TY B.Tech. (CSE) – II [2022-23]

5CS372: Advanced Database System Lab.

Assignment No. 9

Name : Tanaya Mukund Bhide

PRN : 2020BTECS00011

Batch : T5

Branch: T.Y CSE

Install & deploy the following cloud databases on windows platform :

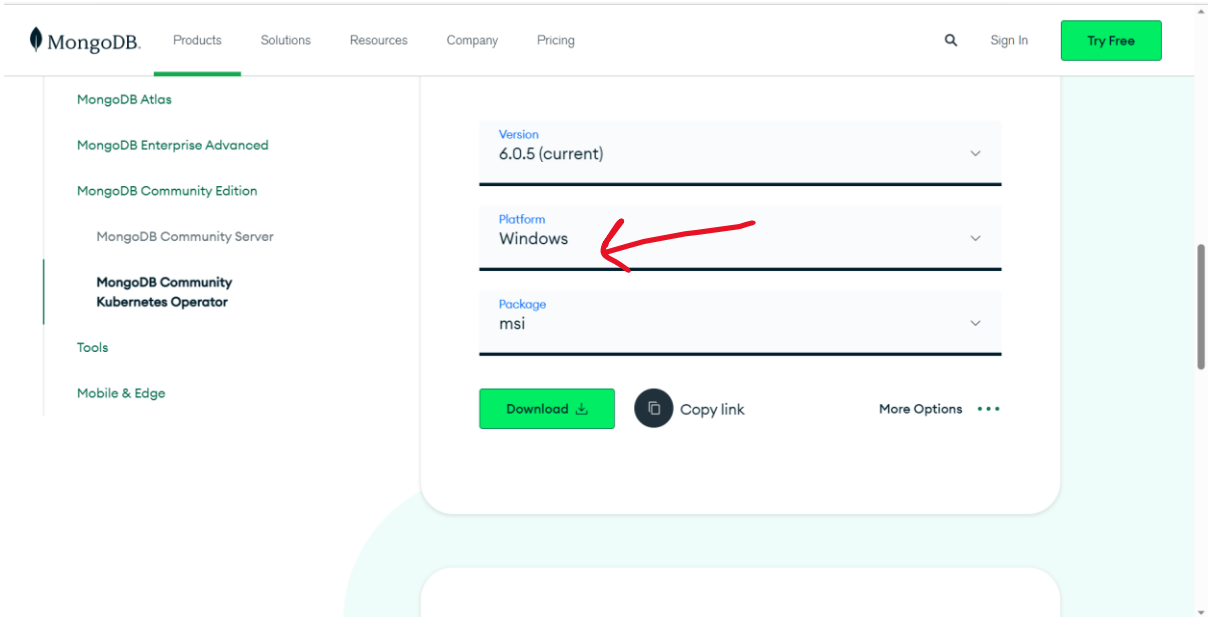
A] MongoDB

B] CassandraDB

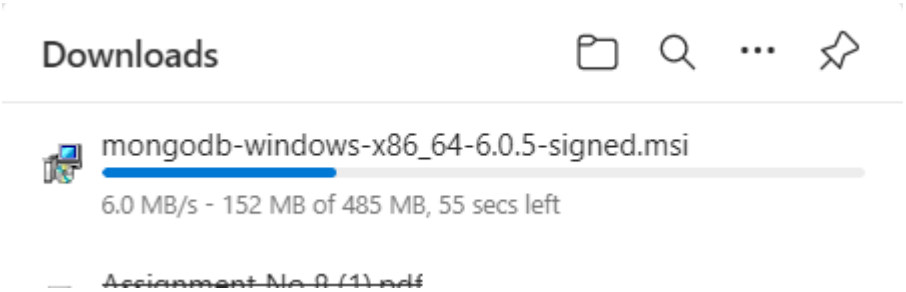
Write Python desktop Application to demonstrate the CRUD operation with above backend cloud databases. Assume any database.

1.MangoDB installation

Go to the MongoDB download page (https://www.mongodb.com/try/download/community) and download the latest version of MongoDB for Windows.

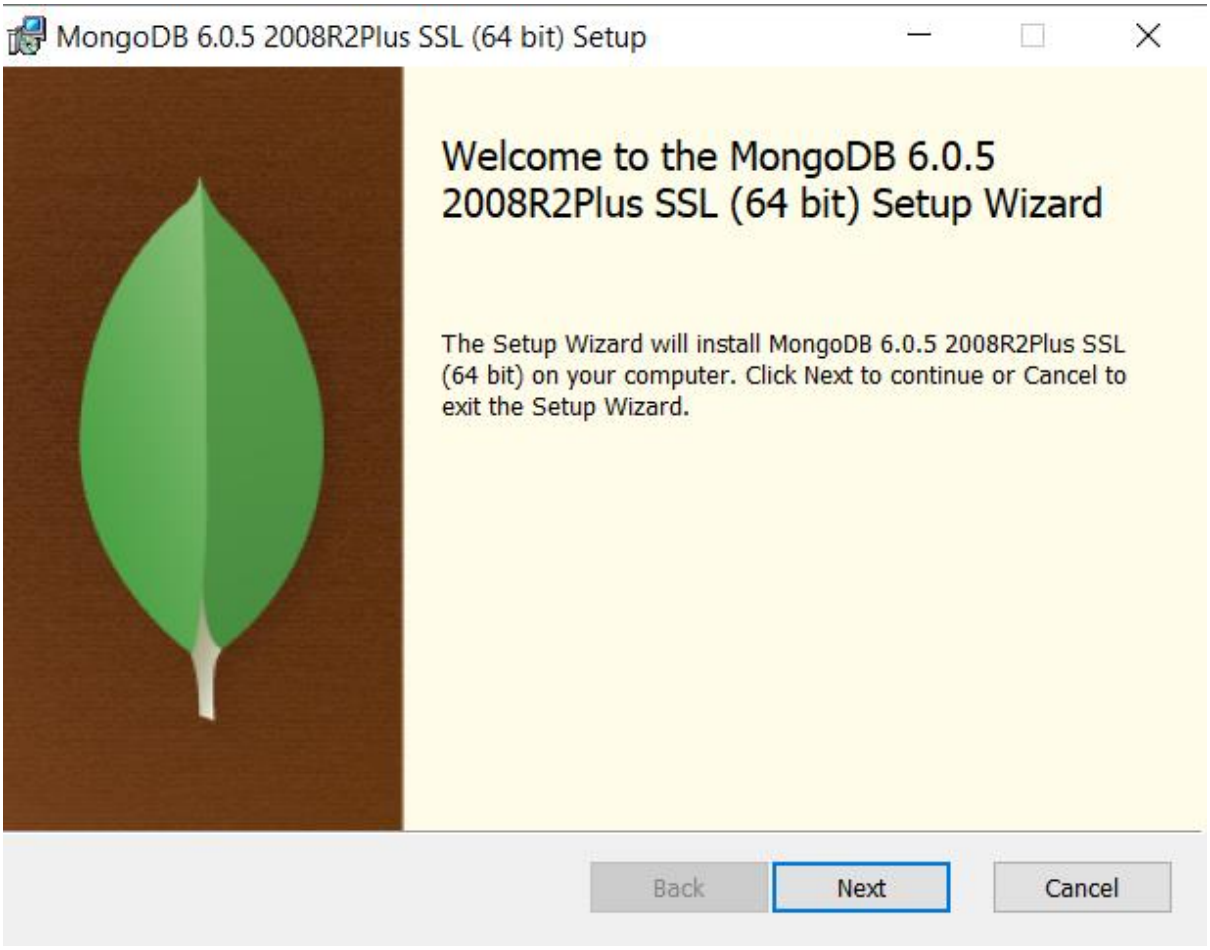


Click the "Download" button to start the download.

