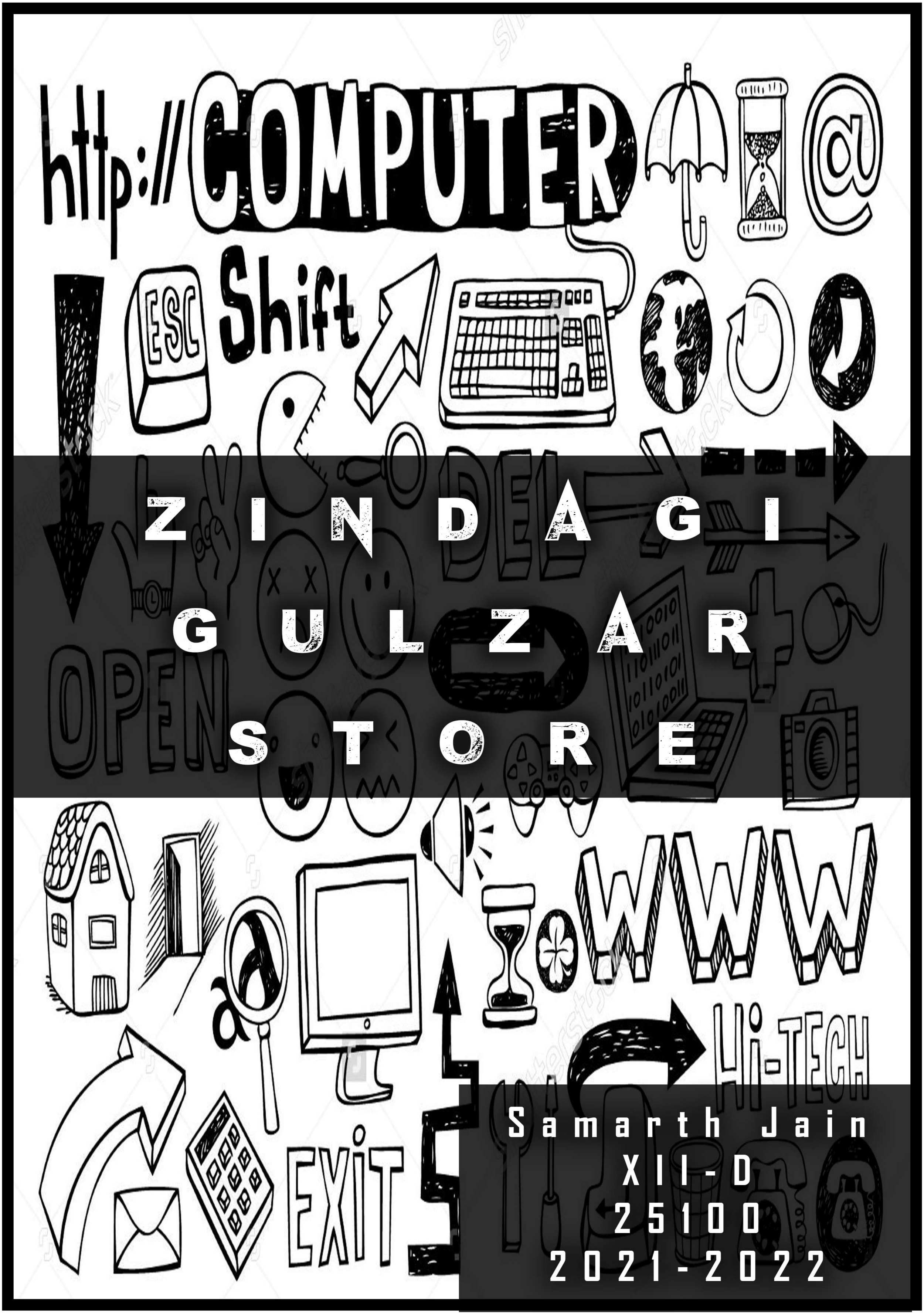

Samarth Jain
XII-D
25100
2021-2022

CERTIFICATE

This is to certify

that Samarth Jain

has worked on "Zindagi

Gulzar Store"

under my supervision

and has completed

it to my satisfaction

R. Nagpal

INDEX

AIM

DATA STORED

TABLES

MODULES & FUNCTION

CODE

OUTPUT

ACKNOWLEDGEMENT

BIBLIOGRAPHY

AIM

To create a database system for an Electronics Store to manage bills, products,taxes.

Due Taxes Table

```
mysql> select*from due_taxes;
```

Product_ID	Gst_Per_Product	Quantity_Sold	GST_Collected	Status
AP-D001	20800	0	0	Paid
AP-E010	4702	0	0	Paid
AP-L001	19798	0	0	Paid
AP-L002	25596	0	0	Paid
AP-M010	11228	0	0	Paid
AP-M011	13500	0	0	Paid
AP-M012	16852	0	0	Paid
AP-M013	29250	0	0	Paid
AS-L001	12750	0	0	Paid
AS-L002	24298	0	0	Paid
HP-L001	8456	0	0	Paid
LV-L001	8040	0	0	Paid
LV-L002	9600	0	0	Paid
MI-L001	9690	0	0	Paid
MI-M010	2950	0	0	Paid
MI-M011	1700	0	0	Paid
MS-L001	11200	0	0	Paid
MS-L002	15760	0	0	Paid
MS-L003	16800	0	0	Paid
OO-E010	600	0	0	Paid
OP-M010	3300	0	0	Paid
SG-E010	1200	0	0	Paid
SG-M010	8760	0	0	Paid
SG-M011	10680	0	0	Paid
SG-M012	3480	0	0	Paid
SY-H010	3898	0	0	Paid
VV-M010	2728	0	0	Paid
RM-M010	4760	0	0	Due

28 rows in set (0.00 sec)

