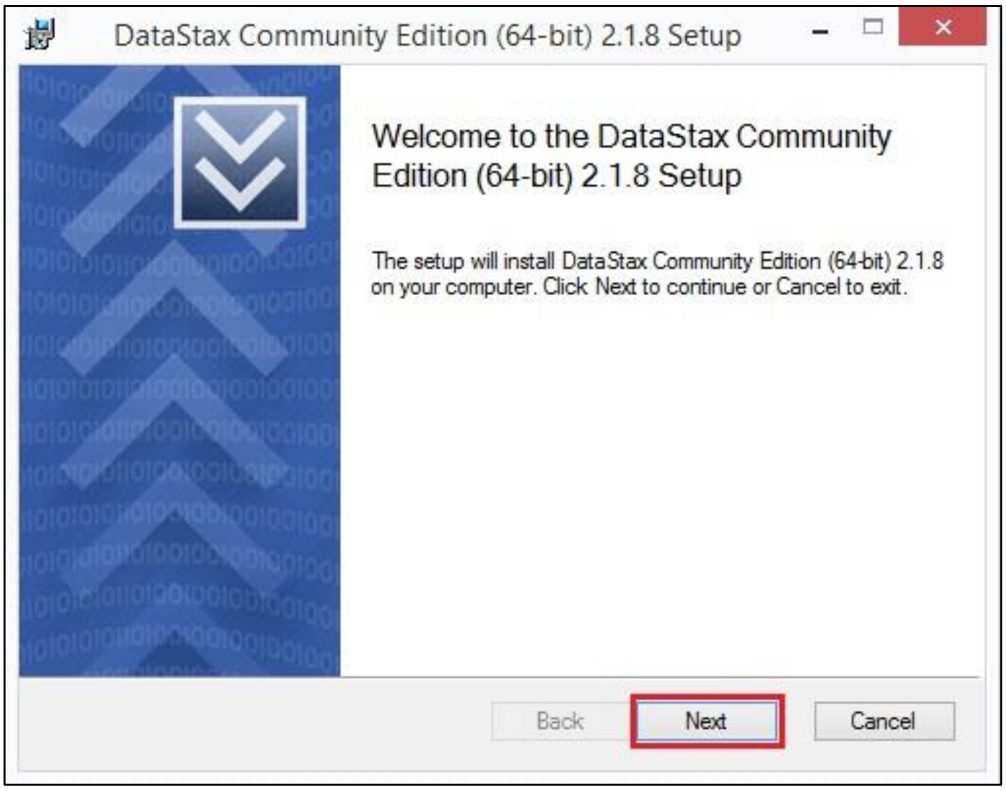

MongoDB 4.0.5 2008R2Plus SSL (64 bit) Setup

Completed the MongoDB 4.0.5 2008R2Plus SSL (64 bit) Setup Wizard

Click the Finish button to exit the Setup Wizard.

