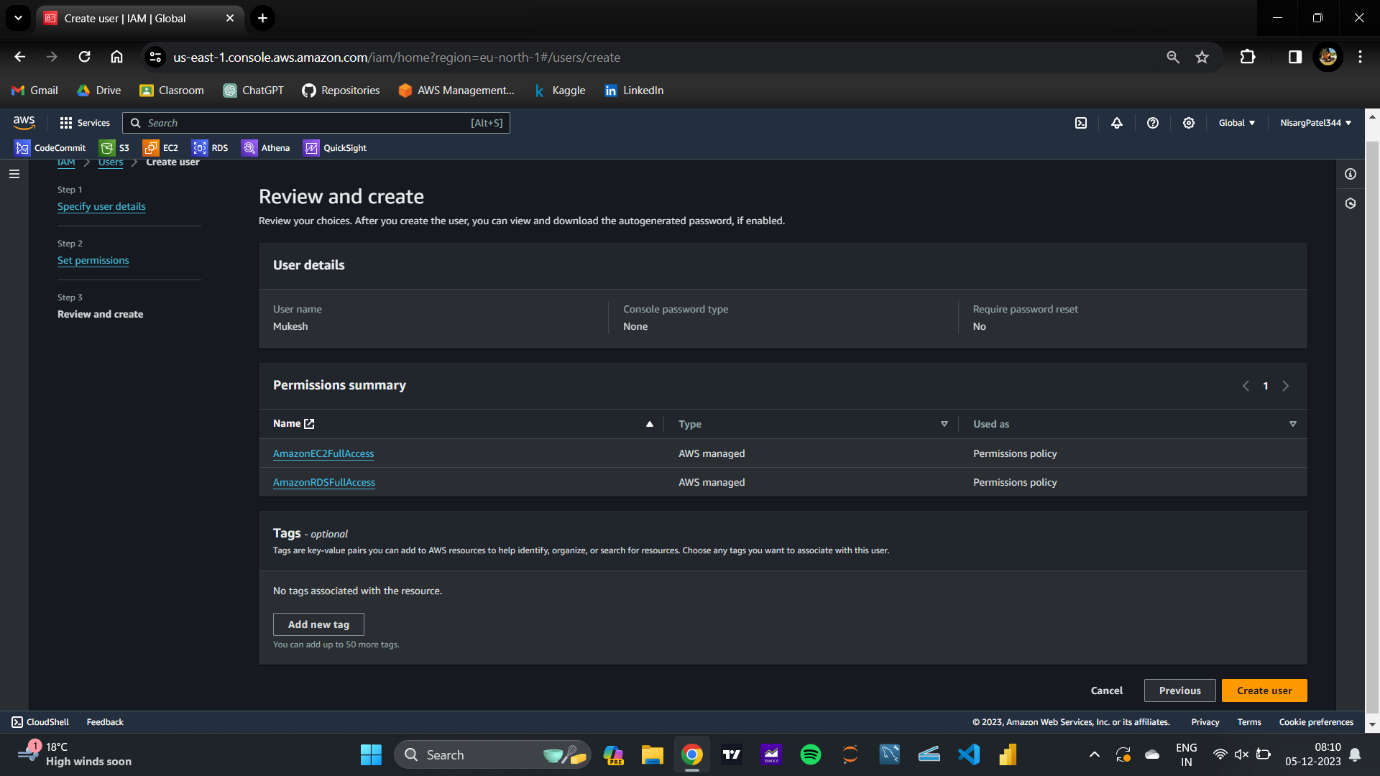
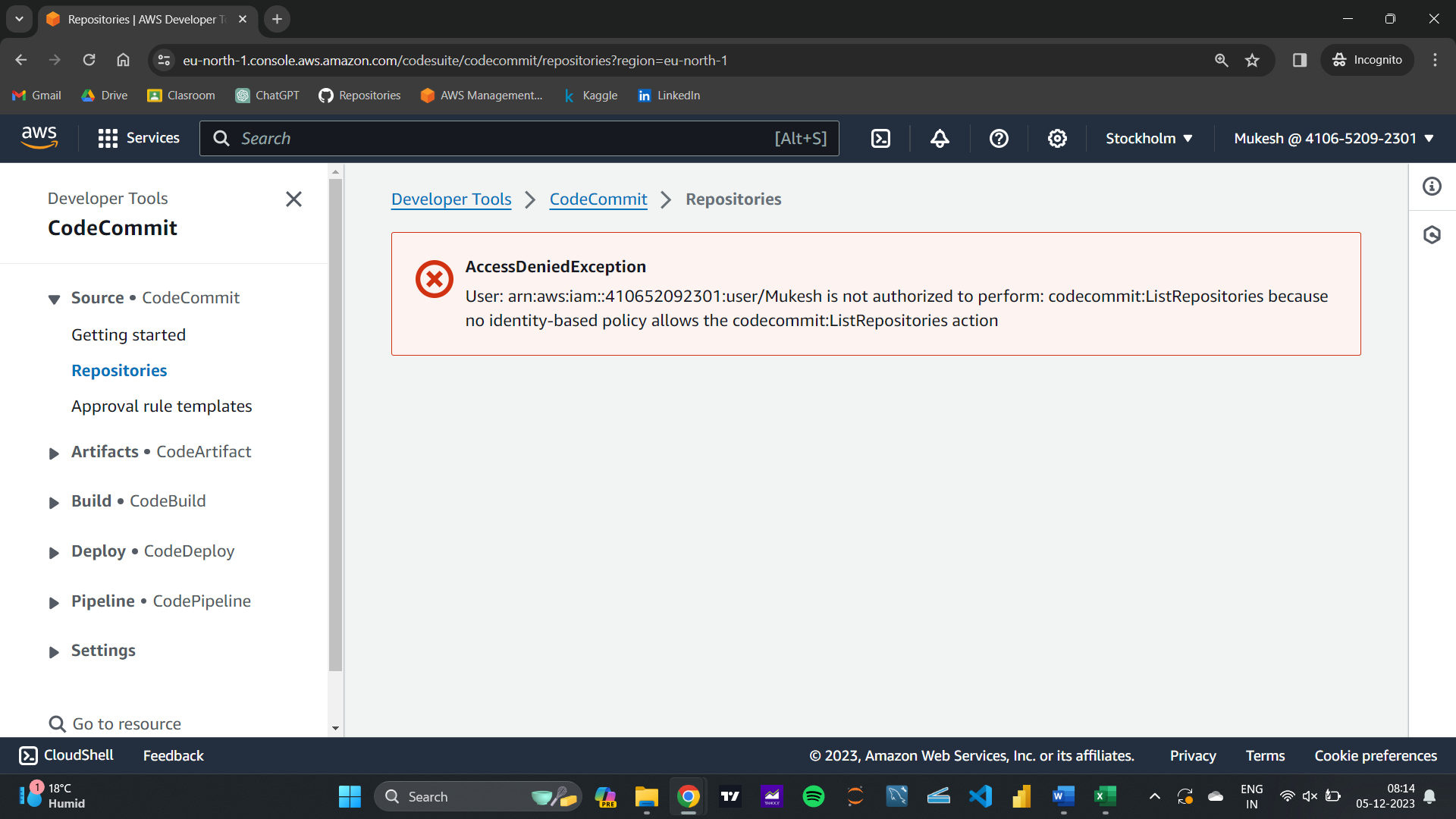