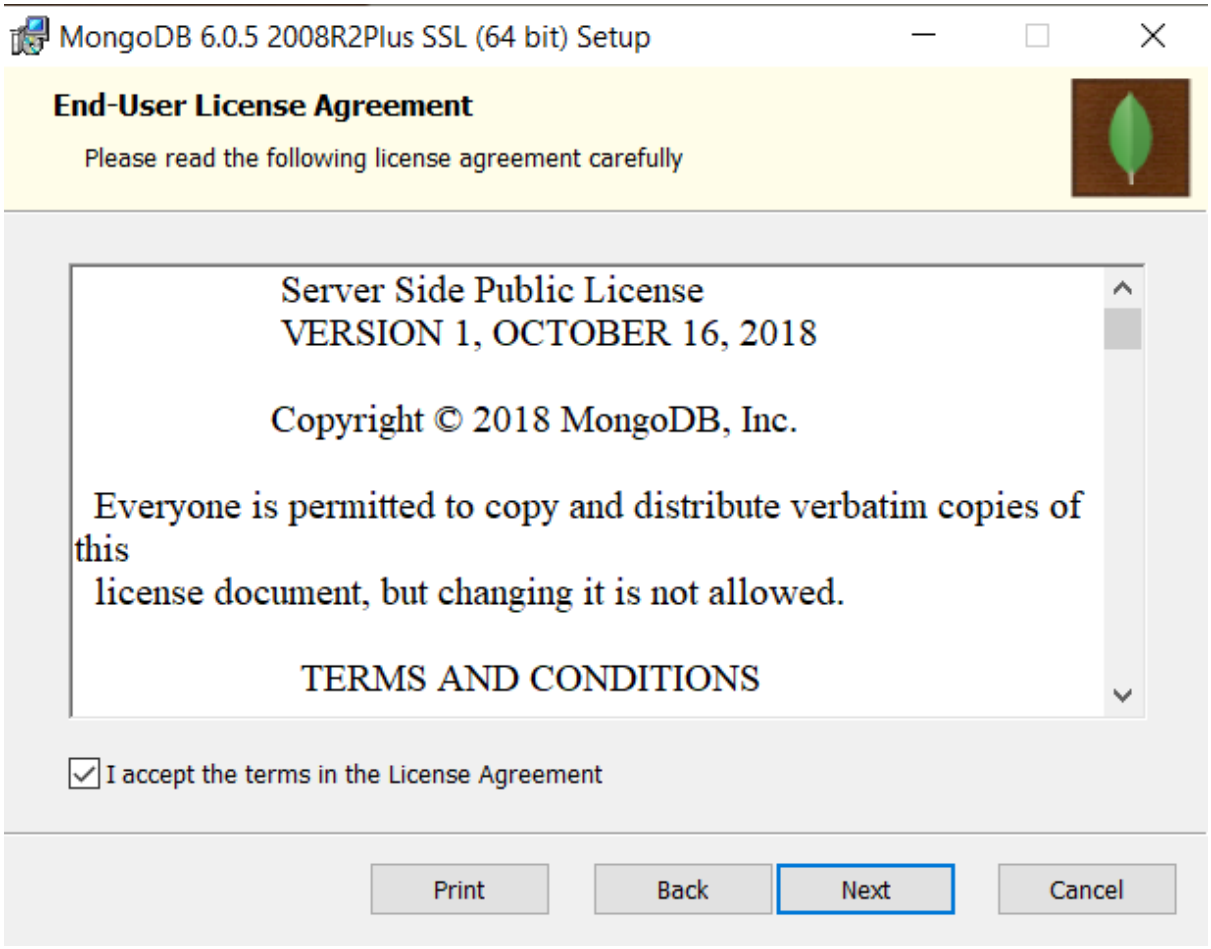


Once the download is complete, run the installer and follow the on-screen instructions to complete the installation process.

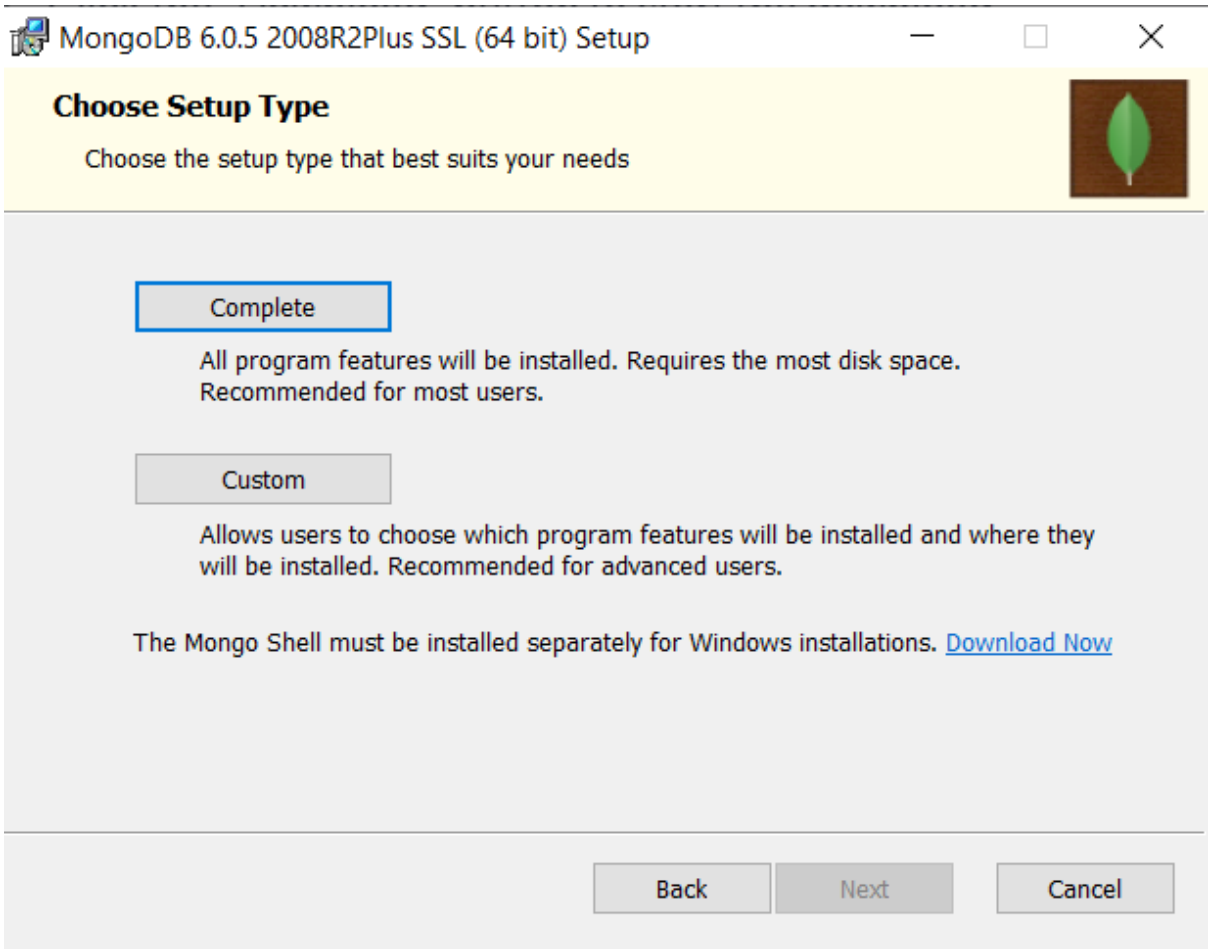
In the "MongoDB Setup Wizard" window, click "Next" to proceed.



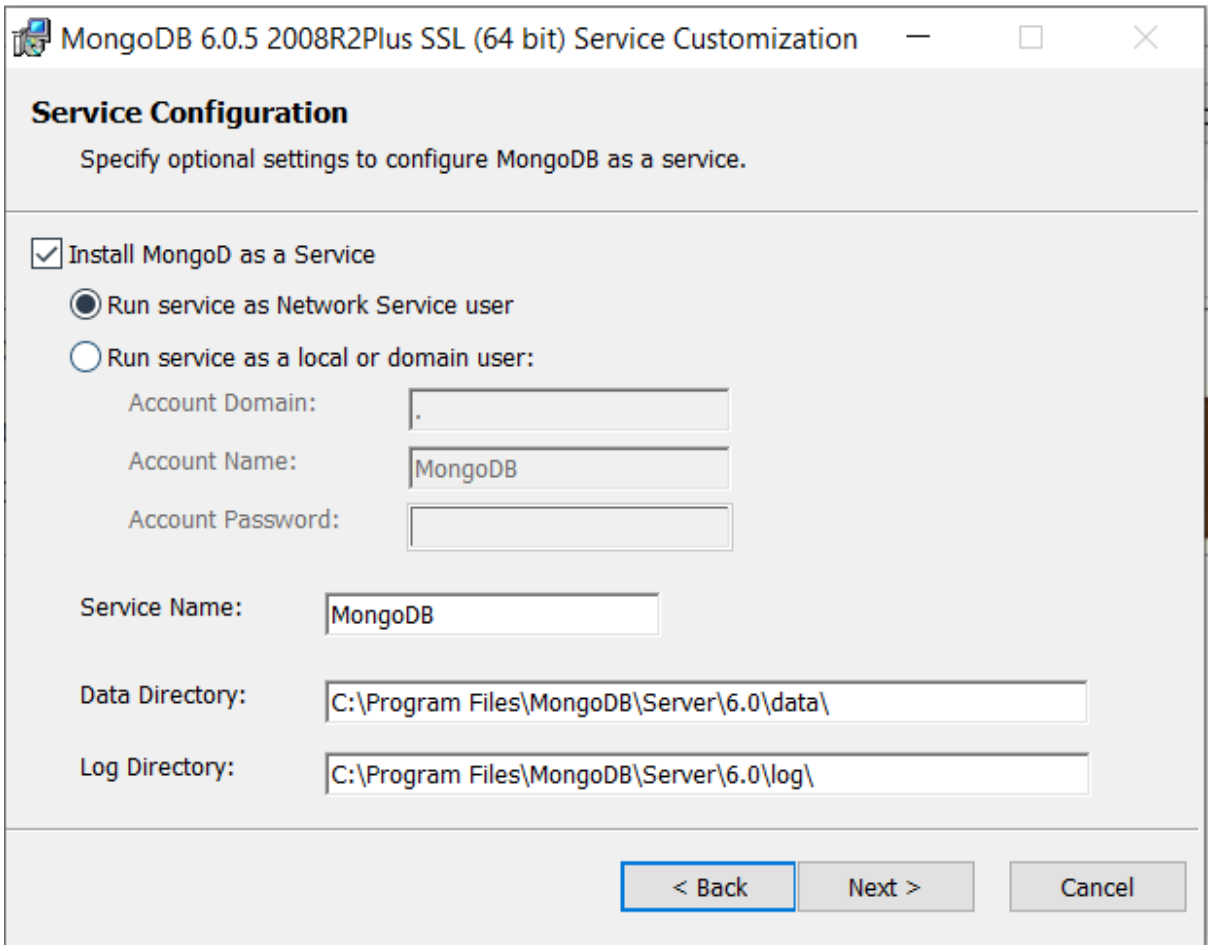
Review and accept the terms of the license agreement, then click "Next".