Products Table

```
mysql> select*from products;
+---------------+------------+-----------------------------+----------------+-------+-------+---------------+-------+
| Serial_Number | Product_ID | product_name                | Price_Per_Unit | SGST  | CGST  | Quantity_Sold | Stock |
+---------------+------------+-----------------------------+----------------+-------+-------+---------------+-------+
|             1 | AP-D001    | Apple I-Mac 24              |         160000 | 10400 | 10400 |           311 |  1189 |
|             2 | AP-E010    | Apple Airpods Pro           |          20900 |  2351 |  2351 |            54 |  2446 |
|             3 | AP-L001    | Apple Macbook Air           |         109990 |  9899 |  9899 |             1 |   999 |
|             4 | AP-L002    | Apple Macbook Pro           |         142200 | 12798 | 12798 |            10 |   990 |
|             5 | AP-M010    | Iphone 11                   |          49900 |  5614 |  5614 |            10 |   490 |
|             6 | AP-M011    | Iphone 12                   |          60000 |  6750 |  6750 |             2 |   798 |
|             7 | AP-M012    | Iphone 13                   |          74900 |  8426 |  8426 |             0 |  1000 |
|             8 | AP-M013    | Iphone 13 Pro               |         130000 | 14625 | 14625 |            15 |   985 |
|             9 | AS-L001    | Asus Tuf Dash Gaming        |          85000 |  6375 |  6375 |             6 |  1994 |
|            10 | AS-L002    | Asus Zenbook Duo 14         |         134990 | 12149 | 12149 |            14 |  1186 |
|            11 | HP-L001    | HP 15S-DU                   |          60400 |  4228 |  4228 |             0 |   400 |
|            12 | LV-L001    | Lenovo IPG3-15              |          67000 |  4020 |  4020 |             0 |   800 |
|            13 | LV-L002    | Lenovo Legion 5i            |          80000 |  4800 |  4800 |             1 |  1499 |
|            14 | MI-L001    | Mi Notebook 14              |          57000 |  4845 |  4845 |             4 |   996 |
|            15 | MI-M010    | Mi 11 Lite                  |          29500 |  1475 |  1475 |             6 |  1994 |
|            16 | MI-M011    | Redmi 10S                   |          17000 |   850 |   850 |             2 |  1998 |
|            17 | MS-L001    | Microsoft Surface Laptop Go |          70000 |  5600 |  5600 |             1 |   999 |
|            18 | MS-L002    | Microsoft Surface Laptop 4  |          98499 |  7880 |  7880 |             0 |   500 |
|            19 | MS-L003    | Microsoft Surface Laptop 3  |         105000 |  8400 |  8400 |            20 |   480 |
|            20 | OO-E010    | Oppo Enco W11               |           5000 |   300 |   300 |          1002 |  1998 |
|            21 | OP-M010    | Oneplus Nord 2              |          29999 |  1650 |  1650 |             6 |   894 |
|            28 | RM-M010    | Realme GT2                  |          34000 |  2380 |  2380 |             0 |   500 |
|            22 | SG-E010    | Samsung Galaxy Buds         |          10000 |   600 |   600 |             4 |  2196 |
|            23 | SG-M010    | S22                         |          72999 |  4380 |  4380 |             0 |  1000 |
|            24 | SG-M011    | S22+                        |          88999 |  5340 |  5340 |            18 |   982 |
|            25 | SG-M012    | Samsung Galaxy A72          |          28999 |  1740 |  1740 |             1 |  1199 |
|            26 | SY-H010    | Sony WH-1000XM4             |          29990 |  1949 |  1949 |             9 |  1191 |
|            27 | VV-M010    | Vivo Y73                    |          20990 |  1364 |  1364 |             9 |   591 |
+---------------+------------+-----------------------------+----------------+-------+-------+---------------+-------+
28 rows in set (0.00 sec)
```

DATA STORED

Billing Table

```
mysql> select*from billing;
```

Bill_Number	Serial_No	Date	Customer_Name	Mobile_Number	product_name	Product_ID	Price_Per_Unit	Quantity	SGST	CGST	Total_Price
1	1xx1	2020-03-01	Piyush	9873312411	Apple I-Mac 24	AP-D001	160000	6	14400	14400	1132800
1	1xx2	2020-03-01	Piyush	9873312411	Iphone 13 Pro	AP-M013	130000	10	14625	14625	1592500
1	1xx3	2020-03-01	Piyush	9873312411	Apple Airpods Pro	AP-E010	20900	4	2351	2351	102408
1	1xx4	2020-03-01	Piyush	9873312411	Mi 11 Lite	MI-M010	29500	3	1475	1475	97350
1	1xx5	2020-03-01	Piyush	9873312411	Sony WH-1000XM4	SY-H010	29990	8	1949	1949	271104
2	2xx1	2020-03-16	Kronos	9213640490	Oppo Enco W11	OO-E010	2500	1	150	150	2800
3	3xx1	2020-03-19	Cellu Don	9315683225	Apple Macbook Pro	AP-L002	142200	10	12798	12798	1677960
3	3xx2	2020-03-19	Cellu Don	9315683225	Asus Zenbook Duo 14	AS-L002	134990	14	12149	12149	2230032
3	3xx3	2020-03-19	Cellu Don	9315683225	Apple Airpods Pro	AP-E010	20900	50	2351	2351	1280100
3	3xx4	2020-03-19	Cellu Don	9315683225	Oneplus Nord 2	OP-M010	29999	1	1650	1650	33299
3	3xx5	2020-03-19	Cellu Don	9315683225	S22+	SG-M011	88999	16	5340	5340	1594864
3	3xx6	2020-03-19	Cellu Don	9315683225	Apple I-Mac 24	AP-D001	160000	300	14400	14400	56640000
3	3xx7	2020-03-19	Cellu Don	9315683225	Microsoft Surface Laptop 3	MS-L003	105000	20	8400	8400	2436000
3	3xx8	2020-03-19	Cellu Don	9315683225	Oppo Enco W11	OO-E010	2500	1000	150	150	2800000
3	3xx9	2020-03-19	Cellu Don	9315683225	Vivo Y73	VV-M010	20990	8	1364	1364	189744
4	4xx1	2022-01-01	Sammy Boy	958430903	Iphone 13 Pro	AP-M013	130000	5	14625	14625	796250
4	4xx2	2022-01-01	Sammy Boy	958430903	Iphone 11	AP-M010	49900	5	5614	5614	305640
5	5xx1	2022-01-05	Aditya Johnny	890339723	S22+	SG-M011	88999	2	5340	5340	199358
5	5xx2	2022-01-05	Aditya Johnny	890339723	Samsung Galaxy A72	SG-M012	28999	1	1740	1740	32479
5	5xx3	2022-01-05	Aditya Johnny	890339723	Microsoft Surface Laptop Go	MS-L001	70000	1	5600	5600	81200
5	5xx4	2022-01-05	Aditya Johnny	890339723	Lenovo Legion 5i	LV-L002	80000	1	4800	4800	89600
6	6xx1	2022-01-10	AdiSeb	989291021	Asus Tuf Dash Gaming	AS-L001	85000	5	6375	6375	488750
6	6xx2	2022-01-10	AdiSeb	989291021	Samsung Galaxy Buds	SG-E010	10000	2	600	600	22400
6	6xx3	2022-01-10	AdiSeb	989291021	Oneplus Nord 2	OP-M010	29999	5	1650	1650	166495
6	6xx4	2022-01-10	AdiSeb	989291021	Mi 11 Lite	MI-M010	29500	3	1475	1475	97350
6	6xx5	2022-01-10	AdiSeb	989291021	Mi Notebook 14	MI-L001	57000	4	4845	4845	266760
7	7xx1	2022-02-01	Yum	882992133	Apple I-Mac 24	AP-D001	160000	3	14400	14400	566400
7	7xx2	2022-02-01	Yum	882992133	Apple I-Mac 24	MS-M003	160000	5	14400	14400	944000
7	7xx3	2022-02-01	Yum	882992133	Iphone 11	AP-M010	49900	5	5614	5614	305640
8	8xx1	2022-02-07	JJN	456780987	Oppo Enco W11	OO-E010	2500	1	150	150	2800
9	9xx1	2022-02-16	shreyas	987654567	Asus Tuf Dash Gaming	AS-L001	85000	1	6375	6375	97750
9	9xx2	2022-02-16	shreyas	987654567	Iphone 12	AP-M011	60000	1	6750	6750	73500
9	9xx3	2022-02-16	shreyas	987654567	Apple Macbook Air	AP-L001	109990	1	9899	9899	129788
10	10xx1	2022-03-02	Ryan	987765412	Apple I-Mac 24	AP-D001	160000	1	14400	14400	188800
10	10xx2	2022-03-02	Ryan	987765412	Vivo Y73	VV-M010	20990	1	1364	1364	23718
10	10xx3	2022-03-02	Ryan	987765412	Samsung Galaxy Buds	SG-E010	10000	1	600	600	11200
10	10xx4	2022-03-02	Ryan	987765412	Samsung Galaxy Buds	MI-M012	10000	1	600	600	11200
10	10xx5	2022-03-02	Ryan	987765412	Sony WH-1000XM4	SY-H010	29990	1	1949	1949	33888
10	10xx6	2022-03-02	Ryan	987765412	Redmi 10S	MI-M011	17000	1	850	850	18700
11	11xx1	2022-03-03	shampoo	848484848	Apple I-Mac 24	AP-D001	160000	1	14400	14400	188800
11	11xx2	2022-03-03	shampoo	848484848	Redmi 10S	MI-M011	17000	1	850	850	18700
12	12xx1	2022-03-03	Sambav	9999408982	Iphone 12	AP-M011	60000	1	6750	6750	73500
12	12xx2	2022-03-03	Sambav	9999408982	Samsung Galaxy Buds	SG-E010	10000	1	600	600	11200