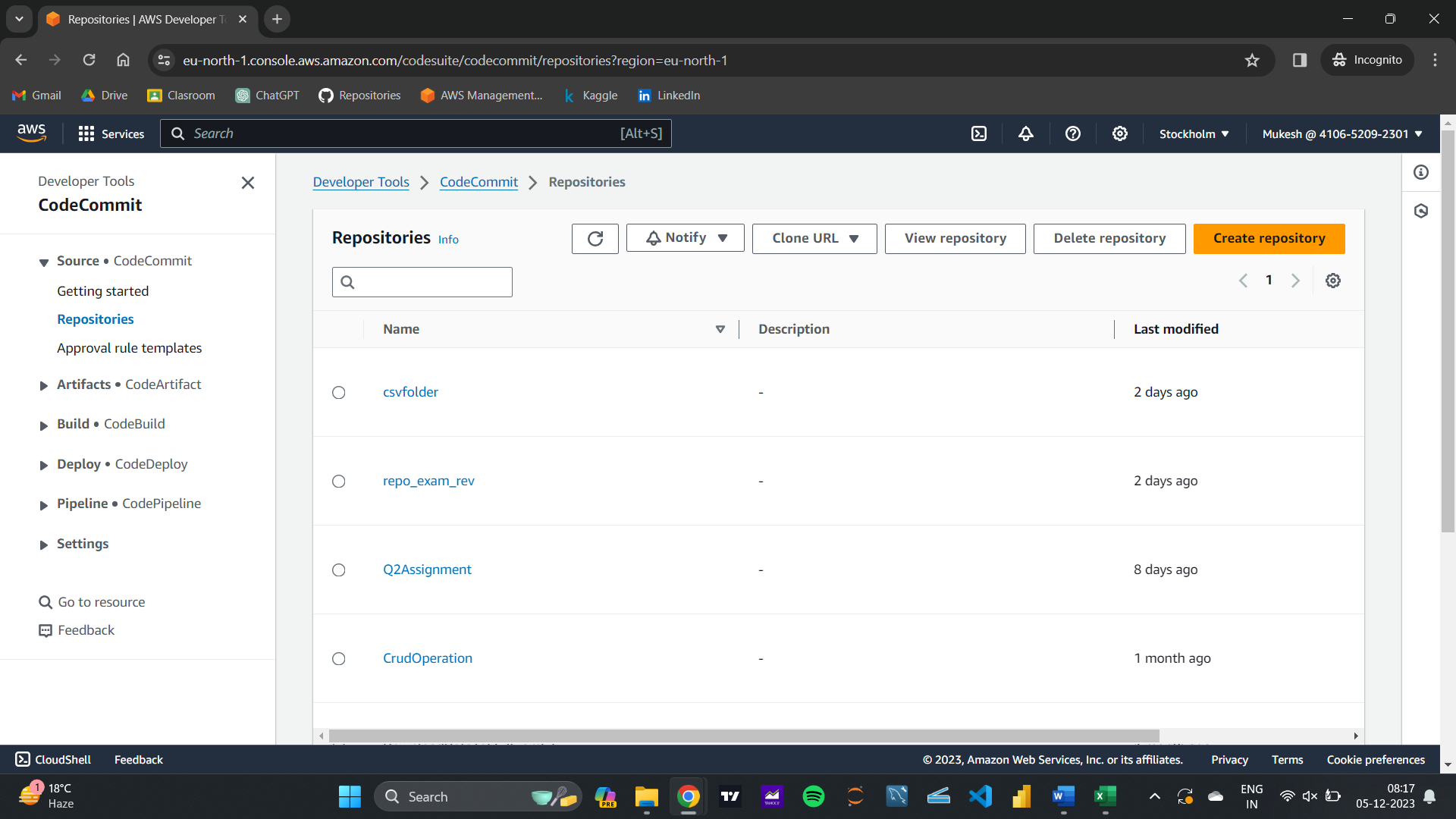
**This is the copyright of Patel Nisarg**