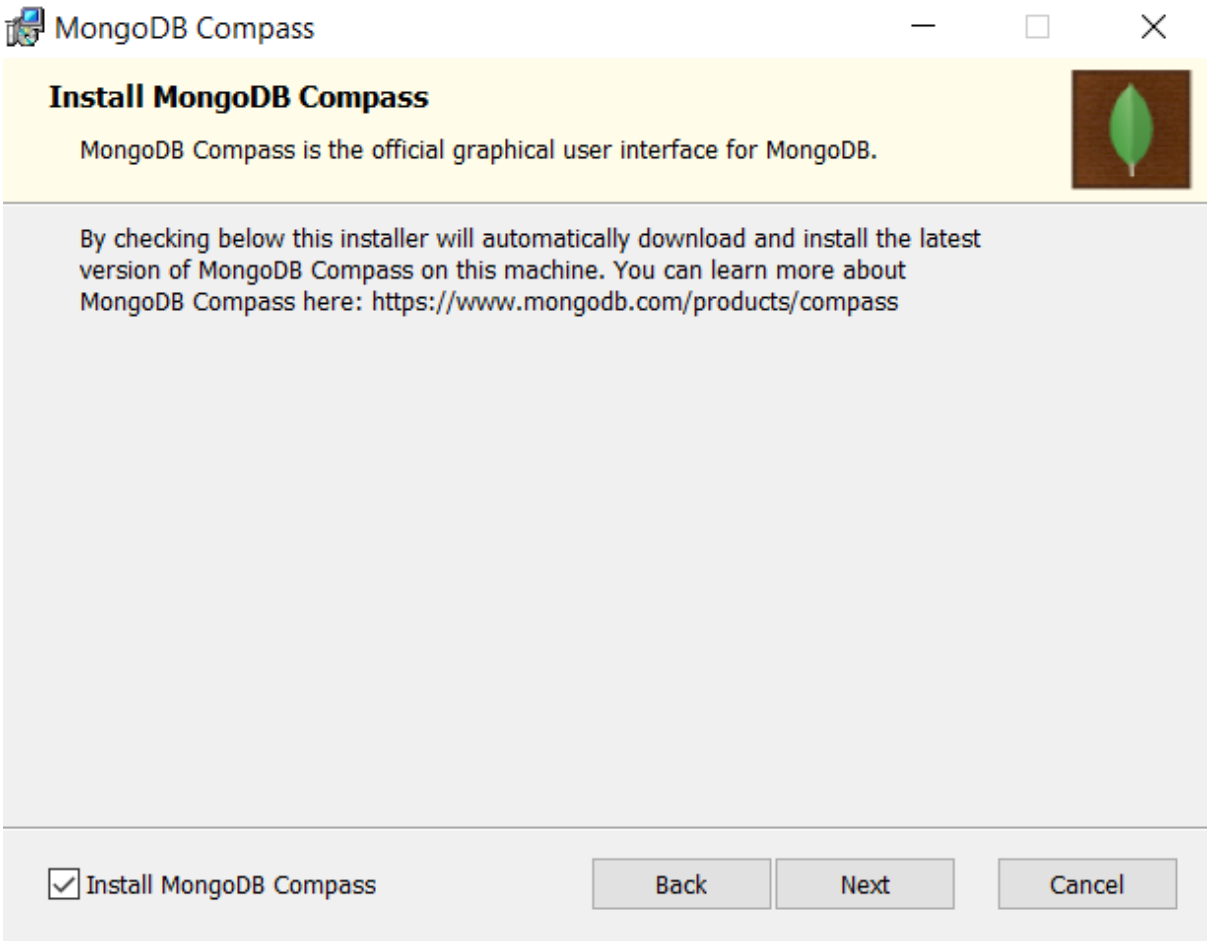


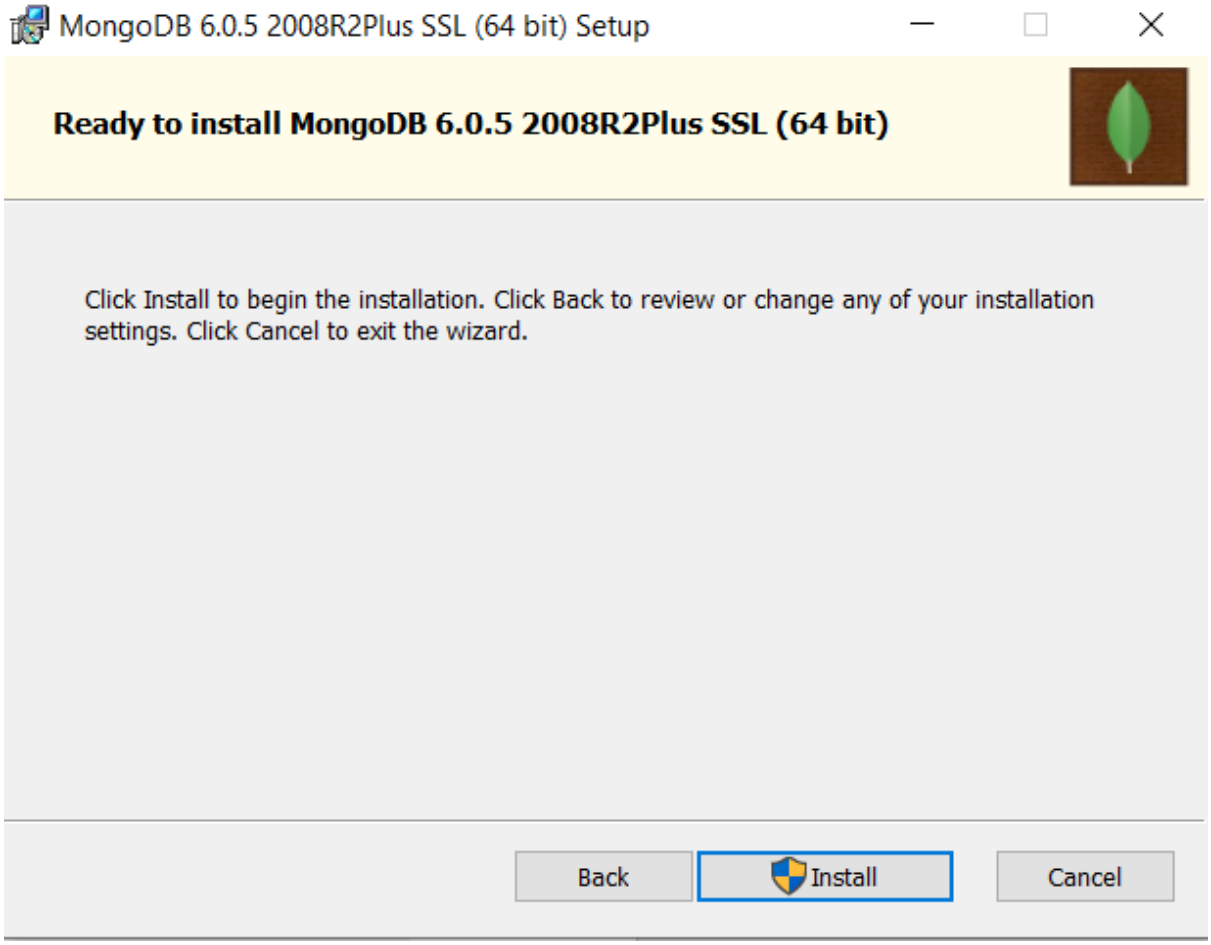
Choose the components you wish to install (typically, all components are selected by default) and click "Next".