2. CassandraDB:

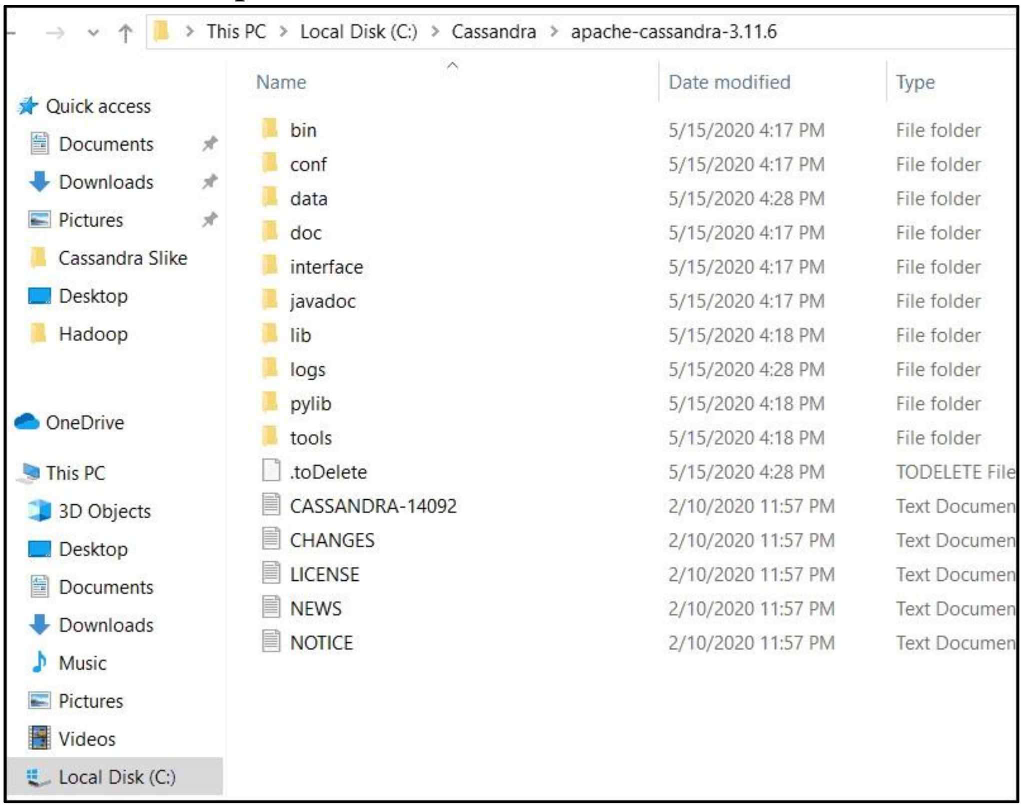
Step 1. Download setup from Apache Downloads & extract it.


DataStax Community Edition (64-bit) 2.1.8 Setup

Welcome to the DataStax Community Edition (64-bit) 2.1.8 Setup

The setup will install DataStax Community Edition (64-bit) 2.1.8 on your computer. Click Next to continue or Cancel to exit.

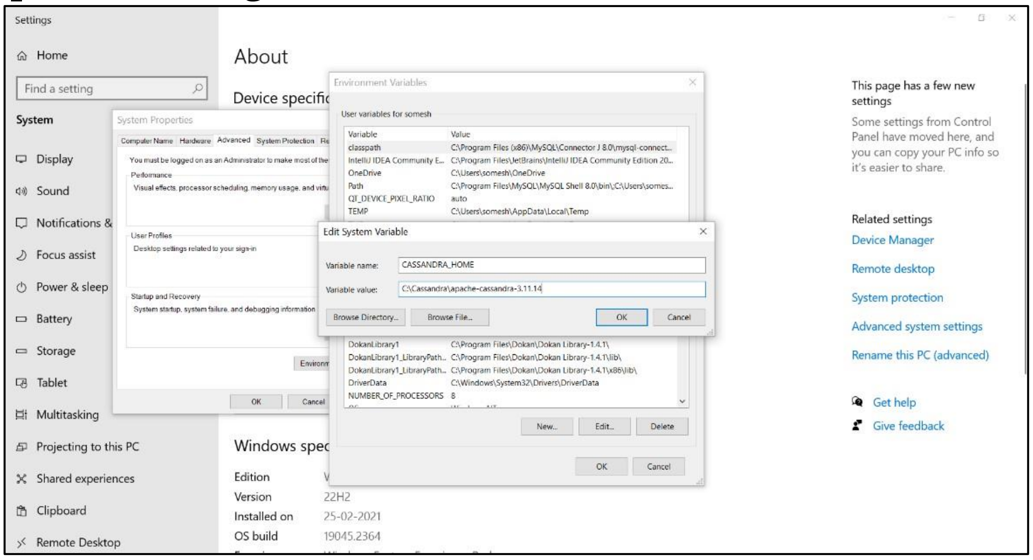
Step 2. Unzip the folder, and place the content in the C:Cassandraapache-cassandra-3.11.6 folder.



