COMPUTER SCIENCE PROJECT

CONSTUCTION COMPANY DATABASE MANAGEMENT SYSTEM

2021-22

PROJECT REPORT

Abin Raj

Sedulous..

Jewal Gigal

Jewal Gigal.

Table of Contents

Introduction

This program of ours, Construction Company Database Management System, CCDBMS, has its main utility at this time of the pandemic, where placing an order by going to a shop physically is both tedious and unsafe.
We present CCDBMS, which can act as an intermediate in cyberspace for providing a connection to both the vendor and customer. An admin controlled stock range and customer-oriented suggestive platform where a customer can suggest or request an order, which might either be unavailable due to lack of stock or hasn't been a part of the catalogue. Doing so can help the vendor in getting a bigger exposure to the customer wants and needs, at the same time providing the customers with the freedom of wants.

Objective

The problems of getting to reach the right audience for the vendor and reaching the right place for the customers are attempted to be solved in our project, CCDBMS. It used Tkinter in the frontend and MySQL as a backend.

This project provides the Customer with the following facilities :
- Purchase items
- Make payments
- Place orders
- Give projects

The Vendor gets the following facilities:
- Request for payments
- Sell items
- Complete Orders
- Take projects
- Complete projects

Project Resource Used

Softwares used

Pycharm

DataGrip

Azure Cloud

MySQL and Python

SQL is designed to query and extract data from tables within a database.
Python is particularly well suited for structured(tabular) data which can be fetched using SQL.

Tkinter

Tkinter is a Python library that helps in building simple GUI applications
To implement it: 'import tkinter as tk'

Hardware

- Keyboard and Optical mouse for input
- Processor: M1 and Intel i5
- Video display unit / External Monitor
- OS: macOS 11 and Windows 10

MySQL program

```
create table Admin
(`Admin id` varchar(4) not null
      primary key,
   Password   int       null,
   logs      int       null);
```

```
create table Admin
(`Admin id` varchar(4) not null
 primary key,
 Password int null,
 logs int null);
```

```
create table Customer
(Date date null,
 Name varchar(22) not null,
 Age int null,
 address varchar(50) not null
 primary key,
 Project varchar(15) null);
```

```
create table Item
(Quantity int null,
 ItemNumber varchar(5) not null
 primary key,
 Price int null,
 Orders varchar(10) null);
```

```
create table Item
(Quantity int null,
 ItemNumber varchar(5) not null
 primary key,
 Price int null,
 Orders varchar(10) null);
```

MySQL program

```sql
create table OrderTable
(Customer_name varchar(21) null,
 Items varchar(19) null,
 Address varchar(50) null,
 AMOUNT int null,
 Phone int not null
 primary key,
 Project varchar(20) null);
```

```sql
create table Payment
(Name varchar(50) not null,
 Phone bigint not null,
 Address varchar(50) null,
 Item varchar(50) null,
 primary key (Name, Phone));
```

```sql
create table Project
(ProjectNumber varchar(4) not null,
 CustomerName varchar(21) not null,
 CustomerNumber varchar(4) not null,
 VendorNumber varchar(25) null,
 Item varchar(20) null,
 Details int not null,
 primary key (CustomerNumber, ProjectNumber,
CustomerName),
 constraint Project_Cust_no_uindex
 unique (CustomerNumber));
```

```sql
create table Vendor
( Email varchar(25) not null,
 Name varchar(30) not null,
 Phone bigint not null,
 ShopName varchar(30) null,
 Address varchar(50) null,
 Items varchar(35) null,
 primary key (Email, Name, Phone),
 constraint Name
 unique (Name, Phone));
```

```sql
create table users
(Email varchar(50) not null
 primary key,
 Username varchar(25) null,
 Password varchar(25) null);
```