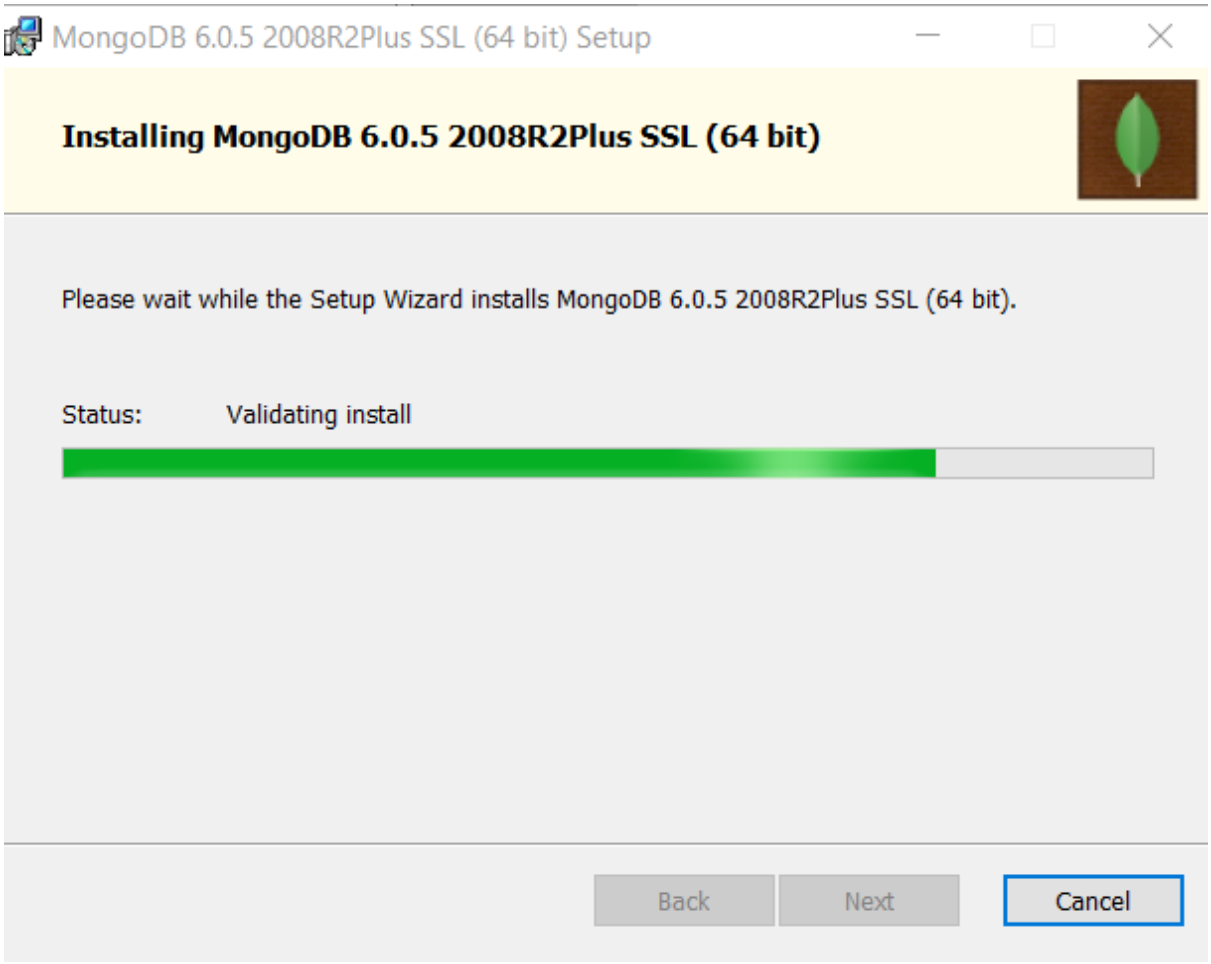


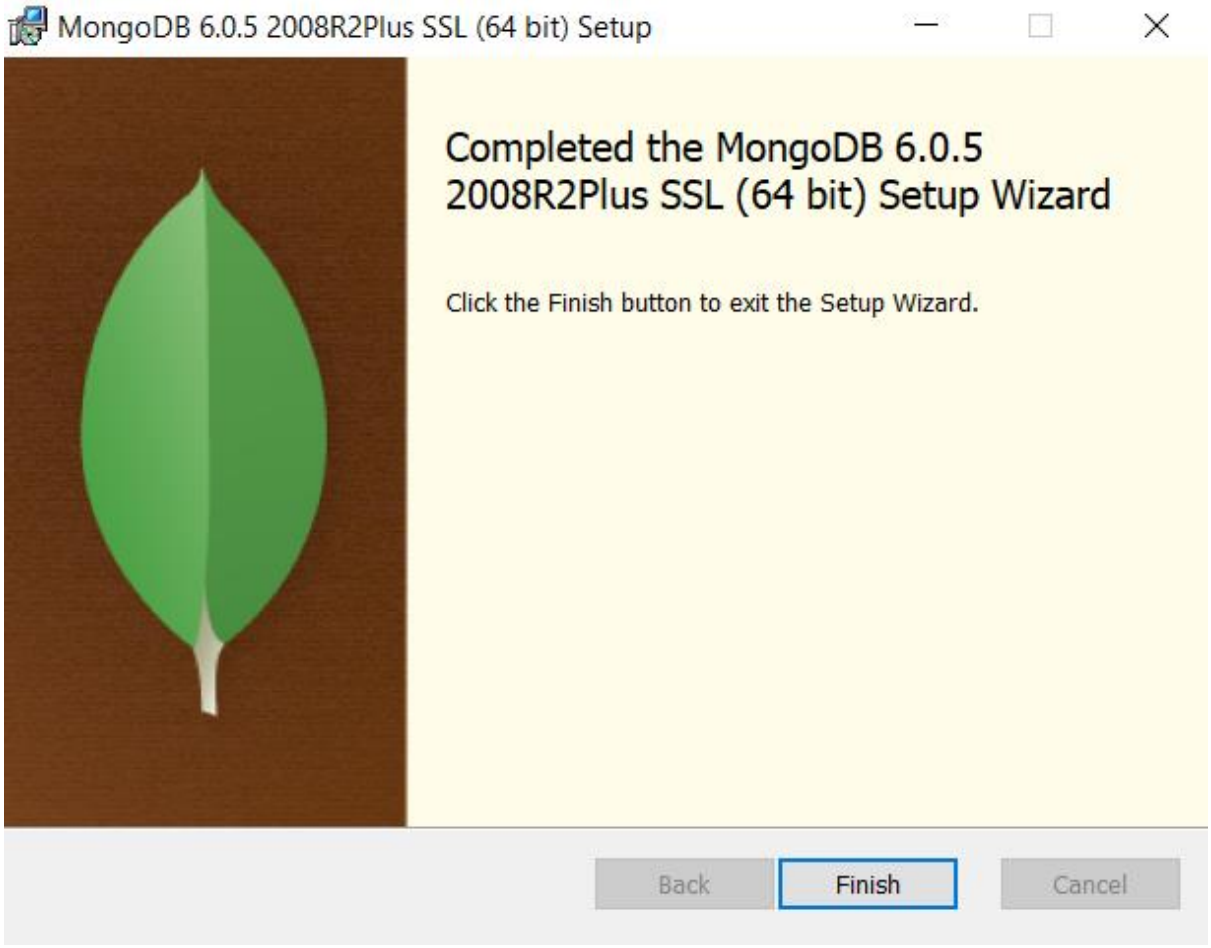
Choose whether to install MongoDB as a service (recommended), then click "Next".Choose the destination folder for the installation (the default location is typically sufficient), then click "Next".Configure the "Service Configuration" options as desired (you can leave the default settings if unsure), then click "Next".




MongoDB Compass
Install MongoDB Compass
MongoDB Compass is the official graphical user interface for MongoDB.
By checking below this installer will automatically download and install the latest version of MongoDB Compass on this machine. You can learn more about MongoDB Compass here: https://www.mongodb.com/products/compass
Install MongoDB Compass
Back
Next
Cancel