This PC > Local Disk (C:) > Cassandra > apache-cassandra-3.11.6

Name	Date modified	Type
bin	5/15/2020 4:17 PM	File folder
conf	5/15/2020 4:17 PM	File folder
data	5/15/2020 4:28 PM	File folder
doc	5/15/2020 4:17 PM	File folder
interface	5/15/2020 4:17 PM	File folder
javadoc	5/15/2020 4:17 PM	File folder
lib	5/15/2020 4:18 PM	File folder
logs	5/15/2020 4:28 PM	File folder
pylib	5/15/2020 4:18 PM	File folder
tools	5/15/2020 4:18 PM	File folder
.toDelete	5/15/2020 4:28 PM	TODELETE File
CASSANDRA-14092	2/10/2020 11:57 PM	Text Documen
CHANGES	2/10/2020 11:57 PM	Text Documen
LICENSE	2/10/2020 11:57 PM	Text Documen
NEWS	2/10/2020 11:57 PM	Text Documen
NOTICE	2/10/2020 11:57 PM	Text Documen

Step 3. Configure the environment variables.



Environment Variables

User variables for somesh

Variable	Value
classpath	C:\Program Files (x86)\MySQL\Connector J 8.0\mysql-connect...
IntelliJ IDEA Community E...	C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 20...
OneDrive	C:\Users\somesh\OneDrive
Path	C:\Program Files\MySQL\MySQL Shell 8.0\bin\;C:\Users\somes...
QT_DEVICE_PIXEL_RATIO	auto
TEMP	C:\Users\somesh\AppData\Local\Temp

Edit System Variable

Variable name: CASSANDRA_HOME

Variable value: C:\Cassandra\apache-cassandra-3.11.14

Variable	Value
DokanLibrary1	C:\Program Files\Dokan\Dokan Library-1.4.1\
DokanLibrary1_LibraryPath...	C:\Program Files\Dokan\Dokan Library-1.4.1\lib\
DokanLibrary1_LibraryPath...	C:\Program Files\Dokan\Dokan Library-1.4.1\x86\lib\
DriverData	C:\Windows\System32\Drivers\DriverData
NUMBER_OF_PROCESSORS	8

Edition
Version 22H2
Installed on 25-02-2021
OS build 19045.2364

This page has a few new settings

Some settings from Control Panel have moved here, and you can copy your PC info so it's easier to share.

Step 4. Start Cassandra from cmd.

Python GUI Application:

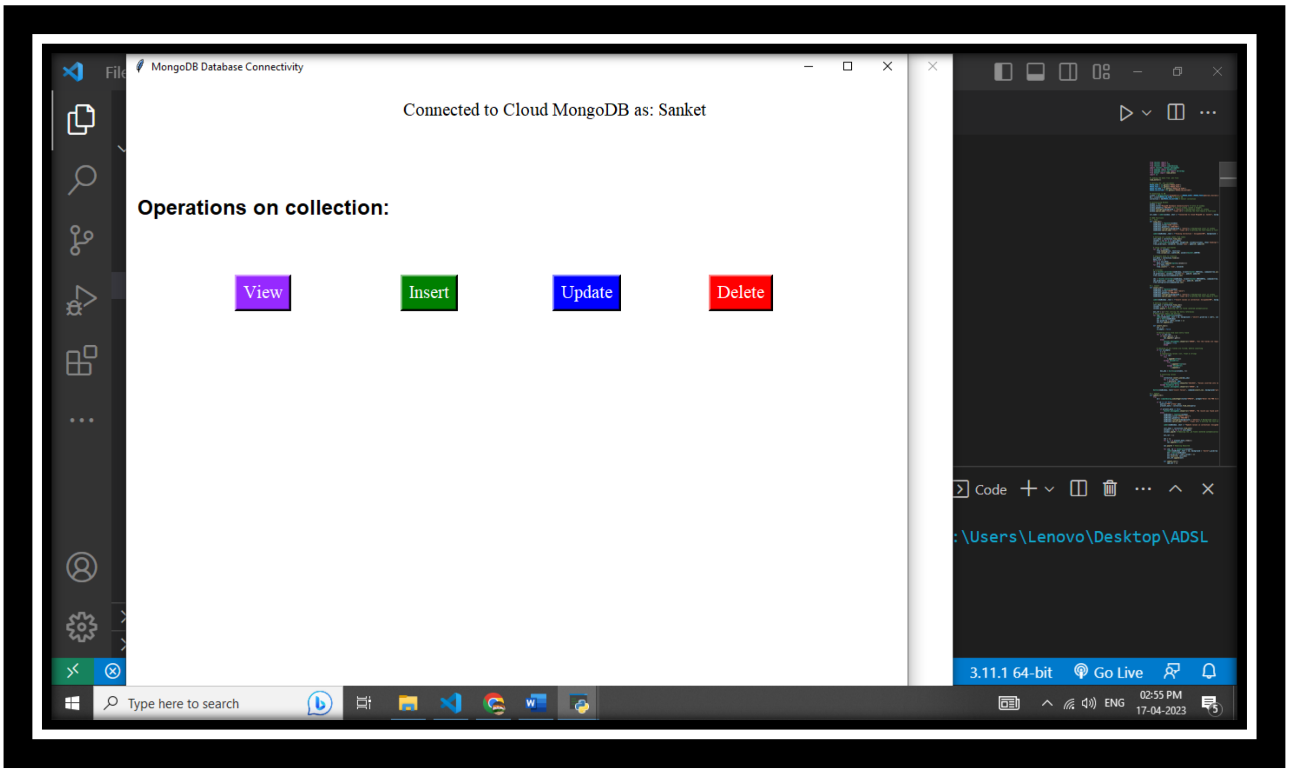
```
from tkinter import *
from tkinter import ttk
from tkinter import simpledialog
import tkinter, tkinter.messagebox
from pymongo import MongoClient
from pymongo.server_api import ServerApi
from dotenv import load_dotenv
import os

# Loading the data from .env file
load_dotenv()

# Getting the .env variables
MONGO_USER = os.getenv("MONGO_USER")
MONGO_PASS = os.getenv("MONGO_PASS")
MONGO_DB_NAME = os.getenv("MONGO_DB_NAME")
MONGO_COLLECTION = os.getenv("MONGO_COLLECTION")

# Connecting to DB
client =
MongoClient(f"mongodb+srv://{MONGO_USER}:{MONGO_PASS}@adslab.jtazlad.mongodb
.net/test", server_api=ServerApi('1'))
db = client[MONGO_DB_NAME] # Select DB
collection = db[MONGO_COLLECTION] # Select collection

# Initializing Window
window = Tk()
window.title("MongoDB Database Connectivity") # Title of window
window.geometry('900x900') # Size of window (width X height)
window.configure(background = "white"); # Background color of window
window.option_add("*Font", "Times 16") # Setting the font-family & font-size

usr_name = Label(window ,text = f"Connected to Cloud MongoDB as: Sanket",
background = "white").grid(row = 0, column = 1, pady=20)

# CRUD Functions
# 1. View
def view_tb():
    newWindow = Toplevel(window)
    newWindow.title("VIEW Table")
    newWindow.geometry('1500x900')
    newWindow.configure(background = "white"); # Background color of window
    newWindow.option_add("*Font", "Times 16") # Setting the font-family &
font-size
```

```
    Label(newWindow ,text = f"Viewing Collection - Assignment09", background
= "white").grid(row = 0, column = 0, padx=10, pady=10)

    # Getting all column names from table
    coll_keys = collection.find_one()
    columns = [a for a in coll_keys]
    tree = ttk.Treeview(newWindow, height=20, columns=columns,
show='headings')
    tree.grid(row=1, column=0, sticky='news', padx=10, pady=10)

    # setup columns attributes
    for col in columns:
        tree.heading(col, text=col)
        tree.column(col, width=100, anchor=tkinter.CENTER)

    # populate data to treeview
    all_data = collection.find({})
    data_list = []
    for a in all_data:
        data_list.append(tuple(a.values()))
    for d in data_list:
        tree.insert('', 'end', value=d)

    # scrollbar
    sb = tkinter.Scrollbar(newWindow, orient=tkinter.VERTICAL,
command=tree.yview)
    sb.grid(row=1, column=1, sticky='ns', padx=0, pady=10)
    tree.config(yscrollcommand=sb.set)

    sbx = tkinter.Scrollbar(newWindow, orient=tkinter.HORIZONTAL,
command=tree.xview)
    sbx.grid(row=2, column=0, sticky='ew', padx=10, pady=0)
    tree.config(xscrollcommand=sbx.set)

# 2. Insert
def insert_tb():
    newWindow = Toplevel(window)
    newWindow.title("INSERT into Table")
    newWindow.geometry('900x900')
    newWindow.configure(background = "white"); # Background color of window
    newWindow.option_add("*Font", "Times 16") # Setting the font-family &
font-size

    Label(newWindow ,text = f"Insert values in collection: Assignment09",
background = "white").grid(row = 0, column = 0, padx=10, pady=10)

    # Getting columns names
    coll_keys = collection.find_one()
    columns = [a for a in coll_keys]
    columns.pop(0) # Removing the _id field (entered automatically)