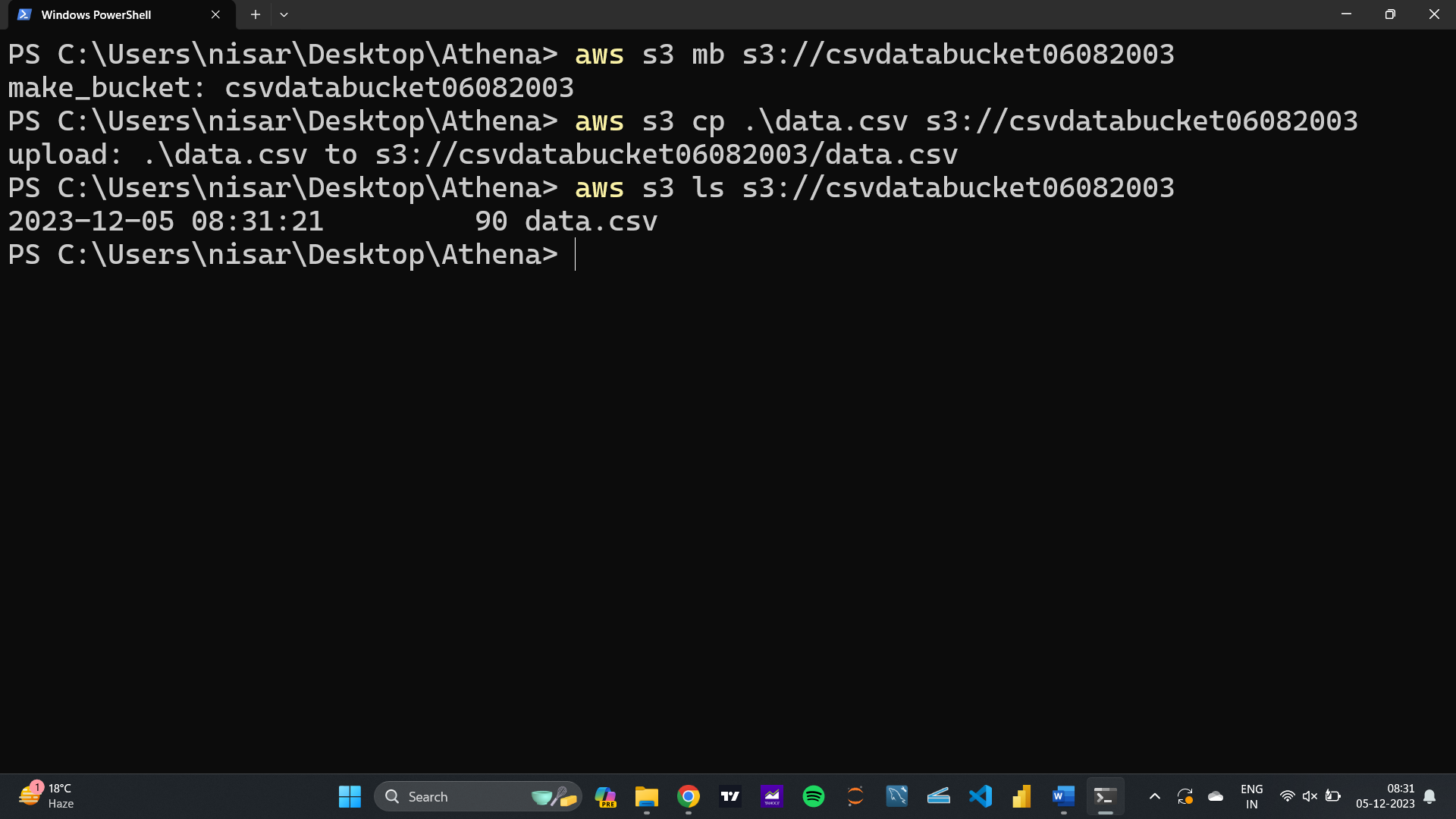
**Question 1.A**

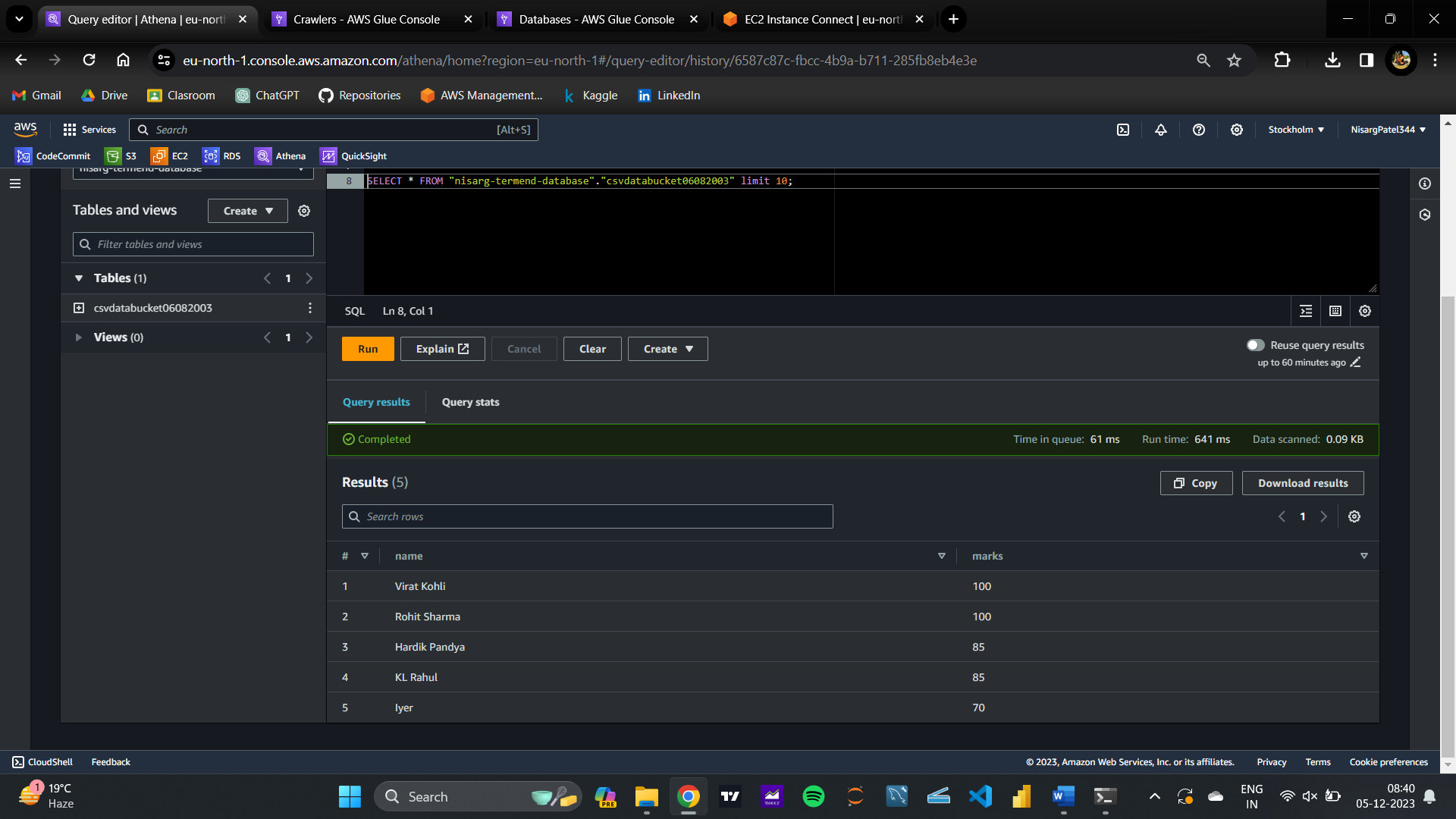




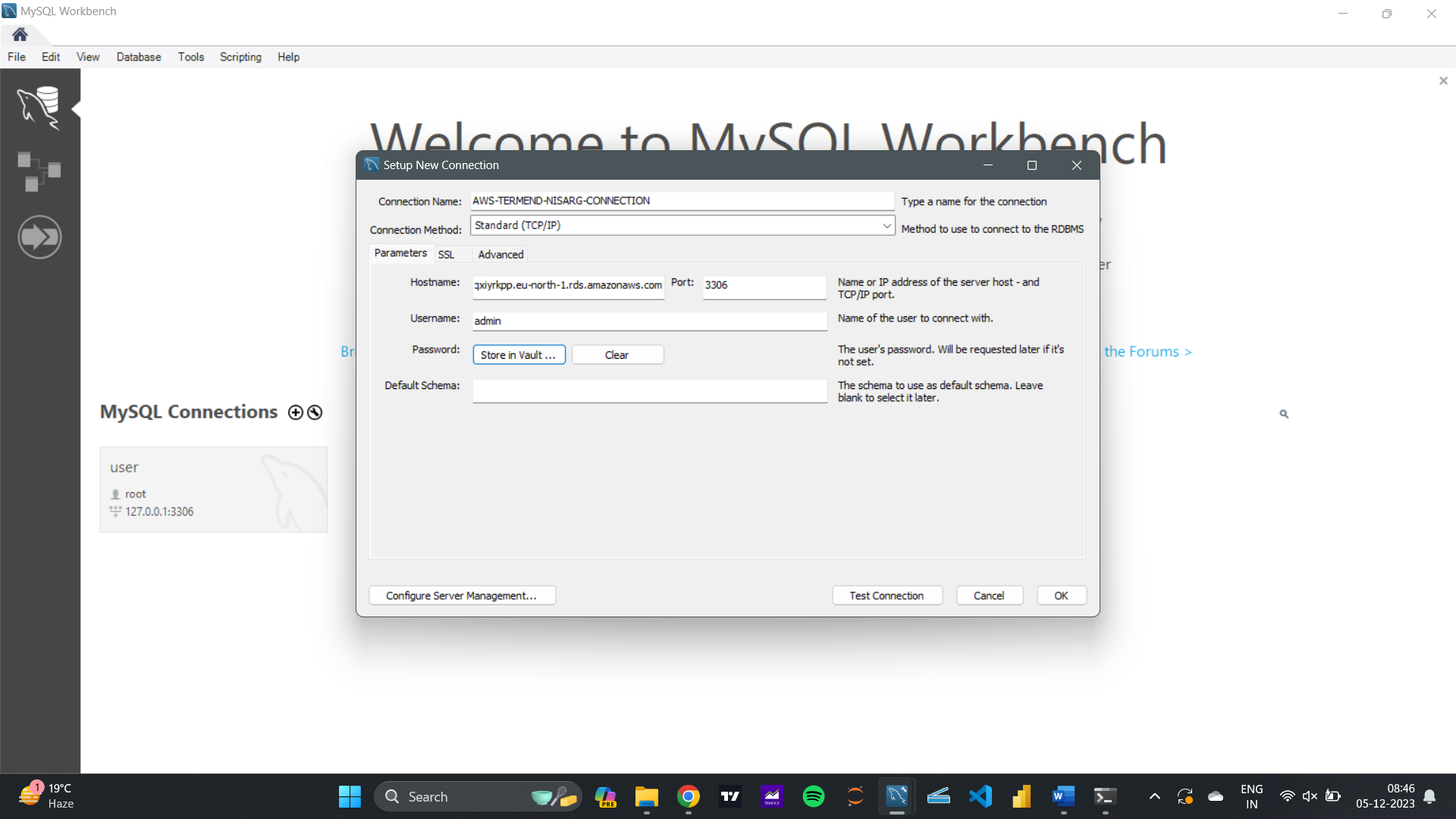


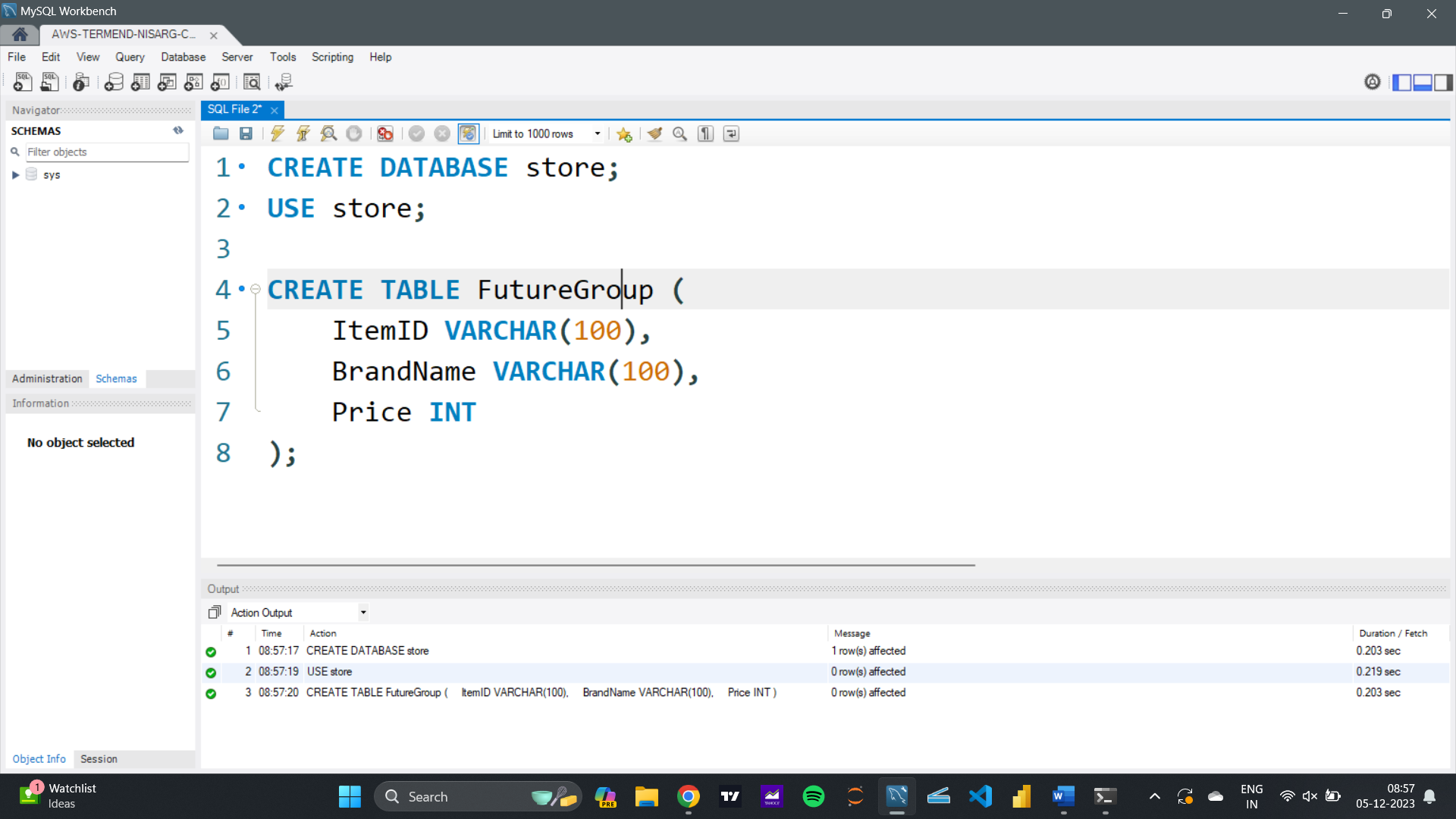
**Question 1.B**





**Question 2.B**





Python Code Made By Me For Operation RDS Database using Python

import mysql.connector

# Establish a connection to the MySQL database

mydb = mysql.connector.connect(

host="nisarg-termend-50071-database.ctifqxiyrkpp.eu-north-1.rds.amazonaws.com",

user="admin",

password="11111111"