Login Screen

```
from tkinter import *
from PIL import ImageTk
from tkinter import messagebox
import mysql.connector as ms
import smtplib

class Login:
    def __init__(self, root):

        self.root = root
        self.root.title("Login and registration system for Apps")
        self.root.geometry("1366x700+0+0")  # sze of window
        self.root.resizable(False, False)  # to allow resizing swap False with True
        self.loginform()

    def loginform(self):  # Login Window
        # Frame for BG image
        Frame_login = Frame(self.root, bg='white')
        Frame_login.place(x=0, y=0, height=700, width=1366)  # Frame size same as size of Window

        self.img = ImageTk.PhotoImage(file="Dubai Skyline.jpg")
        img = Label(Frame_login, image=self.img)
        img.place(x=0, y=0, width=1366, height=700)  # img made as a label nd placed on the frame

        # Frame for arranging widgets like Labels, textboxes
        frame_input = Frame(self.root, bg='white')
        frame_input.place(x=320, y=130, height=450, width=700)

        # Arranging every element onto the frame_input
        label1 = Label(frame_input, text="Login Here", font=('Bernard MT Condensed', 32, 'bold'), fg="black",
                       bg='white')
        label1.place(x=240, y=20)

        Username_label = Label(frame_input, text="Username", font=("Goudy old style", 20, "bold"), fg='#40e0d0',
                               bg='white')
        Username_label.place(x=210, y=95)

        self.Username_txt = Entry(frame_input, font=("times new roman", 12), bg='lightgray')
        self.Username_txt.place(x=210, y=145, width=270, height=35)
        self.Username_txt.insert(0, 'Enter your Username or E-mail')

        Password_label = Label(frame_input, text="Password", font=("Goudy old style", 20, "bold"), fg='#40e0d0',
                               bg='white')
        Password_label.place(x=210, y=195)

        self.Password_txt = Entry(frame_input, font=("times new roman", 12), bg='lightgray')
        self.Password_txt.place(x=210, y=245, width=270, height=35)
        self.Password_txt.insert(0, 'Enter your Password')

        Forgot_pass_btn = Button(frame_input, text="Forgot Password?", cursor='hand2', command=self.forgot_pass,
                                 font=('calibri', 10), bg='white', fg='black', bd=0)
        Forgot_pass_btn.place(x=290, y=305)

        Login_btn = Button(frame_input, text="Login", command=self.login, cursor="hand2",
                           font=("times new roman", 15), fg="Black", bg="Black", bd=0, width=15, height=1)
        Login_btn.place(x=255, y=340)

        Register_btn = Button(frame_input, command=self.Register, text="Not Registered?Register", cursor="hand2",
                              font=("calibri", 10), bg='white', fg="black", bd=0)
        Register_btn.place(x=280, y=390)  # Register button for those not registered

    def login(self):
        if self.Username_txt.get() == "" or self.Password_txt.get() == "":
            messagebox.showerror("Error", "All fields are required", parent=self.root)

        else:  # SQL connection
            try:
                mydb = ms.connect(host='localhost', user='root', password='Sedulous', database='CCDBMS')

                cur = mydb.cursor()
                cur.execute("select * from users where username='{}' and password='{}'".
                            format(self.Username_txt.get(), self.Password_txt.get()))  # Need clarity
                if not mydb.is_connected():
                    print('Connection Failed  !!')
                row1 = cur.fetchone()
                if row1 == None:
                    # messagebox.showerror('Error', 'Invalid Username And Password', parent=self.root)
                    # self.login_clear()
                    # self.Username_txt.focus_set()
                    cur.execute("select * from users where email='{}' and password='{}'".
                                format(self.Username_txt.get(), self.Password_txt.get()))
                    row2 = cur.fetchone()

                    if row2 == None:
                        messagebox.showerror('Error', 'Invalid Username And Password', parent=self.root)

                        self.login_clear()
                        self.Username_txt.focus_set()
                    else:
                        messagebox.showinfo('Login Status', 'You Have Successfully Logged in', parent=self.root)
                else:
                    messagebox.showinfo('Login Status', 'You Have Successfully Logged in', parent=self.root)


            except:
                messagebox.showinfo('ERROR !!', 'An Unexpected Error has occurred', parent=self.root)

    def Register(self):  # Frame for first tym registrations .. Page design.. Comes here after clicking 'not registered' 'button

        Frame_login1 = Frame(self.root, bg="white")  # Frame for Back ground image
        Frame_login1.place(x=0, y=0, height=700, width=1366)

        self.img = ImageTk.PhotoImage(file="Dubai Skyline.jpg")  # loading img from local drive
        img = Label(Frame_login1, image=self.img)
        img.place(x=0, y=0, width=1366, height=700)

        frame_input = Frame(self.root, bg='White')  # and this Frame for register here, username password etc etc   labels and entry boxes
        frame_input.place(x=320, y=130, height=450, width=700)

        label1 = Label(frame_input, text="Register Here", font=('Bernard MT Condensed', 32, 'bold'), fg="#40e0d0",
                       bg='white')
        label1.place(x=35, y=20)

        # For username... label and textbox
        Username_label = Label(frame_input, text="Username", font=("Goudy old style", 20, "bold"), fg='#40e0d0',
                               bg='white')
        Username_label.place(x=30, y=95)

        self.Username_Entry = Entry(frame_input, font=("times new roman", 15), bg='lightgray')
        self.Username_Entry.place(x=30, y=145, width=270, height=35)
        self.Username_Entry.insert(0, 'Enter your Username')
```

```
# For Password... Label and textbox
   Password_label = Label(frame_input, text="Password", font=("Goudy old style", 20, "bold"), fg='#40e0d0',
   bg='white')
   Password_label.place(x=30, y=195)

   self.Password_Entry = Entry(frame_input, font=("times new roman", 15), bg='lightgray')
   self.Password_Entry.place(x=30, y=245, width=270, height=35)
   self.Password_Entry.insert(0, 'Enter your Password')

#  For Email ID ...
    Email_label = Label(frame_input, text="Email-id", font=("Goudy old style", 20, "bold"), fg='#40e0d0',
              bg='white')
    Email_label.place(x=360, y=95)

    self.Email_Entry = Entry(frame_input, font=("times new roman", 15), bg='lightgray')
    self.Email_Entry.place(x=360, y=145, width=270, height=35)
    self.Email_Entry.insert(0, 'Enter your E-mail ID')

    # Confirm Password... Label and Box
    Confirm_pass = Label(frame_input, text="Confirm Password", font=("Goudy old style", 20, "bold"), fg='#40e0d0',
               bg='white')
    Confirm_pass.place(x=360, y=195)

    self.Confirm_pass = Entry(frame_input, font=("times new roman", 15), bg='lightgray')
    self.Confirm_pass.place(x=360, y=245, width=270, height=35)
    self.Confirm_pass.insert(0, 'Re-enter your Password')

    # Register Button
    Register_btn = Button(frame_input, command=self.register, text="Register", cursor="hand2",
               font=("times new roman", 15), fg="#40e0d0", bg="#40e0d0", bd=0, width=15, height=1)

    Register_btn.place(x=240, y=340)

    # Already registered  button
    btn3 = Button(frame_input, command=self.loginform, text="Already Registered?Login", cursor="hand2",
           font=("calibri", 10), bg='white', fg="#40e0d0", bd=0)
    btn3.place(x=248, y=390)

  def register(self):  # Register Button function
    if self.Username_Entry.get() == "" or self.Password_Entry.get() == "" or self.Email_Entry.get() == "" or self.Confirm_pass.get() == "":
      messagebox.showerror("Error", "All Fields Are Required", parent=self.root)
    elif self.Password_Entry.get() != self.Confirm_pass.get():
      messagebox.showerror("Error", "Password and Confirm Password Should Be Same", parent=self.root)

    else:
      mydb = ms.connect(host="localhost", user="root", password="Sedulous", database="CCDBMS")
      cursor = mydb.cursor()

      cursor.execute("select * from users where email='{}'".format(self.Email_Entry.get()))
      row = cursor.fetchone()
      try:
        if row != None:
          messagebox.showerror("Error", "User already Exist,Please try with another Email", parent=self.root)

          self.reg_clear()
          self.Username_Entry.focus()

        else:
          cursor.execute("insert into users values('{}','{}','{}')".format(self.Email_Entry.get(),
                                        self.Username_Entry.get(),
                                        self.Password_Entry.get()))
          mydb.commit()
          mydb.close()

          messagebox.showinfo("Success", "Registration Successful", parent=self.root)

          self.reg_clear()
      except Exception as es:

          messagebox.showerror("Error", f"Error due to:{str(es)}", parent=self.root)

  def login_clear(self):
    self.Username_txt.delete(0, END)