    ent_ref = [] # For storing the Entry references
    # Populating Labels and Entries
    for ind, nm in enumerate(columns):
        Label(newWindow ,text = nm, background = "white").grid(row = ind+1,
column = 0, padx=10, pady=10)
        ent = Entry(newWindow)
        ent.grid(row = ind+1,column = 1)
        ent_ref.append(ent)

    def insert_val():
        val = []
        is_empty = False

        # Getting value from each entry field
        for r in ent_ref:
            if len(r.get()) > 0:
                val.append(r.get())
            else:
                tkinter.messagebox.showerror("ERROR", "All the fields are
required!")
                is_empty = True
                break

        # Checking if all fields are filled, before inserting
        if not is_empty:
            v = []
            # Typecasting values (int, float & string)
            for x in val:
                try:
                    v.append(int(x))
                except ValueError:
                    try:
                        v.append(float(x))
                    except ValueError:
                        v.append(x)

            doc_obj = dict(zip(columns, v))

            # Inserting values
            try:
                collection.insert_one(doc_obj)
                for r in ent_ref:
                    r.delete(0, END)
                tkinter.messagebox.showinfo("SUCCESS", "Values inserted into
Collection successfully!")
```

```
            except Exception as e:
                tkinter.messagebox.showerror("ERROR", e)

    Button(newWindow, text="Insert Values", command=insert_val,
background="green", foreground="white").grid(row = ind+2, column = 1,
pady=20, sticky='ew')

# 3. Update
def update_tb():
    try:
        id = simpledialog.askinteger(title="UPDATE", prompt="Enter the PRN
to be updated: ")

        if id is not None:
            query={"PRN":{"$eq":id}}
            present_data = collection.find_one(query)

            if present_data is None:
                tkinter.messagebox.showerror("ERROR", "No record was found
with the given PRN !")
            else:
                newWindow = Toplevel(window)
                newWindow.title("UPDATE Table")
                newWindow.geometry('900x900')
                newWindow.configure(background = "white"); # Background
color of window
                newWindow.option_add("*Font", "Times 16") # Setting the
font-family & font-size

                Label(newWindow ,text = f"Update values in collection:
Assignment09", background = "white").grid(row = 0, column = 0, padx=10,
pady=10)

                coll_keys = collection.find_one()
                columns = [a for a in coll_keys]
                columns.pop(0) # Removing the _id field (entered
automatically)

                ent_ref = []

                val = []
                for k, v in present_data.items():
                    val.append(str(v))

                val.pop(0) # Removing ObjectId

                for ind, nm in enumerate(columns):
                    Label(newWindow ,text = nm, background =
"white").grid(row = ind+1, column = 0, padx=10, pady=10)
                    ent = Entry(newWindow)
                    ent.grid(row = ind+1,column = 1)