43 rows in set (0.00 sec)

TABLES

```
mysql> show tables;
+--------------------+
| Tables_in_project |
+--------------------+
| billing            |
| due_taxes          |
| products           |
+--------------------+
3 rows in set (0.01 sec)

mysql>
```

```
mysql> desc billing;
+----------------+--------------+------+-----+---------+-------+
| Field          | Type         | Null | Key | Default | Extra |
+----------------+--------------+------+-----+---------+-------+
| Bill_Number    | int(11)      | YES  |     | NULL    |       |
| Serial_No      | varchar(100) | YES  | UNI | NULL    |       |
| Date           | date         | YES  |     | NULL    |       |
| Customer_Name  | char(25)     | YES  |     | NULL    |       |
| Mobile_Number  | varchar(10)  | YES  |     | NULL    |       |
| product_name   | char(255)    | YES  |     | NULL    |       |
| Product_ID     | varchar(10)  | YES  |     | NULL    |       |
| Price_Per_Unit | int(9)       | YES  |     | NULL    |       |
| Quantity       | int(11)      | YES  |     | NULL    |       |
| SGST           | float        | YES  |     | NULL    |       |
| CGST           | float        | YES  |     | NULL    |       |
| Total_Price    | int(9)       | YES  |     | NULL    |       |
+----------------+--------------+------+-----+---------+-------+
12 rows in set (0.03 sec)
```

```
mysql> desc due_taxes;
+------------------+--------------+------+-----+---------+-------+
| Field            | Type         | Null | Key | Default | Extra |
+------------------+--------------+------+-----+---------+-------+
| Product_ID       | varchar(10)  | YES  |     | NULL    |       |
| Gst_Per_Product  | float        | YES  | MUL | NULL    |       |
| Quantity_Sold    | int(9)       | YES  |     | NULL    |       |
| GST_Collected    | float        | YES  |     | NULL    |       |
| Status           | char(10)     | YES  |     | NULL    |       |
+------------------+--------------+------+-----+---------+-------+
5 rows in set (0.01 sec)
```

```
mysql> desc products;
+----------------+--------------+------+-----+---------+----------------+
| Field          | Type         | Null | Key | Default | Extra          |
+----------------+--------------+------+-----+---------+----------------+
| Serial_Number  | int(11)      | NO   | UNI | NULL    | auto_increment |
| Product_ID     | varchar(10)  | NO   | PRI |         |                |
| product_name   | char(255)    | YES  | UNI | NULL    |                |
| Price_Per_Unit | int(9)       | YES  |     | NULL    |                |
| SGST           | int(11)      | YES  |     | NULL    |                |
| CGST           | int(11)      | YES  |     | NULL    |                |
| Quantity_Sold  | int(11)      | YES  |     | NULL    |                |
| Stock          | int(6)       | YES  |     | NULL    |                |
+----------------+--------------+------+-----+---------+----------------+
8 rows in set (0.01 sec)
```

MODULES & FUNCTIONS

Mysql.Connector :

a driver that is required to access mysql database, you install it with “m pip install mysql-connector-python”.

Mysql.Connector.connect() :

This function sets up a connection, establishing a session with the MySQL server. It takes 4 arguments; user,password,root, database. Database is an optional argument.

MODULES & FUNCTIONS

commit() :

The commit() method of the Connection interface saves all the modifications made since the last commit.

fetchall() :

The method fetches all (or all remaining) rows of a query result set and returns a list of tuples. If no more rows are available, it returns an empty list.

MODULES & FUNCTIONS

fetchone() :

This method retrieves the next row of a query result set and returns a single sequence, or None if no more rows are available.

close() :

Use close() when you are done using a cursor. This method closes the cursor, resets all results, and ensures that the cursor object has no reference to its original connection object.

CSV :

The csv module implements classes to read and write tabular data in CSV format.

with open() :

This function returns a file object, also called a handle, as it is used to read or modify the file accordingly.

MODULES & FUNCTIONS

csv.writer() :

Return a writer object responsible for converting the user's data into delimited strings on the given file-like object.

writerows():

Write all elements in rows (an iterable of row objects as described above) to the writer's file object, formatted according to the current dialect.