    self.Password_txt.delete(0, END)

  def reg_clear(self):
    self.Username_Entry.delete(0, END)
    self.Password_Entry.delete(0, END)
    self.Confirm_pass.delete(0, END)
    self.Email_Entry.delete(0, END)

  def forgot_pass(self):
    Frame_bg = Frame(self.root, bg="white")  # guess its for Back ground image
    Frame_bg.place(x=0, y=0, height=700, width=1366)

    self.img = ImageTk.PhotoImage(file="Dubai Skyline.jpg")  # loading img from local drive
    img = Label(Frame_bg, image=self.img)
    img.place(x=0, y=0, width=1366, height=700)

    # Frame for arranging widgets like Labels, textboxes
    frame_input = Frame(self.root, bg='white')
    frame_input.place(x=320, y=130, height=450, width=700)

    # Arranging every element onto the frame

    label1 = Label(frame_input, text="Trouble Login in?", font=('Bernard MT Condensed', 28, 'bold'), fg="black",
           bg='White')
    label1.place(x=330, y=45, anchor='center')  # trouble login label

    Username_label = Label(frame_input, text="Username", font=("Goudy old style", 20, "bold"), fg='#40e0d0',
              bg='White')
    Username_label.place(x=50, y=110)
```

```
        self.Username_Entry = Entry(frame_input, font=("times new roman", 12), bg='lightgray')
        self.Username_Entry.place(x=260, y=110, width=270, height=35)
        self.Username_Entry.insert(0, "Enter your Username")

        Reg_email_label = Label(frame_input, text="Registered Mail", font=("Goudy old style", 20, "bold"), fg='#40e0d0',
                      bg='white')
        Reg_email_label.place(x=50, y=170)

        self.Reg_email_entry = Entry(frame_input, font=("times new roman", 12), bg='lightgray')
        self.Reg_email_entry.place(x=260, y=170, width=270, height=35)
        self.Reg_email_entry.insert(0, "Enter your Registered Email Address")

        send_mail_btn = Button(frame_input, text="Send Mail", command=self.send_mail, cursor="hand2",
                      font=("times new roman", 15), fg="#40e0d0", bg="#40e0d0", bd=0, width=15, height=1)
        send_mail_btn.place(x=80, y=270)
        # Back to Login form
        Back_btn = Button(frame_input, text="Back", command=self.loginform, cursor="hand2",
                    font=("times new roman", 15), fg="#40e0d0", bg="#40e0d0", bd=0, width=15, height=1)
        Back_btn.place(x=350, y=270)

        Label_note = Label(frame_input,
                     text='An E-mail with the Login Credentials will be sent to the Registered Email, after verification',
                     font=('Kozuka Gothic Pr6N M', 12, 'bold'), bg='white', fg='#40e0d0')
        Label_note.place(x=10, y=320)
    def send_mail(self):

        if self.Username_Entry.get()=='' or self.Reg_email_entry.get()=='':
            messagebox.showerror("Error", "All Fields Are Required", parent=self.root)
        else:
            mydb = ms.connect(host="localhost", user="root", password="Sedulous", database="CCDBMS")
            cursor = mydb.cursor()

            cursor.execute("select * from users where email='{}' and
username='{}'".format(self.Reg_email_entry.get(),self.Username_Entry.get()))
            row = cursor.fetchone()
            if row == None:
                messagebox.showerror("User Doesn't Exist", "Please Register or contact IT support for extended assistance",
parent=self.root)
            else:


                gmail_user = 'con.man.jih@gmail.com'
                gmail_password = '1234@conmanjih'

                sent_from = gmail_user
                to = row[0]

                forgot_password=row[2]
                subject = 'Forgot Your Password. Find your Credentials Here.'
                body = """
                    Dear User,
                        Here are login credentials
                        Username : %s
                        Password : %s"""%(self.Username_Entry.get(), forgot_password)

                email_text = """\
                    From: %s
                    To: %s
                    Subject: %s

                    %s
                    """ % (sent_from, to, subject, body)

                try:
                    smtp_server = smtplib.SMTP_SSL('smtp.gmail.com', 465)
                    smtp_server.ehlo()
                    smtp_server.login(gmail_user, gmail_password)
                    smtp_server.sendmail(sent_from, to, email_text)
                    smtp_server.close()
                    print("Email sent successfully!")
                except Exception as ex:
                    print("Something went wrong....", ex)
    def appscreen(self):  # after login
        pass


root = Tk()
object = Login(root)

root.mainloop()
```