                    ent.insert(0, val[ind])
                    ent_ref.append(ent)

                def update_val():
                    upd_val = []
                    is_empty = False

                    for r in ent_ref:
                        if len(r.get()) > 0:
                            upd_val.append(r.get())
                        else:
                            tkinter.messagebox.showerror("ERROR", "All the
fields are required!")
                            is_empty = True
                            break

                    if not is_empty:
                        v = []
                        for x in upd_val:
                            try:
                                v.append(int(x))
                            except ValueError:
                                try:
                                    v.append(float(x))
                                except ValueError:
                                    v.append(x)

                        try:
                            doc_obj = dict(zip(columns, v))
                            new_data = {"$set":doc_obj}
                            collection.update_one(present_data, new_data)
                            newWindow.destroy()
                            tkinter.messagebox.showinfo("SUCCESS", "Values
updated successfully!")
                        except Exception as e:
                            tkinter.messagebox.showerror("ERROR", e)

                Button(newWindow, text="Update Values", command=update_val,
background="blue", foreground="white").grid(row = ind+2, column = 1,
pady=20, sticky='ew')

    except Exception as e:
        tkinter.messagebox.showerror("ERROR", e)
```

```python
# 4. Delete
def delete_tb():
    try:
        id = simpledialog.askinteger(title="DELETE", prompt="Enter the PRN 
to be deleted: ")

        if id is not None:
            query={"PRN":{"$eq":id}}
            present_data = collection.find_one(query)

            if present_data is None:
                tkinter.messagebox.showerror("ERROR", "Cannot DELETE!\nNo 
record was found with the given PRN !")
            else:
                collection.delete_one(query)
                tkinter.messagebox.showinfo("SUCCESS", "Deleted record from 
Collection successfully!")

    except Exception as e:
        tkinter.messagebox.showerror("ERROR", e)

# CRUD operation buttons
Label(window ,text = "Operations on collection:", background = "white", 
font='Helvetica 18 bold').grid(row = 3, column = 0, padx=10, pady=60)

view_btn = Button(window, text="View", command=view_tb, 
background="#9629ff", foreground="white", border=3).grid(row = 4, column = 
0)
insert_btn = Button(window, text="Insert", command=insert_tb, 
background="green", foreground="white", border=3).grid(row = 4, column = 1, 
sticky='w', columnspan=1)
update_btn = Button(window, text="Update", command=update_tb, 
background="blue", foreground="white", border=3).grid(row = 4, column = 1, 
columnspan=2)
delete_btn = Button(window, text="Delete", command=delete_tb, 
background="red", foreground="white", border=3).grid(row = 4, column = 2)

window.mainloop() # window remains until user closes it
client.close() # Closing the connection to database
```