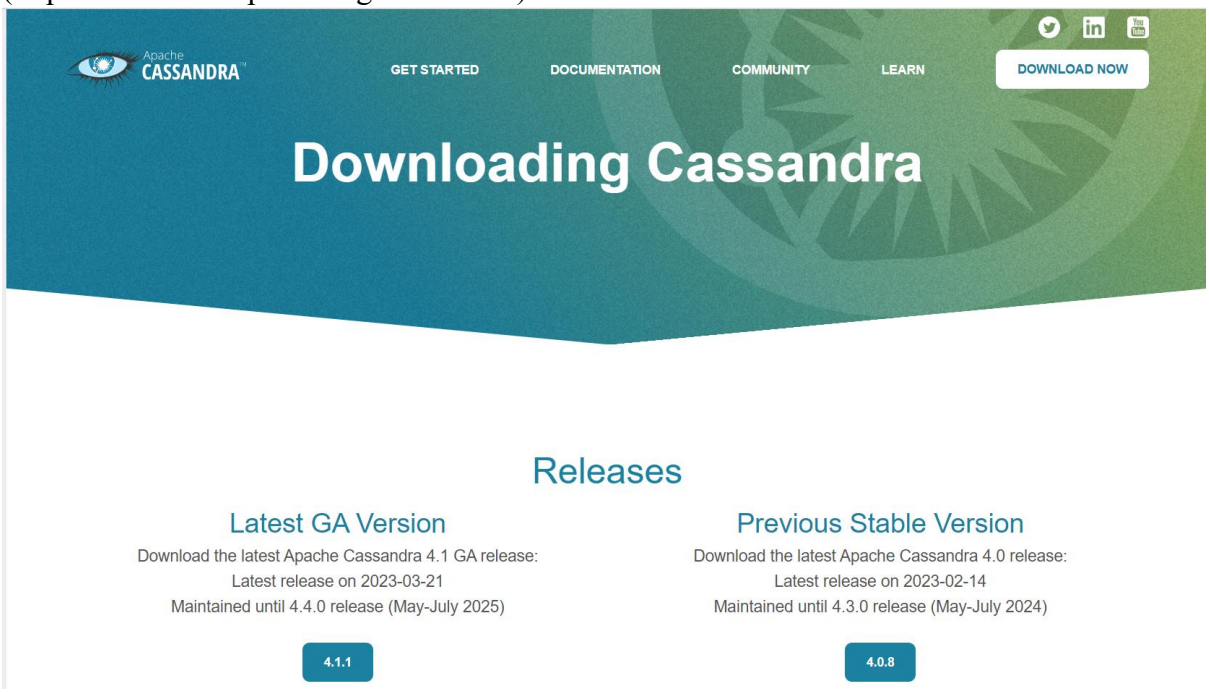

MongoDB 6.0.5 2008R2Plus SSL (64 bit) Setup
Ready to install MongoDB 6.0.5 2008R2Plus SSL (64 bit)
Click Install to begin the installation. Click Back to review or change any of your installation settings. Click Cancel to exit the wizard.
Back
Install
Cancel


MongoDB 6.0.5 2008R2Plus SSL (64 bit) Setup
Installing MongoDB 6.0.5 2008R2Plus SSL (64 bit)
Please wait while the Setup Wizard installs MongoDB 6.0.5 2008R2Plus SSL (64 bit).
Status:
Validating install
Back
Next
Cancel

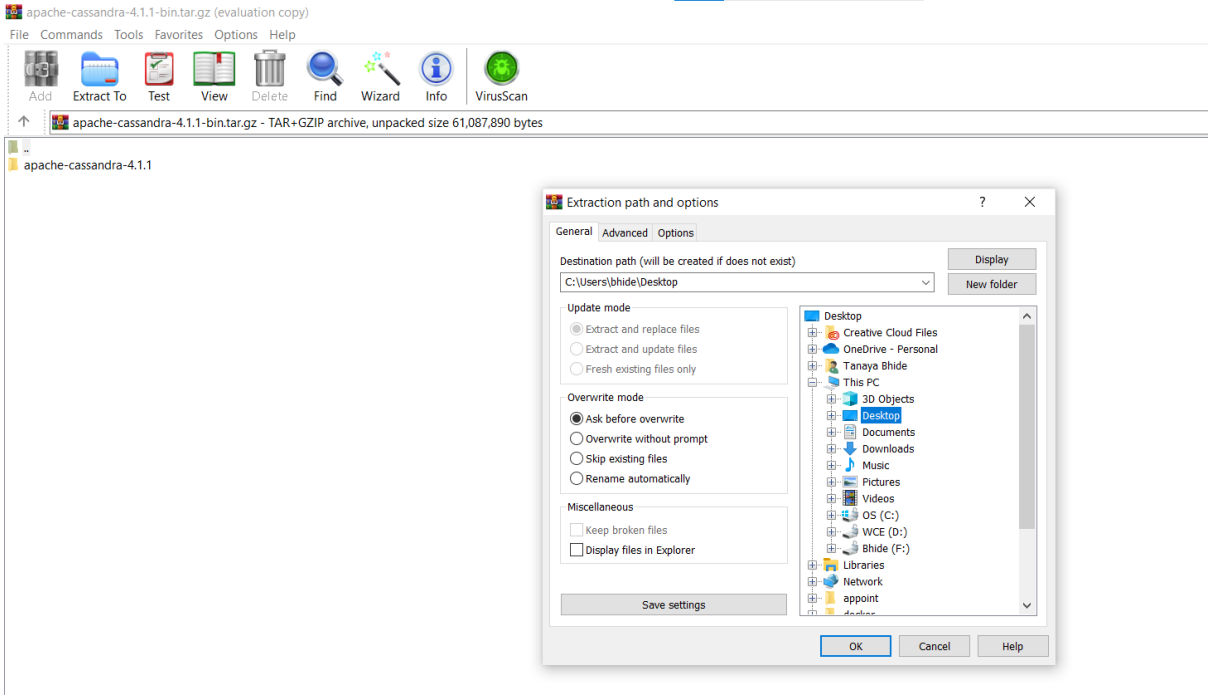

MongoDB 6.0.5 2008R2Plus SSL (64 bit) Setup
Completed the MongoDB 6.0.5 2008R2Plus SSL (64 bit) Setup Wizard
Click the Finish button to exit the Setup Wizard.
Back
Finish
Cancel

B] CassandraDB

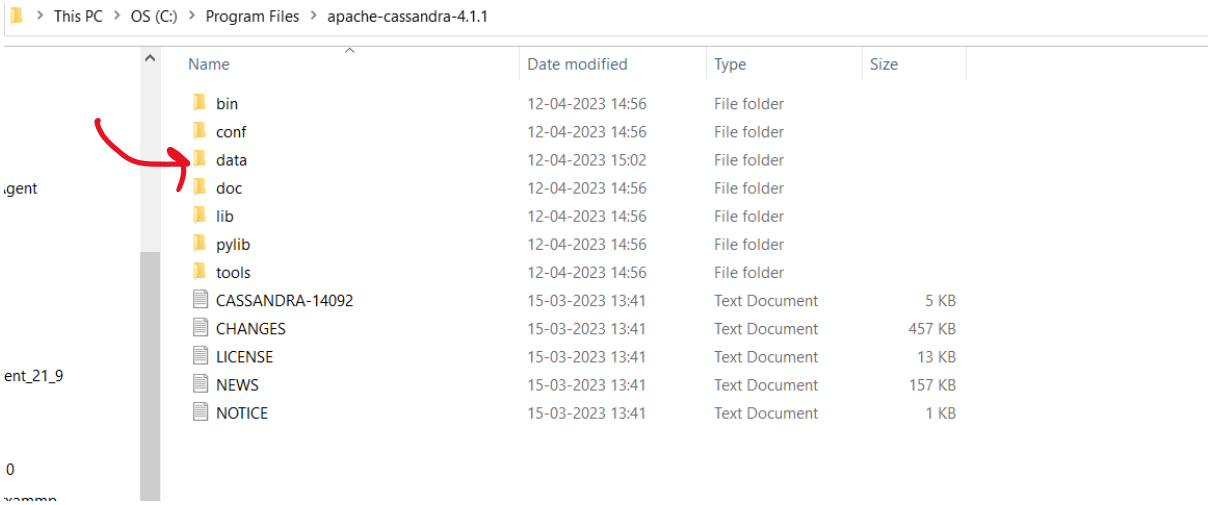
Download the latest version of CassandraDB from the official website (https://cassandra.apache.org/download/).



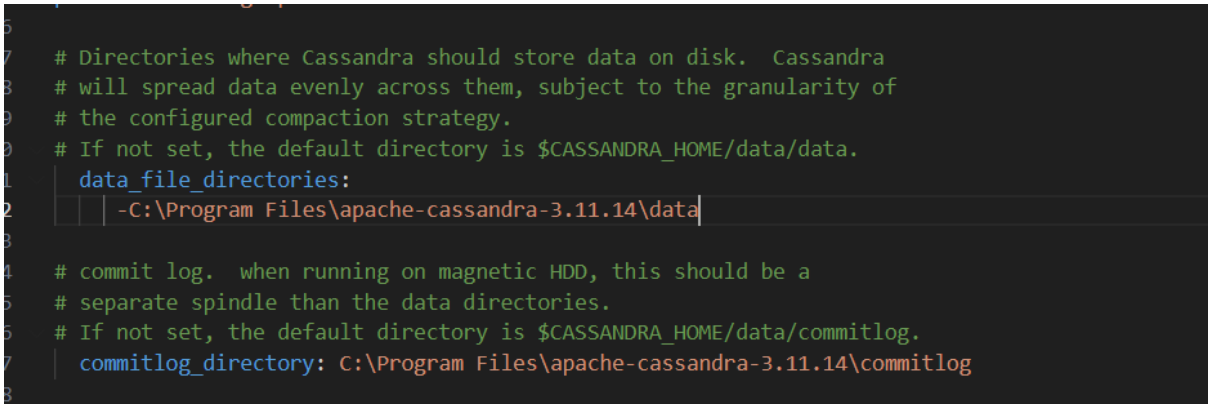
Extract the files: Extract the downloaded file to a location on your hard drive.