Python program

Main Program

```
import mysql.connector as ms
import time

mydb = ms.connect(host='localhost',
             user='root',
             password='Sedulous')
if not mydb.is_connected():
   print('Could not connect to database Assignment')
   print('Exiting ......')
   time.sleep(3)
   exit()
else:
   mycursor = mydb.cursor()

import mysql.connector
mydb = mysql.connector.connect(
   host = "localhost",
   user = "root",
   password = "Sedulous",
   database = "CCDBMS" )
mycursor = mydb.cursor()

def NeedItem() :
   global tItem, tQuantity, tOT
   myCursor.execute("SELECT * FROM Item")
   my_wo = tkinter.Tk()
   my_wo.title("Available Items")
   my_wo.geometry("1366x700+0+0")
   i=0
   for Item in myCursor:
      for j in range(len(Item)):
         e = Entry(my_wo, width=10, fg='blue')
         e.grid(row=i, column=j)
         e.insert(END, Item[j])
      i+=1
   my_w = tkinter.Tk()
   my_w.geometry("1366x700+0+0")
   my_w.title("Buy Items")
   l0 = tkinter.Label(my_w,  text='Needs Items',font=('', 16), width=30,anchor="c" )
   l0.grid(row=1,column=1,columnspan=4)
   l1 = tkinter.Label(my_w,  text='Item Number : ', width=10,anchor="c" )
   l1.grid(row=3,column=1)
   tItem = tkinter.Text(my_w,  height=1, width=10,bg='white')
   tItem.grid(row=3,column=2)
   l2 = tkinter.Label(my_w,  text='Quantity : ', width=10,anchor="c" )
   l2.grid(row=4,column=1)
   tQuantity = tkinter.Text(my_w,  height=1, width=10,bg='white')
   tQuantity.grid(row=4,column=2)
   l3 = tkinter.Label(my_w,  text='Order : ', width=10,anchor="c")
   l3.grid(row=5,column=1)
   tOT = tkinter.Text(my_w,  height=1, width=10,bg='white')
   tOT.grid(row=5,column=2)

   b1 = tkinter.Button(my_w,  text='Buy', width=10,command=lambda: delete_data_item())
   b1.grid(row=7,column=2)

def delete_data_item() :
   my_name = tItem.get("1.0",END)
   query="DELETE FROM `Item` WHERE ItemNumber = %s"
   myCursor.execute(query,(my_name,))
   db_connection.commit()
   tItem.delete('1.0',END)
   tQuantity.delete('1.0',END)
   tOT.delete('1.0',END)
   print("Query executed")

def SellsItems() :
   global t1, t2, t3, t4
   my_w = tkinter.Tk()
   my_w.geometry("1366x700+0+0")
   my_w.title("Needs Items")
   l0 = tkinter.Label(my_w,  text='Needs Items',font=('Helvetica', 16), width=30,anchor="c" )
   l0.grid(row=1,column=1,columnspan=4)
   l1 = tkinter.Label(my_w,  text='Item Number : ',width=10,anchor="c" )
   l1.grid(row=3,column=1)
   t1 = tkinter.Text(my_w,  height=1, width=10,bg='white')
   t1.grid(row=3,column=2)