CODE

```
import mysql.connector
con=mysql.connector.connect(user='root',password='',
host='localhost',database='project')

def add_data():
  l="select max(bill_number) from billing"
  cur=con.cursor()
  cur.execute(l)
  rec=cur.fetchall()
  B=rec[0][0]
  bill_no=B+1
  name=input("Enter Customer Name: ")
  date=input("Enter Date of Billing: ")
  mobile_no=int(input('Enter Mobile Number: '))
  no_of_item=int(input('Number of Items Purchased: '))
  maxitem=9
  if (no_of_item>maxitem):
      print()
      print("Please Enter Less than 9 items or Split the
      bill into Two")
  elif (no_of_item≤maxitem):
        for a in range(1,no_of_item+1):
          s_no=str(bill_no)+'xx'+str(a)
          product_code=input('Enter Product Code: ')
          s="select * from products where
          product_id='{}'".format(product_code)
          cur=con.cursor()
          cur.execute(s)
          rec=cur.fetchall()
          for r in rec:
                  product_name=r[2]
                  mrp=r[3]
                  sgst=r[4]
                  cgst=r[5]
                  qty=r[6]
                  stock_available=r[7]
          quantity=int(input("Enter Quantity: "))
          t=(mrp+sgst+cgst)*quantity
```

CODE

```
i="insert into billing values({},'{}','{}','{}','{}',
'{}','{}',{},{},{},{},{})".format(bill_no,s_no,date,
name,mobile_no,product_name,product_code,mrp,quantity,
sgst,cgst,t)
cur=con.cursor()
cur.execute(i)
con.commit()
p="update products set stock={} where product_id='{}'"
.format(stock_available-quantity,product_code)
cur=con.cursor()
cur.execute(p)
con.commit()
w=qty+quantity
q="update products set quantity_sold={} where
product_id='{}'".format(w,product_code)
cur=con.cursor()
cur.execute(q)
con.commit()

 d="select * from due_taxes where product_id='{}'"
.format(product_code)
cur=con.cursor()
cur.execute(d)
record=cur.fetchall()
for u in record:
    quantity_sold=u[2]+quantity
    gst_collected=u[3]+(sgst+cgst)*quantity
    v="update due_taxes set quantity_sold={},
gst_collected={},status='Due' where product_id='{}'"
.format(quantity_sold,gst_collected,product_code)
    cur=con.cursor()
    cur.execute(v)
    con.commit()
    print_bill(bill_no)
    print()
print()
print("Invoice Generated")
print()
```

CODE

```
def search_bill():
    t=0
    bill_no=int(input("Enter Bill Number You Want To
    Search: "))
    mycursor=con.cursor()
    mycursor.execute("SELECT * FROM billing WHERE
    Bill_Number={}".format(bill_no))
    myresult = mycursor.fetchall()
    for x in myresult:
        print()
        print("Bill Number: ",x[0])
        print("Customer Name: ",x[3])
        print("Mobile Number: ",x[4])
        print("Date: ",x[2])
        print()
        break
    p="select serial_no from billing where
    bill_number={}".format(bill_no)
    mycursor.execute(p)
    record=mycursor.fetchall()
    for i in record:
        for j in i:
            s=('select*from billing where serial_no="{}"
            '.format(j))
            cur=con.cursor()
            cur.execute(s)
            rec=cur.fetchall()
            for x in rec:
                print("Serial Number: ",i[-1][-1])
                print("Product Name: ",x[5])
                print("Product ID: ",x[6])
                print("Price Per Unit: ",x[7])
                print("Quantity: ",x[8])
                print("SGST: ",x[9])
                print("CGST: ",x[10])
                print("Total: ",x[11])
                t=t+x[11]
                print()
```

CODE

```
print("Grand Total: ",t)
    print("Do you want to Print Invoice?")
    ch=input("Enter Choice: ")
    if ch=="yes":
        print_bill(bill_no)
        print()
        print("Invoice Generated")
        print()
    else:
        print()
        pass

def search_name():
  t=0
  i=input("Enter Customer Name You Want To Search: ")
  mycursor=con.cursor()
  mycursor.execute("SELECT * FROM billing WHERE
  Customer_Name='{}'".format(i))
  myresult = mycursor.fetchall()
  for y in myresult:
        print()
        print("Bill Number: ",y[0])
        print("Customer Name: ",y[3])
        print("Mobile Number: ",y[4])
        print("Date: ",y[2])
        print()
        break
  p="select serial_no from billing where
  customer_name='{}'".format(i)
  mycursor.execute(p)
  record=mycursor.fetchall()
  for i in record:
        for j in i:
            s=('select*from billing where
            serial_no="{}"'.format(j))
            cur=con.cursor()
            cur.execute(s)
            rec=cur.fetchall()
```

CODE

```
for x in rec:
        print("Serial Number: ",i[-1][-1])
        print("Product Name: ",x[5])
        print("Product ID: ",x[6])
        print("Price Per Unit: ",x[7])
        print("Quantity: ",x[8])
        print("SGST: ",x[9])
        print("CGST: ",x[10])
        print("Total: ",x[11])
        t=t+x[11]
        print()
    print("Grand Total: ",t)
    print()
    print()

def search_mobile():
    t=0
    i=int(input("Enter Mobile Number You Want
    To Search: "))
    mycursor=con.cursor()
    mycursor.execute("SELECT * FROM billing WHERE
    Mobile_Number='{}'".format(i))
    myresult = mycursor.fetchall()
    for x in myresult:
            print()
            print("Bill Number: ",x[0])
            print("Customer Name: ",x[3])
            print("Mobile Number: ",x[4])
            print("Date: ",x[2])
            print()
            break
    p="select serial_no from billing where Mobile_Number=
    '{}'".format(i)
    mycursor.execute(p)
    record=mycursor.fetchall()
    for i in record:
            for j in i:
```

CODE

```
s=('select*from billing where serial_no="{}"'.format(j )
cur=con.cursor()
cur.execute(s)
rec=cur.fetchall()
for x in rec:
     print("Serial Number: ",i[-1][-1])
     print("Product Name: ",x[5])
     print("Product ID: ",x[6])
     print("Price Per Unit: ",x[7])
     print("Quantity: ",x[8])
     print("SGST: ",x[9])
     print("CGST: ",x[10])
     print("Total: ",x[11])
     t=t+x[11]
     print()
print("Grand Total: ",t)
print()

def search_dob():
  t=0
  i=input("Enter Date of Billing You Want To Search: ")
  mycursor=con.cursor()
  mycursor.execute("SELECT * FROM billing WHERE
  date='{}'".format(i))
  myresult = mycursor.fetchall()
  for x in myresult:
        print()
        print("Bill Number: ",x[0])
        print("Customer Name: ",x[3])
        print("Mobile Number: ",x[4])
        print("Date: ",x[2])
        print()
        break
  p="select serial_no from billing where date='{}'"
  .format(i)
  mycursor.execute(p)
  record=mycursor.fetchall()
  for i in record:
```