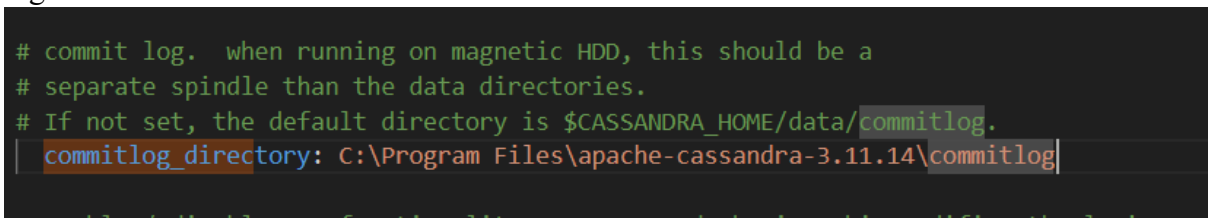
Create a new directory called "data" in the Cassandra directory.



Edit the cassandra.yaml file located in the conf directory to set the data directory path to the newly created "data" directory.



Set the "commitlog_directory" property to a directory where Cassandra can write its commit log.



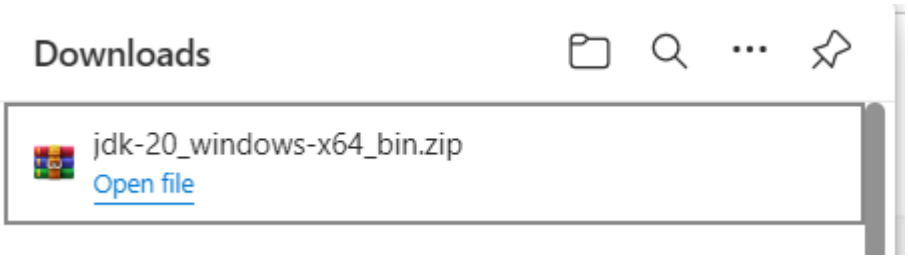
Set the "saved_caches_directory" property to a directory where Cassandra can store its saved caches.

```
# saved caches
# If not set, the default directory is $CASSANDRA_HOME/data/saved_caches.
saved_caches_directory: C:\Program Files\apache-cassandra-3.11.14\saved_caches

# Number of seconds the server will wait for each cache (row, key, etc ...) to load while starting
# the Cassandra process. Setting this to a negative value is equivalent to disabling all cache loading on startup
# while still having the cache during runtime.
# cache_load_timeout_seconds: 30
```

Save and close the cassandra.yaml file.

Cassandra Needs Java Therefore we also need to install JDK

Download and install the latest version of Java SE Development Kit (JDK) from the Oracle website: https://www.oracle.com/java/technologies/javase-downloads.html



After installation, set the JAVA_HOME environment variable to point to the installation directory of the JDK. Here's how to do it:

a. Open the Start menu and search for "Environment Variables".

b. Click on "Edit the system environment variables".

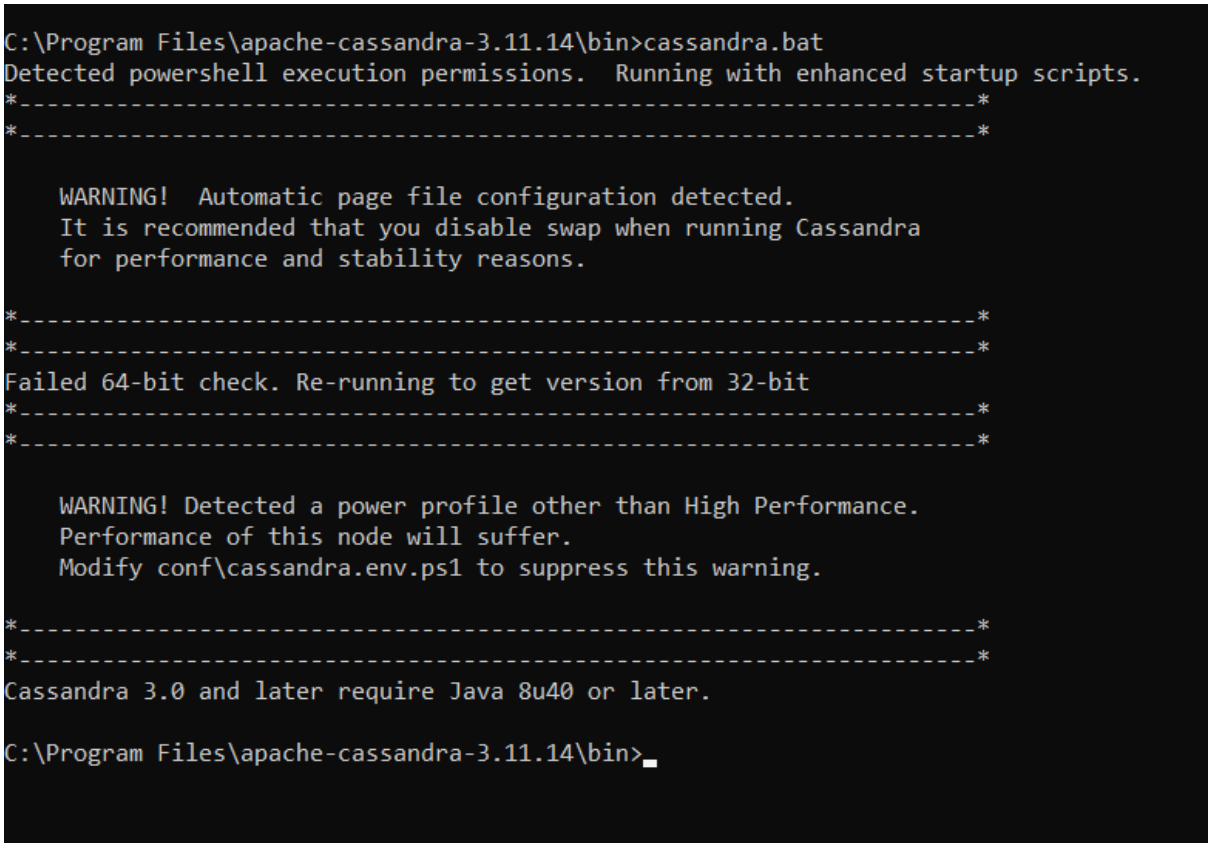
c. Click on the "Environment Variables" button at the bottom of the System Properties window.

d. Under "System Variables", click on the "New" button.

e. In the "Variable name" field, enter JAVA_HOME.

f. In the "Variable value" field, enter the path to the JDK installation directory. g. Click on "OK" to close all the windows.

Open a new command prompt window and try running cassandra.bat



Write Python desktop Application to demonstrate the CRUD operation with above backend cloud databases. Assume any database.

Code :

```
from tkinter import *
from tkinter import messagebox
import pymongo

def connect_to_mongo():
    client = pymongo.MongoClient("mongodb://localhost:27017/")
    db = client["mydatabase"]
    col = db["customers"]
    return col

def create_record(col, name, address, phone):
    mydict = {"name": name, "address": address, "phone": phone}
    col.insert_one(mydict)
    messagebox.showinfo("Success", "Record created successfully")

def read_record(col, name):
    result = col.find_one({"name": name})
    if result:
        return f"name: {result['name']}\naddress: {result['address']}\nphone: {result['phone']}\n"
    else:
        return "Record not found"

def update_record(col, name, address, phone):
    result = col.update_one({"name": name}, {"$set": {"address": address, "phone": phone}})
```

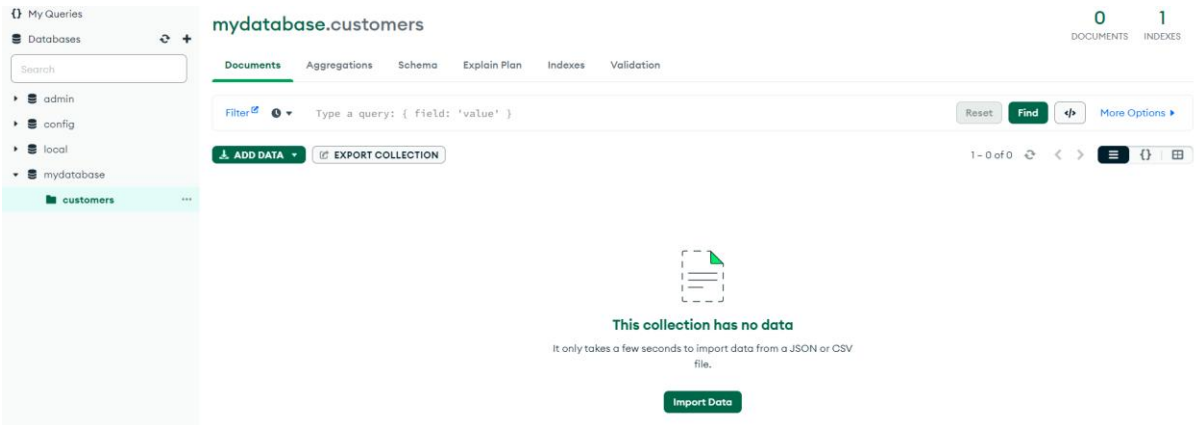
```
    if result.modified_count:
        messagebox.showinfo("Success", "Record updated successfully")
    else:
        messagebox.showerror("Error", "Record not found")

def delete_record(col, name):
    result = col.delete_one({"name": name})
    if result.deleted_count:
        messagebox.showinfo("Success", "Record deleted successfully")
    else:
        messagebox.showerror("Error", "Record not found")

def clear_fields():
    entry1.delete(0, END)
    entry2.delete(0, END)
    entry3.delete(0, END)
    text1.delete(1.0, END)

def create():
    col = connect_to_mongo()
    create_record(col, entry1.get(), entry2.get(), entry3.get())
    clear_fields()

def read():
    col = connect_to_mongo()
    result = read_record(col, entry1.get())
    text1.delete(1.0, END)
    text1.insert(END, result)

def update():
    col = connect_to_mongo()
```

```
    update_record(col, entry1.get(), entry2.get(), entry3.get())
    clear_fields()

def delete():
    col = connect_to_mongo()
    delete_record(col, entry1.get())
    clear_fields()

window = Tk()
window.title("MongoDB CRUD Application")

label1 = Label(window, text="Name")
label1.grid(row=0, column=0)
entry1 = Entry(window)
entry1.grid(row=0, column=1)

label2 = Label(window, text="Address")
label2.grid(row=1, column=0)
entry2 = Entry(window)
entry2.grid(row=1, column=1)

label3 = Label(window, text="Phone")
label3.grid(row=2, column=0)
entry3 = Entry(window)
entry3.grid(row=2, column=1)

button1 = Button(window, text="Create", command=create)
button1.grid(row=3, column=0)

button2 = Button(window, text="Read", command=read)
button2.grid(row=3, column=1)
```