   l2 = tkinter.Label(my_w,  text='Quantity : ',width=10,anchor="c" )
   l2.grid(row=4,column=1)
   t2 = tkinter.Text(my_w,  height=1, width=10,bg='white')
   t2.grid(row=4,column=2)
   l3 = tkinter.Label(my_w,  text='Price : ', width=10,anchor="c")
   l3.grid(row=5,column=1)
   t3 = tkinter.Text(my_w,  height=1, width=10,bg='white')
   t3.grid(row=5,column=2)
   l4 = tkinter.Label(my_w,  text='Order : ', width=10,anchor="c")
   l4.grid(row=6,column=1)
   t4 = tkinter.Text(my_w,  height=1, width=10,bg='white')
   t4.grid(row=6,column=2)
   b1 = tkinter.Button(my_w,  text='Put the item on sale',width=15, command=lambda: add_data_item())
   b1.grid(row=7,column=2)

def add_data_item() :
   my_name = t1.get("1.0",END)
   my_class = t2.get("1.0",END)
   my_mark = t3.get("1.0",END)
   my_gender = t4.get("1.0",END)
   query="INSERT INTO  `Item`(`ItemNumber` ,`Quantity` ,`Price` ,`OrderItem`) VALUES(%s,%s,%s,%s)"
   my_data=(my_name,my_class,my_mark,my_gender)
   myCursor.execute(query,my_data)
```

```
    db_connection.commit()
    t1.delete('1.0',END)
    t2.delete('1.0',END)
    t3.delete('1.0',END)
    t4.delete('1.0',END)
    print("Query executed")

def MakePayment() :
    global tNamePay, tPhonePay, tAddressPay, tItemPay
    myCursor.execute("SELECT * FROM Payment")
    my_wo = tkinter.Tk()
    my_wo.title("Requested payments ")
    my_wo.geometry("1366x700+0+0")
    i=0
    for Item in myCursor:
        for j in range(len(Item)):
            e = Entry(my_wo, width=10, fg='blue')
            e.grid(row=i, column=j)
            e.insert(END, Item[j])
        i=i+1
    my_w = tkinter.Tk()
    my_w.geometry("1366x700+0+0")
    my_w.title("Pay")
    l0 = tkinter.Label(my_w,  text='Make Payment',font=('Helvetica', 16), width=30,anchor="c" )
    l0.grid(row=1,column=1,columnspan=4)
    l1 = tkinter.Label(my_w,  text='Name : ',width=10,anchor="c" )
    l1.grid(row=3,column=1)
    tNamePay = tkinter.Text(my_w,  height=1, width=10,bg='white')
    tNamePay.grid(row=3,column=2)
    l2 = tkinter.Label(my_w,  text='Phone : ', width=10,anchor="c")
    l2.grid(row=4,column=1)
    tPhonePay = tkinter.Text(my_w,  height=1, width=10,bg='white')
    tPhonePay.grid(row=4,column=2)
    l3 = tkinter.Label(my_w,  text='Address : ',width=10,anchor="c" )
    l3.grid(row=5,column=1)
    tAddressPay = tkinter.Text(my_w,  height=1,width=10,bg='white')
    tAddressPay.grid(row=5,column=2)
    l3 = tkinter.Label(my_w,  text='Item : ',width=10,anchor="c" )
    l3.grid(row=5,column=1)