CODE

```
for j in i:
s=('select*from billing where serial_no="{}"'
.format(j))
cur=con.cursor()
cur.execute(s)
rec=cur.fetchall()
for x in rec:
      print("Serial Number: ",i[-1][-1])
      print("Product Name: ",x[5])
      print("Product ID: ",x[6])
      print("Price Per Unit: ",x[7])
      print("Quantity: ",x[8])
      print("SGST: ",x[9])
      print("CGST: ",x[10])
      print("Total: ",x[11])
      t=t+x[11]
      print()
print("Grand Total: ",t)
print()

def search_item():
  i=input("Enter Item Name You Want To Search: ")
  mycursor=con.cursor()
  mycursor.execute("SELECT * FROM billing WHERE
  Product_Name='{}'".format(i))
  myresult = mycursor.fetchall()
  for x in myresult:
      print()
      print("Bill Number: ",x[0])
      print("Customer Name: ",x[3])
      print("Mobile Number: ",x[4])
      print("Date: ",x[2])
      print("Quantity: ",x[8])
      print()
      print()
```

CODE

```
def search_data():
    while True:
        print("1) Search by Bill Number\n2)
        Search by Customer Name\n3) Search by Mobile
        Number\n4) Search by Date of Billing\n5)
        Search by Item\n6) Exit")
        print()
        ch=int(input("Enter your Choice: "))
        if ch==1:
            search_bill()
        elif ch==2:
            search_name()
        elif ch==3:
            search_mobile()
        elif ch==4:
            search_dob()
        elif ch==5:
            search_item()
        elif ch==6:
            break
        else:
            print()
            print("Invalid Choice … ")
            print()
            print("Try Again")
            print()

def print_bill(bill_no):
    import csv

    l='select DATE_FORMAT(date,"%d-%m-%y")from billing
    where bill_number={}'.format(bill_no)
    cur=con.cursor()
    cur.execute(l)
    rec=cur.fetchall()
    data=[["","","","","Zindagi Gulzar","","","",
    ["","","","","Invoice","","","",
    ["","2,Kakaji Marg","","","","","","Date",*rec[0]],
```

CODE

```
["","New Delhi,110032","","","","","","Invoice #",
bill_no],
["","Phone:","420420420","","","",],
[]]
filename = str(bill_no)+".csv"
with open(filename, 'w',newline="") as csvfile:
    csvwriter = csv.writer(csvfile)
    csvwriter.writerows(data)

p="select serial_no from billing where bill_number={}"
.format(bill_no)
cur.execute(p)
record=cur.fetchall()
for i in record:
     for j in i:
          s=('select*from billing where
          serial_no="{}"'.format(j))
          cur=con.cursor()
          cur.execute(s)
          rec=cur.fetchall()
          for r in rec:
               c_name=r[3]
               mob=r[4]
data=[["","Bill To:","","","",],
["","Customer Name:",c_name,"","",""],
["","Mobile No :",mob,"","",],
[],
["","S. No.","Product Name","Product Code",
"Price per unit","Sgst","Cgst","Qty","Total"]]
with open(filename, 'a',newline="") as csvfile:
       csvwriter = csv.writer(csvfile)
       csvwriter.writerows(data)

   t=0
   for i in record:
       for j in i:
           s=('select*from billing where
           serial_no="{}"'.format(j))
```

CODE

```
cur=con.cursor()
cur.execute(s)
rec=cur.fetchall()
for r in rec:
     s_no=i[-1][-1]
     p_name=r[5]
     p_code=r[6]
     ppu=r[7]
     qty=r[8]
     sgst=r[9]
     cgst=r[10]
     total=r[11]
     t=t+total
     data=["",s_no,p_name,p_code,ppu,sgst,cgst,qty,
     total]
     with open(filename, 'a',newline="") as csvfile:
            csvwriter = csv.writer(csvfile)
            csvwriter.writerow(data)

   data=[[],
         ["","Grand Total :-",'','','','','','',t,'',
         '','',],
         [],
         ['','',"Thank you for shopping !!",'','',]]
   with open(filename, 'a',newline="") as csvfile:
        csvwriter = csv.writer(csvfile)
        csvwriter.writerows(data)

def check_products():
    n=input("Enter Product Name: ")
    print()
    mycursor=con.cursor()
    mycursor.execute("Select*from products WHERE
    product_name='{}'".format(n))
    myresult = mycursor.fetchall()
    for x in myresult:
```

CODE

```
        print("Serial Number: ",x[0])
        print("Product Name: ",x[2])
        print("Product ID: ",x[1])
        print("Price Per Unit: ",x[3])
        print("SGST: ",x[4])
        print("CGST: ",x[5])
        print("Quantity Sold: ",x[6])
        print("Stock Remaining: ",x[7])
        print()
        return

def add_products():
    i=input("Enter Product ID: ")
    n=input("Enter Product Name: ")
    ppu=int(input("Enter Price Per Unit: "))
    t=float(input("Enter tax rate: "))
    gst=t*ppu
    sg=gst/2
    cg=gst/2
    st=int(input("Enter Stock: "))
    qty=0
    s="insert into products(product_id,product_name,
    price_per_unit,sgst,cgst,stock,quantity_sold)
    values('{}','{}',{},{},{},{},{})".format(i,n,ppu,
    sg,cg,st,qty)
    mycursor=con.cursor()
    mycursor.execute(s)
    con.commit()
    n="insert into due_taxes(product_id,
    gst_per_product) values('{}',{})".format(i,gst)
    mycursor.execute(n)
    con.commit()
    u="update due_taxes set Quantity_Sold=0,
    Gst_Collected=0,Status='Due WHERE product_id='{}'"
    .format(i)
    mycursor.execute(u)
    con.commit()
    print()
```

