Weekly Report – 4TH week

Name :- Vaibhav Bhardwaj
Class :- MCA 3rd Semester
Roll No. :- 2222888
GitHub:- https://github.com/R34L-D34TH/college-report

Day 15:-

- MySql Connectivity

```
import mysql.connector as db

class Customer:

    def __init__(self):
        self.name = input("Enter Customer Name: ")
        self.phone = input("Enter Customer Phone: ")
        self.email = input("Enter Customer Email: ")


def main():

    customer = Customer()
    print(vars(customer))

    # DataBase Connectivity

    # Step1: Create Connection with Database
    connection = db.connect(user='root',
                            password='',
                            host='127.0.0.1',
                            database='gw2023pds1')

    # Step2: Obtain Cursor to perform SQL operations :)
    cursor = connection.cursor()

    # Step3: Create SQL Statement
    sql = "insert into Customer values" \
          "(null, '{name}', '{phone}', '{email}');".format_map(vars(customer))

    # Step4: Execute SQL Command
    cursor.execute(sql)
    connection.commit()
```

```
    print("Customer Inserted...")


if __name__ == "__main__":
    main()
```

Day 16:-

- Project Activity - CRUD operations in Customer with pandas and tabulate

```
class Customer:

    def __init__(self):
        self.cid = 0
        self.name = ""
        self.phone = ""
        self.email = ""
        self.age = 0
        self.gender = ""
        self.address = ""
        self.createdon = ""

    def read_customer_data(self):
        self.name = input("Enter Customer Name: ")
        self.phone = input("Enter Customer Phone: ")
        self.email = input("Enter Customer Email: ")
        self.age = int(input("Enter Customer Age: "))
        self.gender = input("Enter Customer Gender (male/female): ").lower()
        self.address = input("Enter Customer Address: ")
        # Get the date and time
        self.createdon = str(datetime.datetime.today())
        # Eliminate Milli Seconds
        self.createdon = self.createdon[: self.createdon.rindex(".")]

    def get_insert_sql_query(self):
        sql = "insert into Customer values(null, '{name}', '{phone}', '{email}', {age}, " \
              "'{gender}', '{address}', '{createdon}');".format_map(vars(self))
        return sql

    def get_customers_sql_query(self, phone=""):

        if len(phone) == 0:
            sql = "select * from Customer"
        else:
```

```
    else:
        sql = "select * from Customer where phone = '{}'".format(phone)
    return sql

def get_delete_sql_query(self):
    sql = "delete from Customer where cid = {}".format(self.cid)
    return sql

def get_update_sql_query(self):
    sql = "update Customer set name='{name}', phone='{phone}', email='{email}', age={age}, " \
          "gender='{gender}', address='{address}' where cid = {cid}".format_map(vars(self))
    return sql
```

Day 17:-

- Code the Vets App Project to manage pets

```
from Session16A import customer_menu
from Session16B import pets_menu
from Session16C import consultation_menu
import datetime


def main_menu():
    message = """
    >>Main Menu<<
    1: Manage Customers
    2: Manage Pets
    3: Manage Consultations
    0: Quit
    """
    print(message)
    choice = int(input("Enter Your Choice: "))

    while True:
        if choice == 1:
            customer_menu()
        elif choice == 2:
            pets_menu()
        elif choice == 3:
            consultation_menu()
        elif choice == 0:
            break
        else:
            print("Invalid Choice")

        print(message)
        choice = int(input("Enter Your Choice: "))
```

```python
def main():

    date1 = datetime.datetime.today() # to use this -> import datetime

    welcome = """
    ~~~~~~~~~~~~~~~~~~~~~~
    Welcome to Vets App
    ~~~~~~~~~~~~~~~~~~~~~~
    """
    print(welcome)

    main_menu()

    bye_message = """
    ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    Thank You for Using Vets App
    ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    """
    print(bye_message)

    date2 = datetime.datetime.today()
    print("App Usage:", date2-date1)


if __name__ == "__main__":
    main()
```

Day 18:-

- Vets App Code the Consultation

```
class Consultation:

    def __init__(self):
        self.cnid = 0
        self.cid = 0
        self.pid = 0
        self.problem = ""
        self.heartrate = 0
        self.temperature = 98.4
        self.medicines = ""
        self.createdon = ""

    def read_consultation_data(self):
        self.problem = input("Enter Problem: ")
        self.heartrate = int(input("Enter Heart Rate: "))
        self.temperature = float(input("Enter Temperature: "))
        self.medicines = input("Enter Medicines: ")

        # Get the date and time
        self.createdon = str(datetime.datetime.today())
        # Eliminate Milli Seconds
        self.createdon = self.createdon[: self.createdon.rindex(".")]

    def get_insert_sql_query(self):
        sql = "insert into Consultation values(null, {cid}, {pid}, '{problem}'," \
              "{heartrate}, {temperature}, '{medicines}', '{createdon}');".format_map(vars(self))
        return sql

    def get_consultation_sql_query(self, cid="", pid=""):
        sql = "select * from Consultation"

        if len(cid) != 0:
            sql = "select * from Consultation where cid = {}".format(cid)
```

```
    if len(pid) != 0:
        sql = "select * from Consultation where pid = {}".format(pid)

    return sql


def get_consultation_sql_query_by_date(self, date=""):
    sql = "select * from Consultation where createdon = '{}'".format(date)
    return sql


def get_delete_sql_query(self):
    sql = "delete from Consultation where cnid = {}".format(self.cnid)
    return sql
```

Day 19:-

- Introduction to Raw MongoDB CRUD Operations

```
import pymongo
import certifi # pip install certifi | If SSL error

# ca = certifi.where() # If SSL error

uri = "mongodb+srv://atpl:atpl@cluster0.eh8zx.gcp.mongodb.net/?retryWrites=true&w=majority"
client = pymongo.MongoClient(uri)
# client = pymongo.MongoClient(uri, tlsCAFile=ca) # If SSL error
db = client['gw2023pds1']
collections = db.list_collection_names()
# print(collections)

for collection in collections:
    print(collection)

documents = db['customer'].find()
for document in documents:
    print(document)
```