)

# Create a cursor to execute SQL queries

mycursor = mydb.cursor()

# Initialize empty attributes list and table\_name

attributes = []

table\_name = ""

# Execute a query to show available databases

mycursor.execute("SHOW DATABASES")

databases = [db[0] for db in mycursor.fetchall()]

# Function to create a new record

def create\_record():

if not attributes:

print("No attributes defined. Please select or create a table first.")

return

values = []

placeholders = []

for attribute in attributes:

if attribute[0] == 'id':

continue

value = input(f"Enter {attribute[0]}: ")

values.append(value)

placeholders.append("%s")

insert\_query = f"INSERT INTO {table\_name} ("

insert\_query += ", ".join([attribute[0] for attribute in attributes if attribute[0] != 'id'])

insert\_query += ") VALUES ("

insert\_query += ", ".join(placeholders)

insert\_query += ")"

try:

mycursor.execute(insert\_query, values)

mydb.commit()

print("\nRecord created successfully\n")

except mysql.connector.Error as err:

print(f"Error: {err}")

# Function to read all records

def read\_records():

mycursor.execute(f"SELECT \* FROM {table\_name}")

records = mycursor.fetchall()

if not records:

print("No records found")

else:

for record in records:

print(record)

# Function to update a record

def update\_record():

if not attributes:

print("No attributes defined. Please select or create a table first.")

return

record\_id = input("Enter the ID of the record you want to update: ")

values = []

for attribute in attributes:

value = input(f"Enter new {attribute[0]}: ")

values.append(value)

update\_query = f"UPDATE {table\_name} SET "

update\_query += ", ".join([f"{attribute[0]} = %s" for attribute in attributes])

update\_query += " WHERE id = %s" # Assuming the primary key is 'id'

values.append(record\_id)

try:

mycursor.execute(update\_query, values)

mydb.commit()

print("\nRecord updated successfully\n")

except mysql.connector.Error as err:

print(f"Error: {err}")

# Function to delete a record

def delete\_record():

if not attributes:

print("No attributes defined. Please select or create a table first.")

return

record\_id = input("Enter the ID of the record you want to delete: ")

delete\_query = f"DELETE FROM {table\_name} WHERE id = %s" # Assuming the primary key is 'id'

try:

mycursor.execute(delete\_query, (record\_id,))

mydb.commit()

print("\nRecord deleted successfully\n")

except mysql.connector.Error as err:

print(f"Error: {err}")

print("Options:")

print("1. Create a new database")

print("2. Select an existing database")

choice = input("Enter your choice (1/2): ")

if choice == "1":

# Create a new database

database\_name = input("Enter the database name: ")

mycursor.execute(f"CREATE DATABASE IF NOT EXISTS {database\_name}")

mycursor.execute(f"USE {database\_name}")

table\_name = input("Enter the table name (not a reserved keyword): ")

attributes = []

while True:

attribute\_name = input("Enter an attribute name (or 'done' to finish): ")

if attribute\_name.lower() == 'done':

break

attribute\_datatype = input(f"Enter the datatype for {attribute\_name}: ")

attribute\_length = input(f"Enter the length for {attribute\_name} (or 'max' for maximum length): ")

attributes.append((attribute\_name, attribute\_datatype, attribute\_length))

create\_table\_query = f"CREATE TABLE IF NOT EXISTS `{table\_name}` ("

create\_table\_query += f"id INT AUTO\_INCREMENT PRIMARY KEY, "

for attribute in attributes:

attribute\_name, attribute\_datatype, attribute\_length = attribute

if attribute\_length.lower() == 'max':

create\_table\_query += f"`{attribute\_name}` {attribute\_datatype}, "

else:

create\_table\_query += f"`{attribute\_name}` {attribute\_datatype}({attribute\_length}), "

create\_table\_query = create\_table\_query[:-2] + ")"

mycursor.execute(create\_table\_query)

elif choice == "2":

# Select an existing database

print("Available Databases:")

for db in databases:

print(f"- {db}")

selected\_database = input("Enter the existing database name: ")

if selected\_database in databases:

mycursor.execute(f"USE {selected\_database}")

mycursor.execute(f"SHOW TABLES IN {selected\_database}")

tables = [table[0] for table in mycursor.fetchall()]

if tables:

print(f"Tables in {selected\_database}:")

for table in tables:

print(f"- {table}")

table\_name = input("Enter the existing table name: ")

mycursor.execute(f"DESCRIBE `{table\_name}`")

table\_attributes = mycursor.fetchall()

attributes = [(attr[0], attr[1]) for attr in table\_attributes]

# Check if the 'id' column is in the attributes list and remove it

attributes = [attr for attr in attributes if attr[0] != 'id']

else:

print(f"No tables found in {selected\_database}. You can create a new table.")

table\_name = input("Enter the table name: ")

else:

print("Database not found. You can create a new database.")

database\_name = input("Enter the database name: ")

while True:

print("\nCRUD Operations:")

print("1. Create a new record")

print("2. Read all records")

print("3. Update a record")

print("4. Delete a record")

print("5. Quit\n")

choice = input("Enter your choice (1/2/3/4/5):")

if choice == "1":

create\_record()

elif choice == "2":

read\_records()

elif choice == "3":

update\_record()

elif choice == "4":

delete\_record()

elif choice == "5":