CODE

```
def update_products():
    i=input("Enter Product ID you want to Modify: ")
    d=input("Enter Updated Product ID: ")
    n=input("Enter Updated Product Name: ")
    ppu=int(input("Enter Updated Price Per Unit: "))
    t=float(input("Enter tax rate: "))
    q=int(input("Enter Updated Quantity Sold: "))
    gst=t*ppu
    y="update due_taxes set Product_id='{}',
    gst_per_product={},quantity_sold={} where
    product_id='{}'".format(d,gst,q,i)
    mycursor=con.cursor()
    mycursor.execute(y)
    con.commit()
    sg=gst/2
    cg=gst/2
    st=int(input("Enter Updated Stock: "))
    s="update products set product_id='{}',
    product_name='{}',price_per_unit={},sgst={},cgst={},
    quantity_sold={},stock={} WHERE product_id='{}'"
    .format(d,n,ppu,sg,cg,q,st,i)
    mycursor.execute(s)
    con.commit()
    print()

def check_stock():
    i=input("Enter Product ID you want to check Stock for:
    s="select*from products WHERE Product_ID='{}'".format(
    print()
    mycursor=con.cursor()
    mycursor.execute(s)
    myresult=mycursor.fetchall()
    for x in myresult:
        print("Serial Number: ",x[0])
        print("Product Name: ",x[2])
        print("Price Per Unit: ",x[3])
        print("SGST: ",x[4])
```

CODE

```
        print("CGST: ",x[5])
        print("Quantity Sold: ",x[6])
        print("Stock Remaining: ",x[7])
        print()
def update_stock():
    i=input("Enter Product ID you want to Modify: ")
    st=int(input("Enter Stock you want to Add: "))
    y="select stock from products where product_id='{}'
    ".format(i)
    mycursor=con.cursor()
    mycursor.execute(y)
    myresult=mycursor.fetchall()
    m=int(str(*myresult[0]))
    stock=st+m
    s="update products set stock={} WHERE
    product_id='{}'".format(stock,i)
    mycursor.execute(s)
    con.commit()
    print()

def update_mrp():
    i=input("Enter Product Code you want to Modify: ")
    m=int(input("Enter Updated MRP: "))
    t=float(input("Enter tax rate: "))
    gst=m*t
    y="update due_taxes set gst_per_product={} where
    product_id='{}'".format(gst,i)
    mycursor=con.cursor()
    mycursor.execute(y)
    con.commit()
    x=gst/2
    f="update products set price_per_unit={},sgst={},
    cgst={} where product_id='{}'".format(m,x,x,i)
    mycursor.execute(f)
    con.commit()
    print()
    print("MRP Updated")
    print()
```

CODE

```
def check_taxes():
  i="select product_id,quantity_sold,gst_collected,
  status from due_taxes"
  mycursor=con.cursor()
  mycursor.execute(i)
  myresult=mycursor.fetchall()
  for a in myresult:
     y="select product_name from products where
     product_id='{}'".format(a[0])
     mycursor.execute(y)
     myres=mycursor.fetchall()
     print("Product Name:- ",*myres[0])
     print("Quantity Sold:- ",a[1])
     print("GST_collected:- ",a[2])
     print("Status:- ",a[3])
     print()
  print()

def pay_taxes():
  total_gst=0
  i="select gst_collected from due_taxes"
  mycursor=con.cursor()
  mycursor.execute(i)
  myresult=mycursor.fetchall()
  for a in myresult:
      for x in a:
          total_gst=total_gst+x
  print("Total GST due:- ",total_gst)
  ch=input("Would you like to pay the taxes:- ")
  if ch=="yes" or "YES":
    t="update due_taxes set status='Paid',
    quantity_sold=0,gst_collected=0"
    mycursor.execute(t)
    con.commit()
  print()
```

CODE

```
def update_gst():
  s=input("Enter product code:- ")
  t=float(input("Enter updated tax rate:- "))
  y="select price_per_unit from products where
  product_ID='{}'".format(s)
  mycursor=con.cursor()
  mycursor.execute(y)
  myresult=mycursor.fetchone()
  x=int(*myresult)
  gst=t*x
  i="update due_taxes set gst_per_product={} where
  product_ID='{}'".format(gst,s)
  mycursor.execute(i)
  con.commit()
  e="select SGST,CGST from products WHERE
  product_ID='{}'".format(s)
  mycursor.execute(e)
  myres=mycursor.fetchone()
  for x in myres:
      x=gst/2
      o="update products set sgst={},cgst={} where
      product_ID='{}'".format(x,x,s)
      mycursor.execute(o)
      con.commit()
  print("\nGST Updated")
  print()

def bill_management():
    while True:
        print("1) Add Bill\n2) Search Bill\n3)
        Print Invoice\n4) Exit")
        ch=int(input("Enter Choice: "))
        print()
        if ch==1:
            add_data()
        elif ch==2:
            search_data()
```

```
        elif ch==3:
            bill_no=int(input("Enter Bill Number: "))
            print_bill(bill_no)
            print("Invoice Generated")
        elif ch==4:
            break
        else:
            print("Invalid Choice")
            print("Try Again")
            print()


def tax_management():
    while True:
        print("1) Check Taxes\n2) Pay Taxes\n3)
        Update GST\n4) Exit")
        ch=int(input("Enter Your Choice: "))
        if ch==1:
            check_taxes()
        elif ch==2:
            pay_taxes()
        elif ch==3:
            update_gst()
        elif ch==4:
            break
        else:
            print("Invalid Choice")
            print("Try Again")
            print()

def product_management():
    while True:
        print("1) Check Products\n2) Add Products\n3)
        Update Products\n4) Check Stock\n5)
        Update Stock\n6) Update MRP\n7) Exit")
        ch=int(input("Enter Choice: "))
        print()
```

CODE

```
        if ch==1:
            check_products()
        elif ch==2:
            add_products()
        elif ch==3:
            update_products()
        elif ch==4:
            check_stock()
        elif ch==5:
            update_stock()
        elif ch==6:
            update_mrp()
        elif ch==7:
            break
        else:
            print("Invalid Choice")
            print("Try Again")
            print()

#MAIN
while True:
    print()
    print("Welcome to Zindagi Gulzar!")
    print()
    print("What do you Want to Do!")
    print()
    print("1) Bill Management")
    print("2) Taxes Management")
    print("3) Product Management")
    print("4) Exit")
    print()
    ch=int(input("Enter Your Choice: "))
    print()
    if ch==1:
        bill_management()
    elif ch==2:
        print()
```

CODE

```
    tax_management()
elif ch==3:
    product_management()
elif ch==4:
    print("Sayonara!")
    break
else:
    print("Invalid Choice")
    print("Try Again")
    print()
```

OUTPUT

add data

```
Welcome to Zindagi Gulzar!

What do you Want to Do!

1) Bill Management
2) Taxes Management
3) Product Management
4) Exit

Enter Your Choice: 1

1) Add Bill
2) Search Bill
3) Print Invoice
4) Exit

Enter Choice: 1
```

```
Enter Customer Name: Sambav

Enter Date of Billing: 2022-03-03

Enter Mobile Number: 9999408982

Number of Items Purchased: 2

Enter Product Code: AP-M011

Enter Quantity: 1

Enter Product Code: SG-E010

Enter Quantity: 1

Invoice Generated
```

				Zindagi Gulzar				
				Invoice				
	2,Kakaji Marg						Date	03-03-22
	New Delhi,110032						Invoice #	12
	Phone:	420420420						
	Bill To:							
	Customer Name	Sambav						
	Mobile No :	9999408982						
	S. No.	Product Name	Product Code	Price per unit	Sgst	Cgst	Qty	Total
	1	Iphone 12	AP-M011	60000	6750	6750	1	73500
	2	Samsung Galaxy Buds	SG-E010	10000	600	600	1	11200
	Grand Total :-							84700
		Thank you for shopping !!						