    tItemPay = tkinter.Text(my_w,  height=1, width=10,bg='white')
    tItemPay.grid(row=5,column=2)
    b1 = tkinter.Button(my_w,  text='Pay', width=10,command=lambda: pay())
    b1.grid(row=7,column=2)
    print('Make payment')
def pay() :
    my_name = tNamePay.get("1.0",END)
    my_phone = tPhonePay.get("1.0",END)
    query="DELETE FROM 'Payment' WHERE Name = %s AND Phone = %s"
    myCursor.execute(query,(my_name, my_phone,))
    db_connection.commit()
    tNamePay.delete('1.0',END)
    tPhonePay.delete('1.0',END)
    tAddressPay.delete('1.0',END)
    tItemPay.delete('1.0',END)
    print("Query executed")

def TakePayments() :
    global tNameReq, tPhoneReq, tAddressReq, tItemReq
    my_w = tkinter.Tk()
    my_w.geometry("1366x700+0+0")
    my_w.title("Request Payment")
    l0 = tkinter.Label(my_w,  text='Request Payments',font=('Helvetica', 16), width=30,anchor="c" )
    l0.grid(row=1,column=1,columnspan=4)
    l1 = tkinter.Label(my_w,  text='Customer Name : ',width=45,anchor="c" )
    l1.grid(row=3,column=1)
    tNameReq = tkinter.Text(my_w,  height=1, width=45,bg='white')
    tNameReq.grid(row=3,column=2)
    l2 = tkinter.Label(my_w,  text='Customer Phone: ',width=45,anchor="c" )
    l2.grid(row=4,column=1)
    tPhoneReq = tkinter.Text(my_w,  height=1, width=45,bg='white')
    tPhoneReq.grid(row=4,column=2)
    l3 = tkinter.Label(my_w,  text='Customer Address : ',width=45,anchor="c" )
    l3.grid(row=5,column=1)
    tAddressReq = tkinter.Text(my_w,  height=1,width=45,bg='white')
    tAddressReq.grid(row=5,column=2)
    l4 = tkinter.Label(my_w,  text='Ordered Item : ',width=45,anchor="c" )
    l4.grid(row=6,column=1)
    tItemReq = tkinter.Text(my_w,  height=1, width=40,bg='white')
    tItemReq.grid(row=6,column=2)

    b1 = tkinter.Button(my_w,  text='Put on sale',width=40, command=lambda: requestPayment())
    b1.grid(row=7,column=2)

    b2 = tkinter.Button(my_w, text='Complete Order', width=40, command=lambda: comp_the_order())
    b2.grid(row=7, column=2)

def requestPayment() :
    my_name = tNameReq.get("1.0",END)
    my_class = tPhoneReq.get("1.0",END)
    my_mark = tAddressReq.get("1.0",END)
    my_gender = tItemReq.get("1.0",END)
    query="INSERT INTO  'Payment'(`Name`,`Phone`,`Address`,`Item`) VALUES (%s,%s,%s,%s)"
    my_data=(my_name,my_class,my_mark,my_gender)
    myCursor.execute(query,my_data)
    db_connection.commit()
    tNameReq.delete('1.0',END)
    tPhoneReq.delete('1.0',END)
    tAddressReq.delete('1.0',END)
    tItemReq.delete('1.0',END)
    print("Query executed")
```