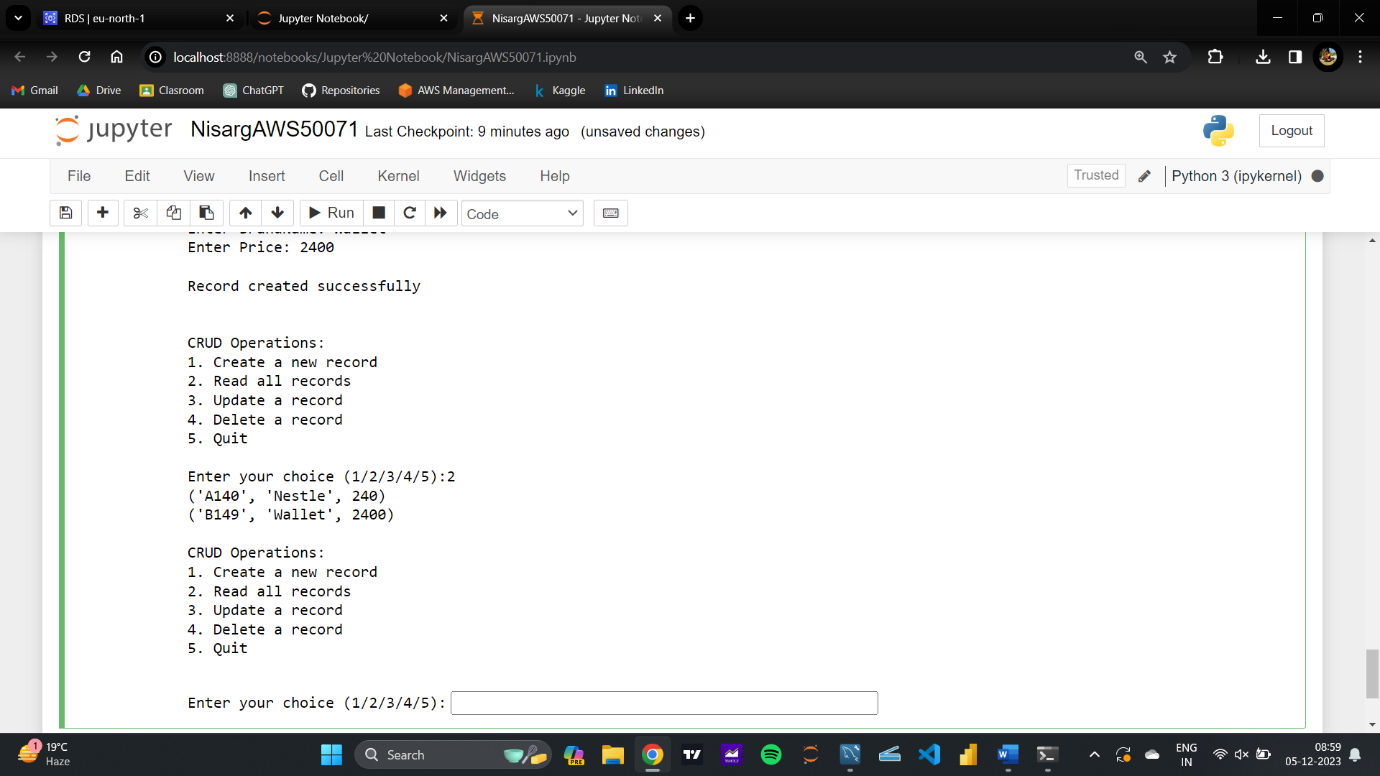
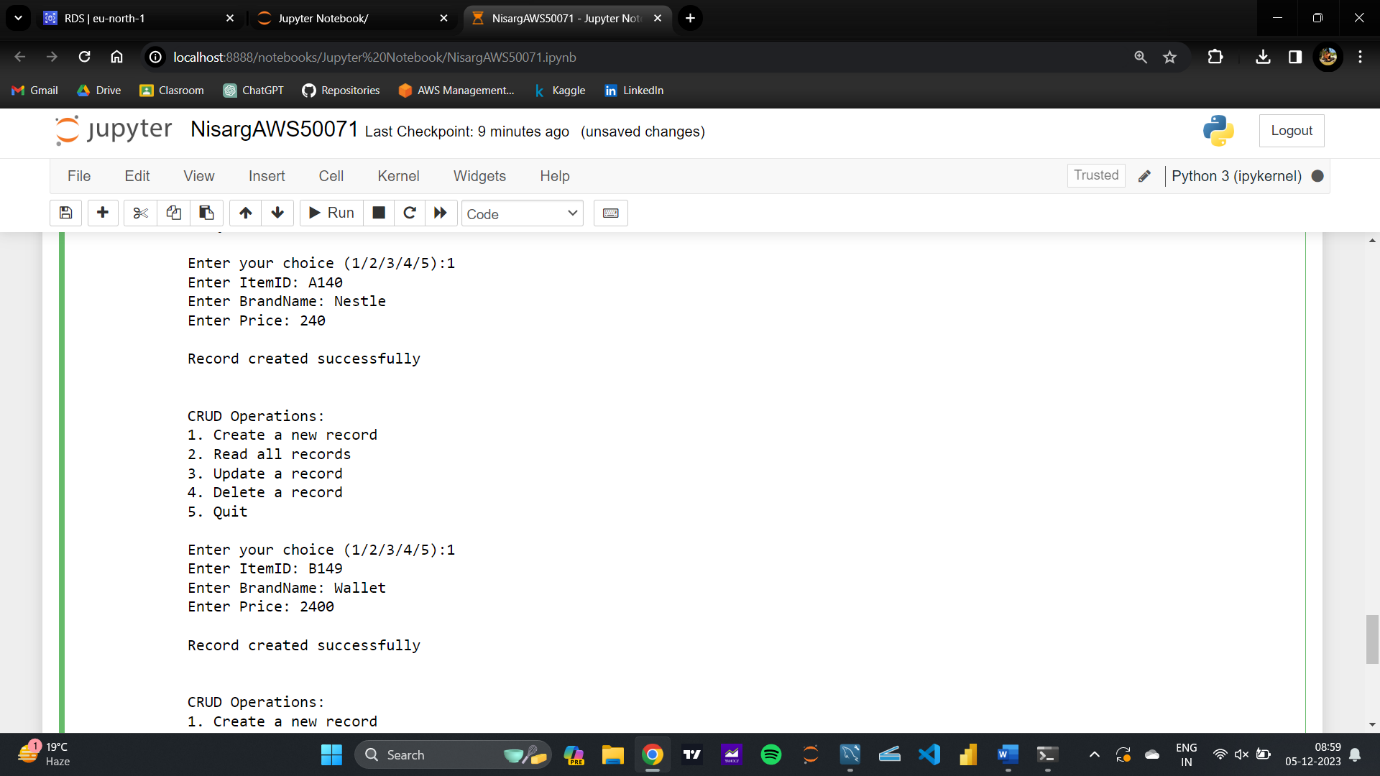
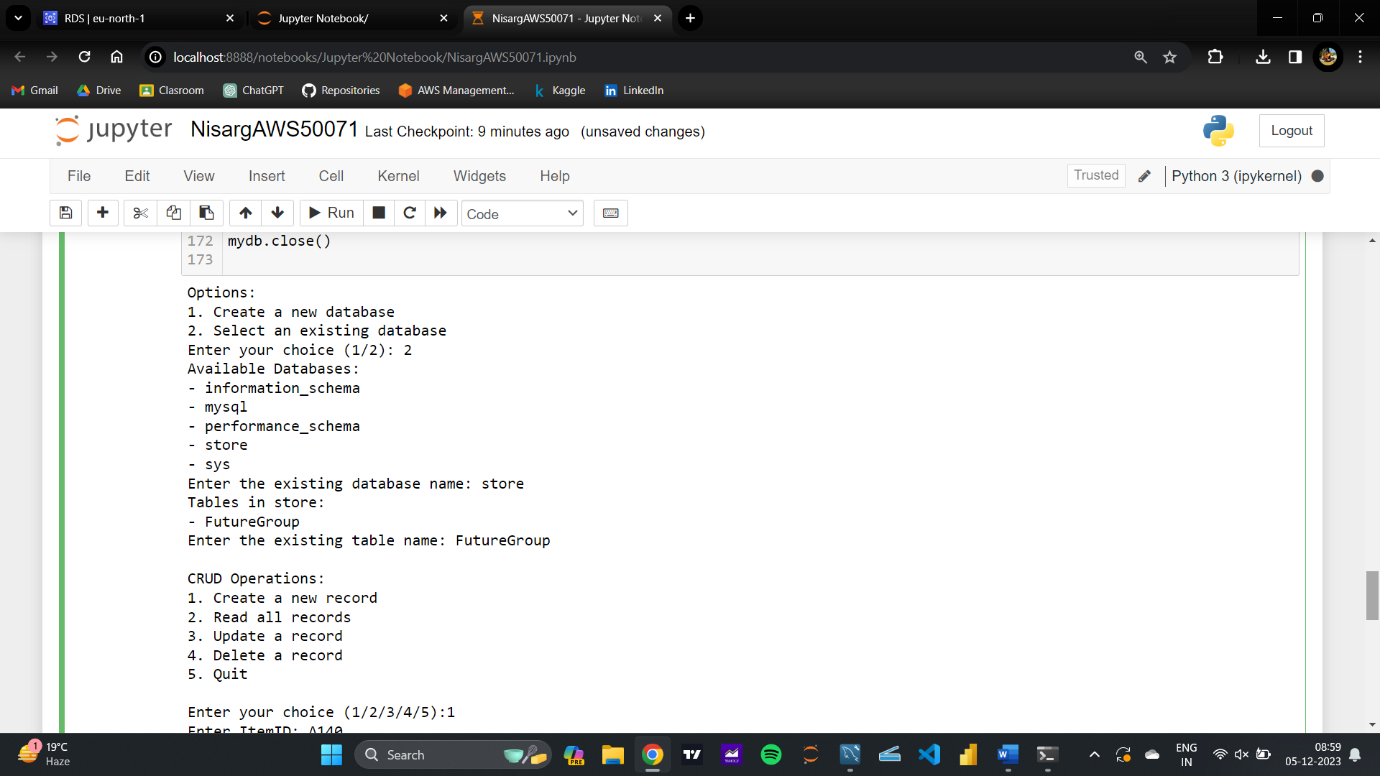
break

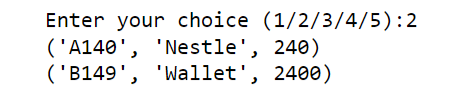
else:

print("Invalid choice. Please enter a valid option.")

# Close the database connection

mydb.close()





This is the screenshot from the upper photo to check precisely, You can verify from upper screenshot also, this is just to visualize easy