OUTPUT

search by bill number

```
1) Add Bill
2) Search Bill
3) Print Invoice
4) Exit

Enter Choice: 2

1) Search by Bill Number
2) Search by Customer Name
3) Search by Mobile Number
4) Search by Date of Billing
5) Search by Item
6) Exit
```

```
Enter your Choice: 1
```

```
Enter Bill Number You Want To Search: 2

Bill Number:  2
Customer Name:  Kronos
Mobile Number:  9213640490
Date:  2020-03-16

Serial Number:  1
Product Name:  Oppo Enco W11
Product ID:  OO-E010
Price Per Unit:  2500
Quantity:  1
SGST:  150.0
CGST:  150.0
Total:  2800

Grand Total:  2800
Do you want to Print Invoice?

Enter Choice: no
```

OUTPUT

search by customer name

```
1) Search by Bill Number
2) Search by Customer Name
3) Search by Mobile Number
4) Search by Date of Billing
5) Search by Item
6) Exit


Enter your Choice: 2

Enter Customer Name You Want To Search: Sammy Boy

Bill Number:  4
Customer Name:  Sammy Boy
Mobile Number:  958430903
Date:  2022-01-01

Serial Number:  1
Product Name:  Iphone 13 Pro
Product ID:  AP-M013
Price Per Unit:  130000
Quantity:  5
SGST:  14625.0
CGST:  14625.0
Total:  796250

Serial Number:  2
Product Name:  Iphone 11
Product ID:  AP-M010
Price Per Unit:  49900
Quantity:  5
SGST:  5614.0
CGST:  5614.0
Total:  305640

Grand Total:  1101890
```

OUTPUT

search by mobile number

```
1) Search by Bill Number
2) Search by Customer Name
3) Search by Mobile Number
4) Search by Date of Billing
5) Search by Item
6) Exit


Enter your Choice: 3

Enter Mobile Number You Want To Search: 456780987

Bill Number:  8
Customer Name:  JJN
Mobile Number:  456780987
Date:  2022-02-07

Serial Number:  1
Product Name:  Oppo Enco W11
Product ID:  OO-E010
Price Per Unit:  2500
Quantity:  1
SGST:  150.0
CGST:  150.0
Total:  2800

Grand Total:  2800
```

OUTPUT

search by date of billing

```
1) Search by Bill Number
2) Search by Customer Name
3) Search by Mobile Number
4) Search by Date of Billing
5) Search by Item
6) Exit


Enter your Choice: 4

Enter Date of Billing You Want To Search: 2022-02-01

Bill Number:  7
Customer Name:  Yum
Mobile Number:  882992133
Date:  2022-02-01

Serial Number:  1
Product Name:  Apple I-Mac 24
Product ID:  AP-D001
Price Per Unit:  160000
Quantity:  3
SGST:  14400.0
CGST:  14400.0
Total:  566400

Serial Number:  2
Product Name:  Apple I-Mac 24
Product ID:  MS-M003
Price Per Unit:  160000
Quantity:  5
SGST:  14400.0
CGST:  14400.0
Total:  944000

Serial Number:  3
Product Name:  Iphone 11
Product ID:  AP-M010
Price Per Unit:  49900
Quantity:  5
SGST:  5614.0
CGST:  5614.0
Total:  305640

Grand Total:  1816040
```

OUTPUT

search by item

```
1) Search by Bill Number
2) Search by Customer Name
3) Search by Mobile Number
4) Search by Date of Billing
5) Search by Item
6) Exit


Enter your Choice: 5

Enter Item Name You Want To Search: Apple Macbook Air

Bill Number:  9
Customer Name:  shreyas
Mobile Number:  987654567
Date:  2022-02-16
Quantity:  1
```

OUTPUT

Print Invoice

```
1) Add Bill
2) Search Bill
3) Print Invoice
4) Exit

Enter Choice: 3
```

```
Enter Bill Number: 2
Invoice Generated
```

				Zindagi Gulzar				
				Invoice				
	2,Kakaji Marg						Date	16-03-20
	New Delhi,110032						Invoice #	2
	Phone:	420420420						
	Bill To:							
	Customer Name Kronos							
	Mobile No :	9213640490						
	S. No.	Product Name	Product Code	Price per unit	Sgst	Cgst	Qty	Total
	1	Oppo Enco W11	OO-E010	2500	150	150	1	2800
	Grand Total :-							2800
		Thank you for shopping !!						

OUTPUT

Check Taxes

```
Welcome to Zindagi Gulzar!

What do you Want to Do!

1) Bill Management
2) Taxes Management
3) Product Management
4) Exit


Enter Your Choice: 2


1) Check Taxes
2) Pay Taxes
3) Update GST
4) Exit

Enter Your Choice: 1
Product Name:-  Apple I-Mac 24
Quantity Sold:-  5
GST_collected:-  144000.0
Status:-  Due

Product Name:-  Apple Airpods Pro
Quantity Sold:-  0
GST_collected:-  0.0
Status:-  Paid

Product Name:-  Apple Macbook Air
Quantity Sold:-  1
GST_collected:-  19798.0
Status:-  Due

Product Name:-  Apple Macbook Pro
Quantity Sold:-  0
GST_collected:-  0.0
Status:-  Paid

Product Name:-  Iphone 11
Quantity Sold:-  5
GST_collected:-  56140.0
Status:-  Due

Product Name:-  Iphone 12
Quantity Sold:-  2
GST_collected:-  27000.0
Status:-  Due

Product Name:-  Iphone 13
Quantity Sold:-  0
GST_collected:-  0.0
Status:-  Paid

Product Name:-  Iphone 13 Pro
Quantity Sold:-  0
GST_collected:-  0.0
Status:-  Paid

Product Name:-  Asus Tuf Dash Gaming
Quantity Sold:-  6
GST_collected:-  76500.0
Status:-  Due

Product Name:-  Asus Zenbook Duo 14
Quantity Sold:-  0
GST_collected:-  0.0
Status:-  Paid
```