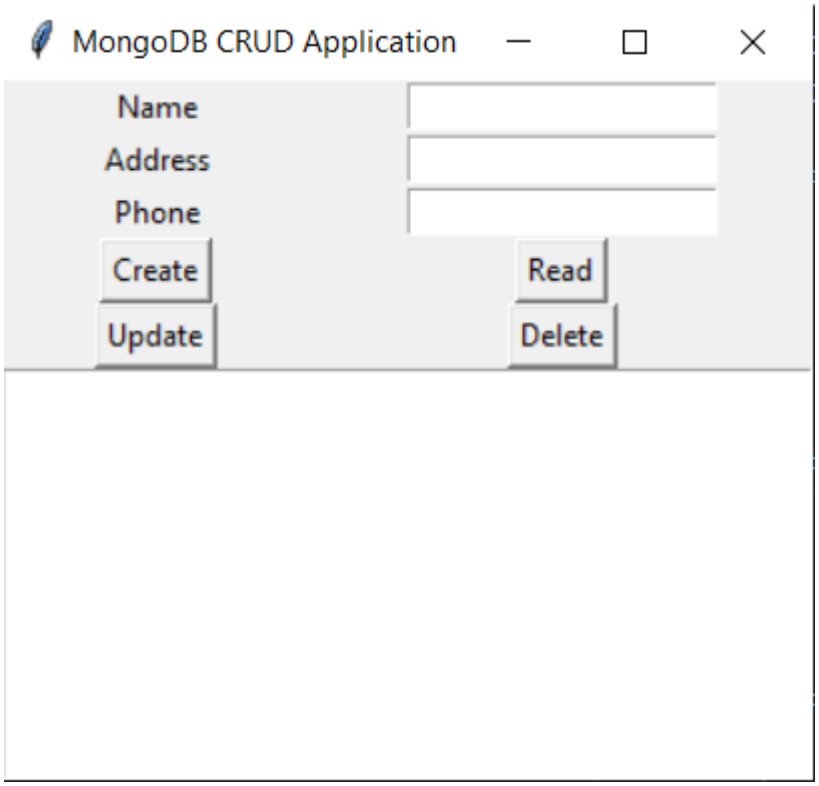
```
button3 = Button(window, text="Update", command=update)
button3.grid(row=4, column=0)

button4 = Button(window, text="Delete", command=delete)
button4.grid(row=4, column=1)

text1 = Text(window, height=10, width=40)
text1.grid(row=5, column=0, columnspan=2)

window.mainloop()
```

This code is a GUI-based application for performing CRUD (Create, Read, Update, Delete) operations on a MongoDB database using Python and Tkinter library. The user interface consists of four input fields for Name, Address, and Phone, and four buttons to perform the corresponding database operations. The create button inserts a new document into the database with the input values, the read button retrieves the document(s) with the given name and displays the information in a text field, the update button modifies the address and phone fields of the document with the given name, and the delete button removes the document with the given name from the database. The pymongo library is used to connect to the MongoDB server and perform the database operations.

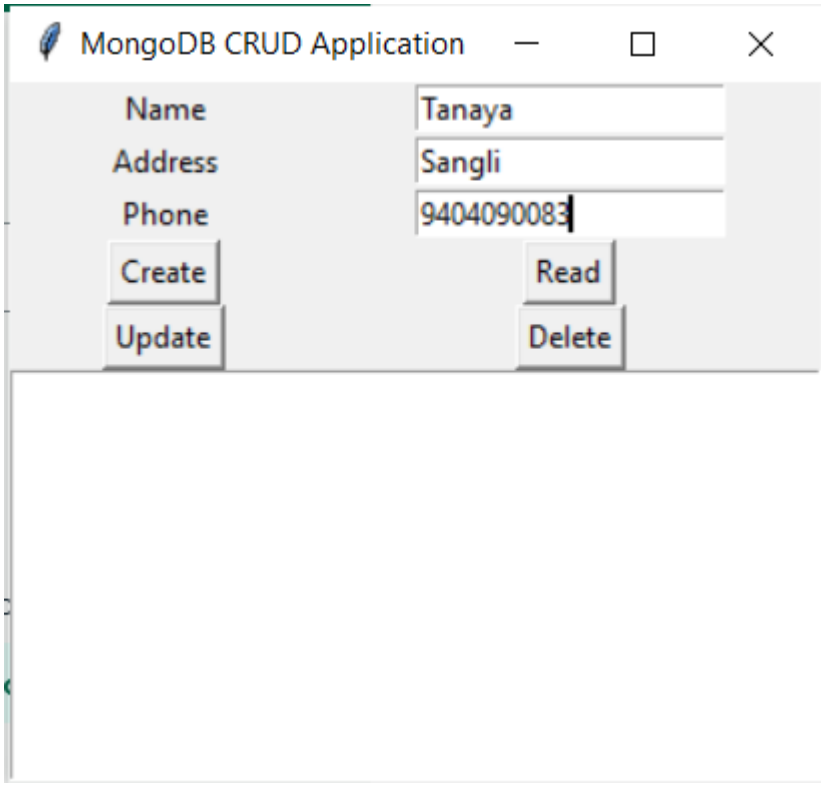
Data base view

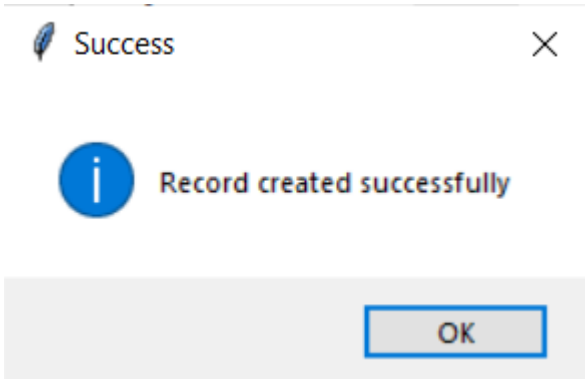


After executing our GUI application, we are presented with this window.



To create a new record, I enter the desired values into form fields and click the "create" button. Once I click it, a message confirms that the record has been successfully created. The values can then be viewed under the "customers" collection in the "mydatabase" section of MongoDB.





mydatabase.customers

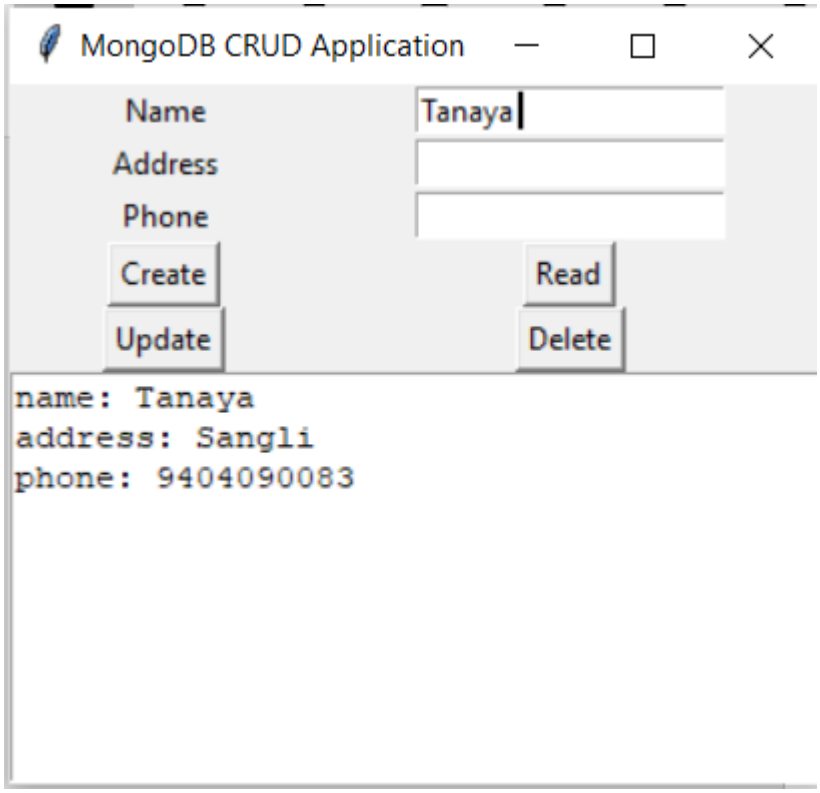
Documents | Aggregations | Schema | Explain Plan | Indexes | Validation