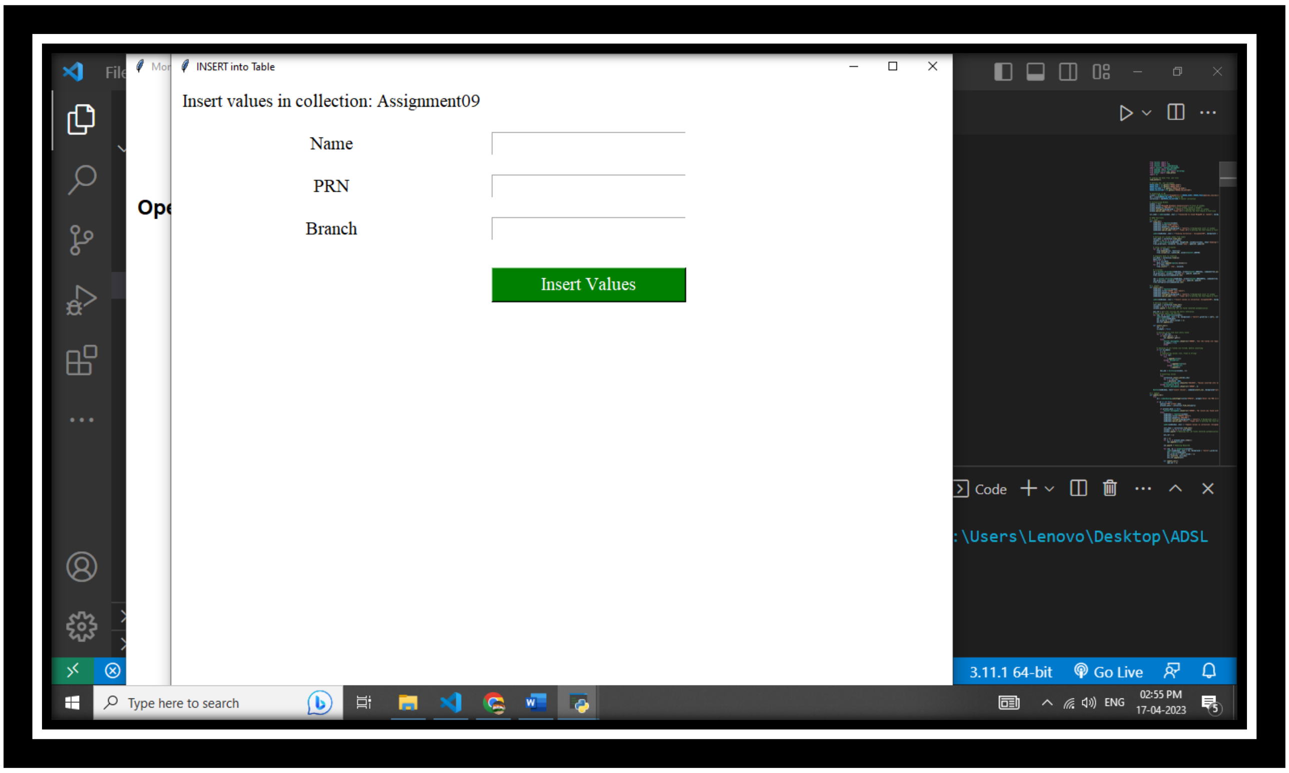
- **Result:**



MongoDB Database Connectivity

Connected to Cloud MongoDB as: Sanket

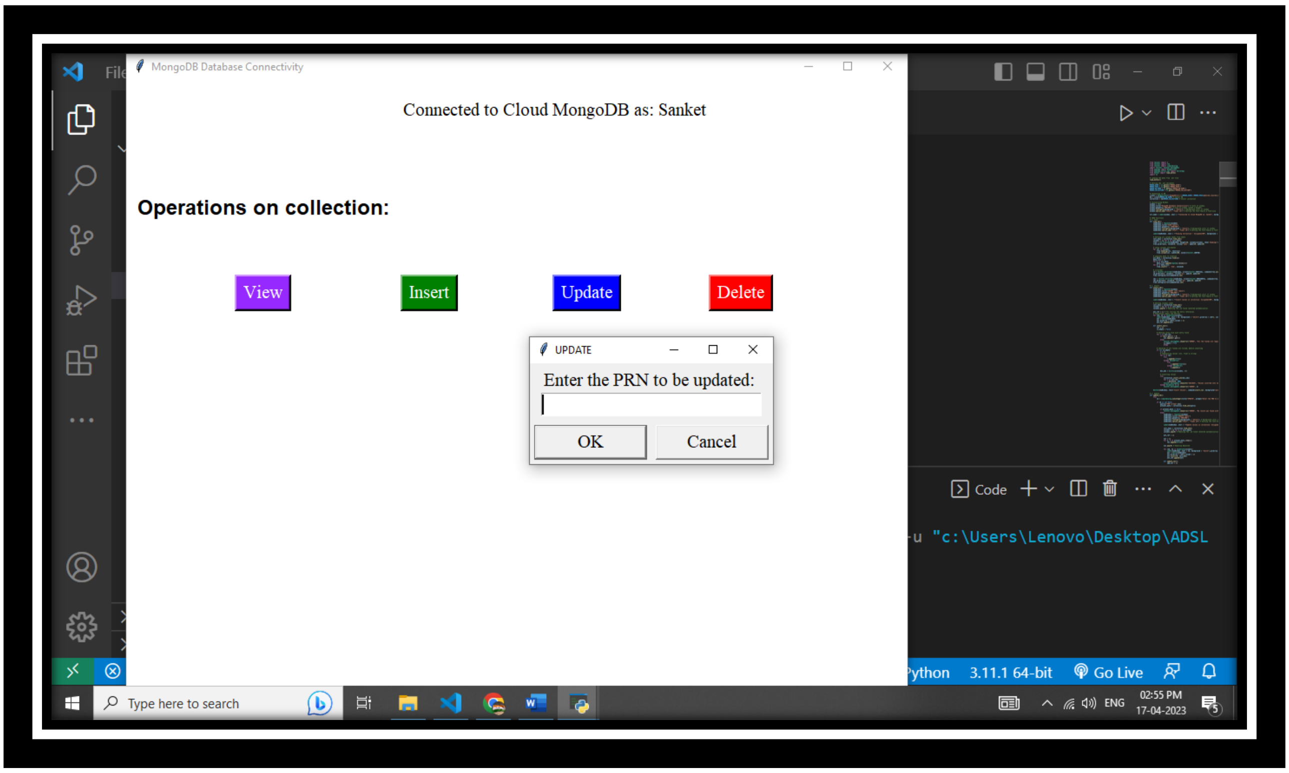
Operations on collection:



INSERT into Table

Insert values in collection: Assignment09

Name	
PRN	
Branch	

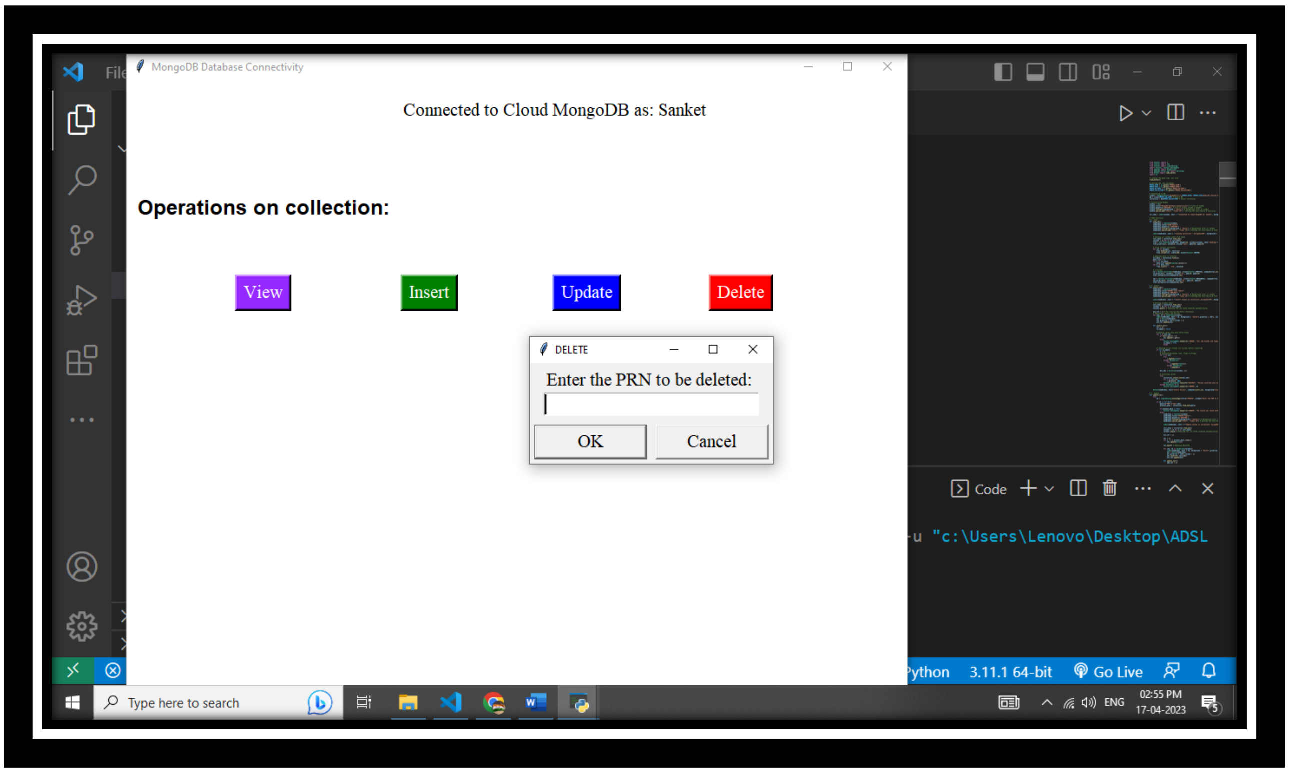


MongoDB Database Connectivity

Connected to Cloud MongoDB as: Sanket

Operations on collection:

```
-u "c:\Users\Lenovo\Desktop\ADSL
```



MongoDB Database Connectivity

Connected to Cloud MongoDB as: Sanket

Operations on collection:

```
-u "c:\Users\Lenovo\Desktop\ADSL
```