OUTPUT

Check Taxes

```
Product Name:-  HP 15S-DU
Quantity Sold:-  0
GST_collected:-  0.0
Status:-  Paid

Product Name:-  Lenovo IPG3-15
Quantity Sold:-  0
GST_collected:-  0.0
Status:-  Paid

Product Name:-  Lenovo Legion 5i
Quantity Sold:-  0
GST_collected:-  0.0
Status:-  Paid

Product Name:-  Mi Notebook 14
Quantity Sold:-  4
GST_collected:-  38760.0
Status:-  Due

Product Name:-  Mi 11 Lite
Quantity Sold:-  3
GST_collected:-  8850.0
Status:-  Due

Product Name:-  Redmi 10S
Quantity Sold:-  2
GST_collected:-  3400.0
Status:-  Due

Product Name:-  Microsoft Surface Laptop Go
Quantity Sold:-  0
GST_collected:-  0.0
Status:-  Paid

Product Name:-  Microsoft Surface Laptop 4
Quantity Sold:-  0
GST_collected:-  0.0
Status:-  Paid

Product Name:-  Microsoft Surface Laptop 3
Quantity Sold:-  0
GST_collected:-  0.0
Status:-  Paid

Product Name:-  Oppo Enco W11
Quantity Sold:-  1
GST_collected:-  300.0
Status:-  Due

Product Name:-  Oneplus Nord 2
Quantity Sold:-  5
GST_collected:-  16500.0
Status:-  Due

Product Name:-  Samsung Galaxy Buds
Quantity Sold:-  4
GST_collected:-  4800.0
Status:-  Due

Product Name:-  S22
Quantity Sold:-  0
GST_collected:-  0.0
Status:-  Paid

Product Name:-  S22+
Quantity Sold:-  0
GST_collected:-  0.0
Status:-  Paid
```

OUTPUT

Check Taxes

```
Product Name:-  Samsung Galaxy A72
Quantity Sold:-  0
GST_collected:-  0.0
Status:-  Paid

Product Name:-  Sony WH-1000XM4
Quantity Sold:-  1
GST_collected:-  3898.0
Status:-  Due

Product Name:-  Vivo Y73
Quantity Sold:-  1
GST_collected:-  2728.0
Status:-  Due
```

Pay Taxes

```
1) Check Taxes
2) Pay Taxes
3) Update GST
4) Exit

Enter Your Choice: 2
Total GST due:-  402674.0

Would you like to pay the taxes:- yes
```

OUTPUT

Pay Taxes

```
1) Check Taxes
2) Pay Taxes
3) Update GST
4) Exit

Enter Your Choice: 2
Total GST due:-  0.0
```

Update GST

```
1) Check Taxes
2) Pay Taxes
3) Update GST
4) Exit

Enter Your Choice: 3

Enter product code:- AP-D001

Enter updated tax rate:- .12

GST Updated
```

OUTPUT

Check Products

```
Welcome to Zindagi Gulzar!

What do you Want to Do!

1) Bill Management
2) Taxes Management
3) Product Management
4) Exit


Enter Your Choice: 3

1) Check Products
2) Add Products
3) Update Products
4) Check Stock
5) Update Stock
6) Update MRP
7) Exit

Enter Choice: 1


Enter Product Name: Oneplus Nord 2

Serial Number:  21
Product Name:  Oneplus Nord 2
Product ID:  OP-M010
Price Per Unit:  29999
SGST:  1650
CGST:  1650
Quantity Sold:  6
Stock Remaining:  894
```

OUTPUT

Add Products

```
1) Check Products
2) Add Products
3) Update Products
4) Check Stock
5) Update Stock
6) Update MRP
7) Exit

Enter Choice: 2


Enter Product ID: RM-M010

Enter Product Name: Realme GT2

Enter Price Per Unit: 34000

Enter tax rate: .14

Enter Stock: 500
```

```
Enter Product Name: Realme GT2

Serial Number:  28
Product Name:  Realme GT2
Product ID:  RM-M010
Price Per Unit:  34000
SGST:  2380
CGST:  2380
Quantity Sold:  0
Stock Remaining:  500
```

OUTPUT

Update Products

```
1) Check Products
2) Add Products
3) Update Products
4) Check Stock
5) Update Stock
6) Update MRP
7) Exit

Enter Choice: 3


Enter Product ID you want to Modify: AP-D001

Enter Updated Product ID: AP-D001

Enter Updated Product Name: Apple I-Mac 24

Enter Updated Price Per Unit: 160000

Enter tax rate: .13

Enter Updated Quantity Sold: 311

Enter Updated Stock: 189
```

OUTPUT

Check Stock

```
1) Check Products
2) Add Products
3) Update Products
4) Check Stock
5) Update Stock
6) Update MRP
7) Exit

Enter Choice: 4


Enter Product ID you want to check Stock for: OP-M010

Serial Number:  21
Product Name:  Oneplus Nord 2
Price Per Unit:  29999
SGST:  1650
CGST:  1650
Quantity Sold:  6
Stock Remaining:  894
```

OUTPUT

Update Stock

```
1) Check Products
2) Add Products
3) Update Products
4) Check Stock
5) Update Stock
6) Update MRP
7) Exit

Enter Choice: 5


Enter Product ID you want to Modify: AP-D001

Enter Stock you want to Add: 1000


Serial Number:  1
Product Name:  Apple I-Mac 24
Product ID:  AP-D001
Price Per Unit:  160000
SGST:  10400
CGST:  10400
Quantity Sold:  311
Stock Remaining:  1189
```

OUTPUT

Update Mrp

```
1) Check Products
2) Add Products
3) Update Products
4) Check Stock
5) Update Stock
6) Update MRP
7) Exit

Enter Choice: 6


Enter Product Code you want to Modify: OO-E010

Enter Updated MRP: 5000

Enter tax rate: .12

MRP Updated

Serial Number:  20
Product Name:  Oppo Enco W11
Product ID:  OO-E010
Price Per Unit:  5000
SGST:  300
CGST:  300
Quantity Sold:  1002
Stock Remaining:  1998
```

ACKNOWLEDGEMENT

I would like to extend my sincere and heartfelt gratitude to my Computer Science teacher, Mrs. R. Nagpal, who has helped me in this endeavor and has always been very cooperative, and without her help, cooperation, guidance and enc ouragement, I would not have been able to do this project.

BIBLIOGRAPHY

https://dev.mysql.com/doc/refman/8.0/en/example-auto-increment.html

https://dev.mysql.com/doc/connector-python/en/

https://docs.python.org/3/library/csv.html