Filter | Type a query: { field: 'value' }

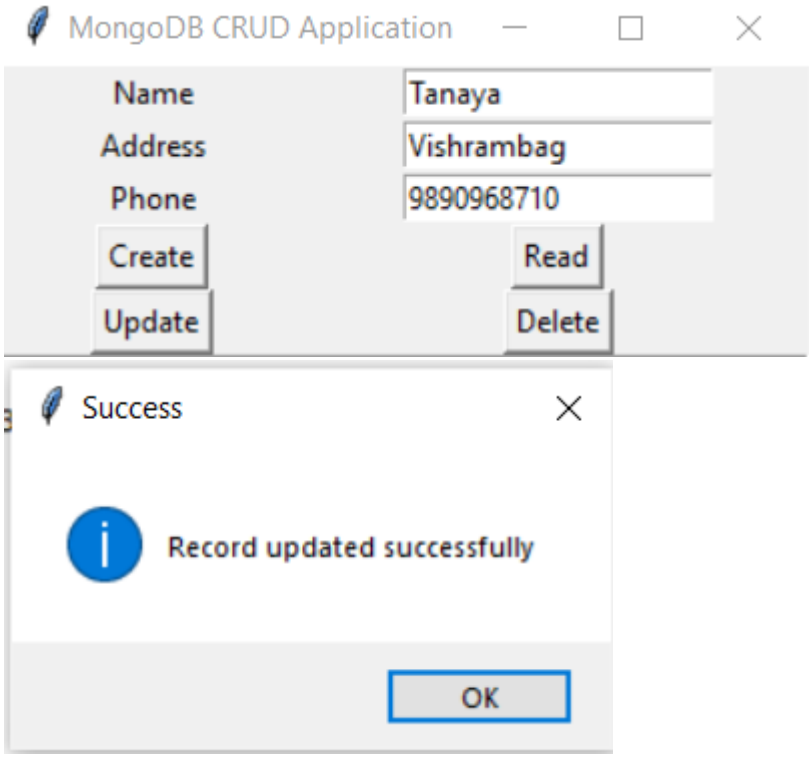
ADD DATA | EXPORT COLLECTION

```
_id: ObjectId('64368ec361b228c4700f33db')
name: "Tanaya "
address: "Sangli"
phone: "9404090083"
```

To retrieve data for a specific name, I input the name into the form fields and click the "read" button. This action displays all the relevant information retrieved from the database on the screen.

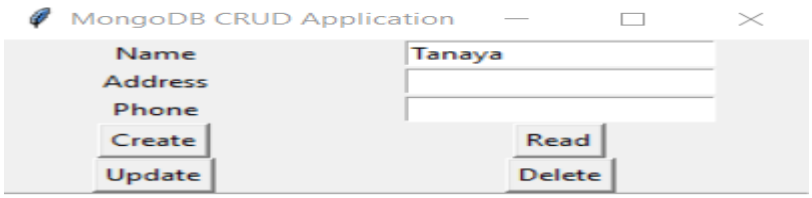


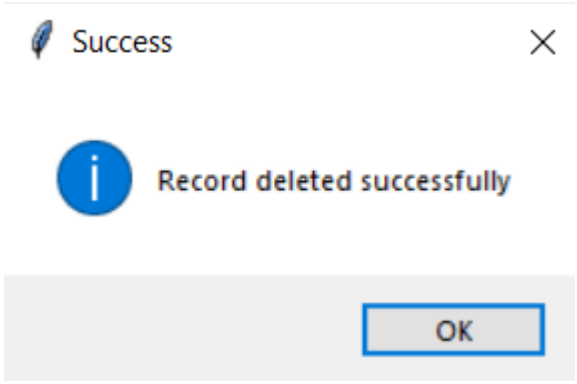
After reviewing the retrieved information, if I need to modify the address or details of a specific person, I simply update the relevant values in the form and click the "update" button. Once I click it, the record is successfully updated, and the changes can be observed in the database.



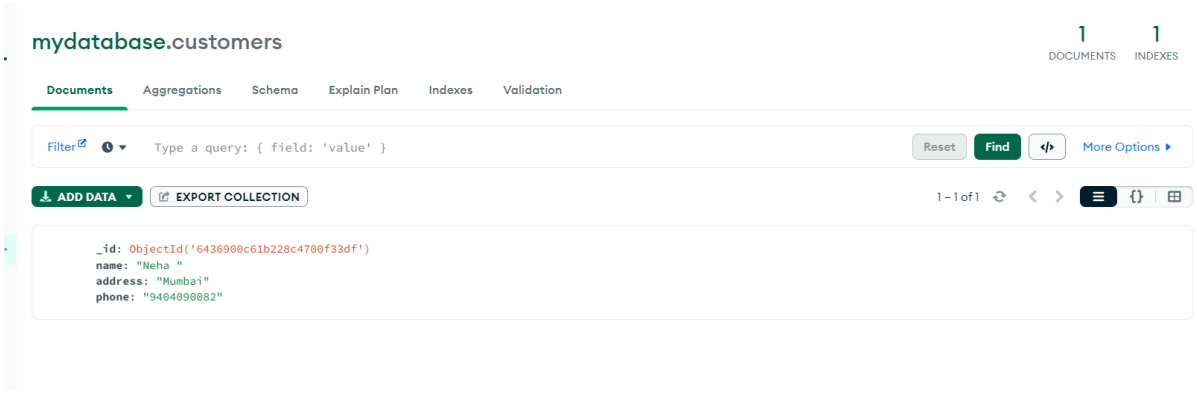
Value got updated successfully

If I enter a name into the designated field and click the "delete" button, the corresponding record is removed from the database. A message confirms that the record has been deleted successfully.

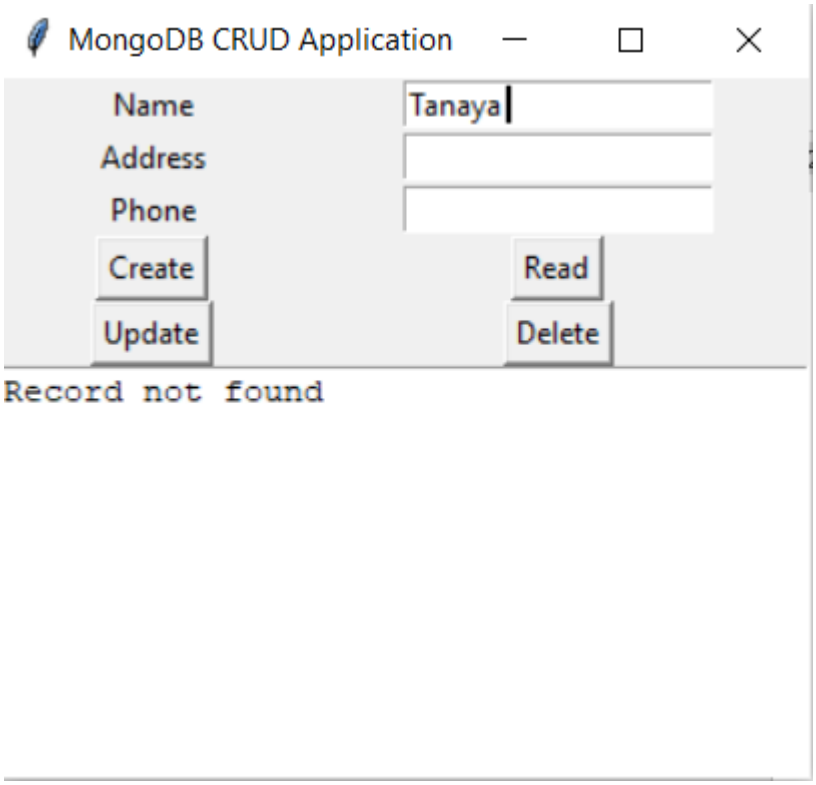




We can see now theres no record in the database with name “Tanaya”



If I try to read the record information of deleted person it says “Record Not Found”



A Python-based CRUD application utilizing MongoDB has been successfully developed