Python program

```
def PlaceOrder() :
   global OrderCname, OrderPhone, OrderAddress, OrderAmount,OrderProject, OrderItem
   my_w = tkinter.Tk()
   my_w.geometry("1366x700+0+0")
   my_w.title("Place Order")

   l0 = tkinter.Label(my_w,  text='Place Order',font=('Helvetica', 16), width=30,anchor="c" )
   l0.grid(row=1,column=1,columnspan=4)

   l1 = tkinter.Label(my_w,  text='Customer Name : ',width=10,anchor="c" )
   l1.grid(row=3,column=1)

   OrderCname = tkinter.Text(my_w,  height=1,width=10,bg='white')
   OrderCname.grid(row=3,column=2)
   l2 = tkinter.Label(my_w,  text='Phone : ', width=10,anchor="c")
   l2.grid(row=4,column=1)
   OrderPhone = tkinter.Text(my_w,  height=1,width=10,bg='white')
   OrderPhone.grid(row=4,column=2)
   l3 = tkinter.Label(my_w,  text='Address : ',width=10,anchor="c" )
   l3.grid(row=5,column=1)
   OrderAddress = tkinter.Text(my_w,  height=1,width=10,bg='white')
   OrderAddress.grid(row=5,column=2)
   l4 = tkinter.Label(my_w,  text='Amount : ',width=10,anchor="c" )
   l4.grid(row=6,column=1)
   OrderAmount = tkinter.Text(my_w,  height=1,width=10,bg='white')
   OrderAmount.grid(row=6,column=2)
   l5 = tkinter.Label(my_w,  text='Project : ',width=10,anchor="c" )
   l5.grid(row=7,column=1)
   OrderProject = tkinter.Text(my_w,  height=1,width=10,bg='white')
   OrderProject.grid(row=7,column=2)
   l6 = tkinter.Label(my_w,  text='Item : ',width=10,anchor="c" )
   l6.grid(row=8,column=1)
   OrderItem = tkinter.Text(my_w,  height=1, width=10,bg='white')
   OrderItem.grid(row=8,column=2)
   b1 = tkinter.Button(my_w,  text='Place Order', width=10,command=lambda: give_order())
   b1.grid(row=10,column=2)
   print('Place order')

def give_order() :
   Cname = OrderCname.get('1.0',END)
   Phone = OrderPhone.get('1.0',END)
   Address = OrderAddress.get('1.0',END)
   Amount = OrderAmount.get('1.0',END)
   Project = OrderProject.get('1.0',END)
   Item = OrderItem.get('1.0',END)
   query = 'INSERT INTO OrderTable VALUES (%s,%s,%s,%s,%s,%s)'
   myData = (Cname, Phone, Address, Amount, Project, Item)
   myCursor.execute(query, myData)
   db_connection.commit()
   OrderCname.delete('1.0', END)
   OrderPhone.delete('1.0', END)
   OrderAddress.delete('1.0', END)
   OrderAmount.delete('1.0', END)
   OrderProject.delete('1.0', END)
   OrderItem.delete('1.0', END)
   print('Give order')

def CompleteOrder() :
   global CompCustName, CompCustPhone
   myCursor.execute("SELECT * FROM OrderTable")
   my_wo = tkinter.Tk()
   my_wo.title("Requested orders ")
   my_wo.geometry("1366x700+0+0")
   i=0
   for Project in myCursor:
      for j in range(len(Project)):
         e = Entry(my_wo, width=10, fg='blue')
         e.grid(row=i, column=j)
         e.insert(END, Project[j])
      i=i+1
   print('Complete order')
   my_w = tkinter.Tk()
   my_w.geometry("1366x700+0+0")
   my_w.title("Complete Order")
   l0 = tkinter.Label(my_w,  text='Complete Order',font=('Helvetica', 16), width=30,anchor="c" )
   l0.grid(row=1,column=1,columnspan=4)
   l1 = tkinter.Label(my_w,  text='Customer Name : ',width=10,anchor="c" )
   l1.grid(row=3,column=1)
   CompCustName = tkinter.Text(my_w,  height=1,width=10,bg='white')
   CompCustName.grid(row=3,column=2)
   l2 = tkinter.Label(my_w,  text='Phone : ', width=10,anchor="c")
   l2.grid(row=4,column=1)
   CompCustPhone = tkinter.Text(my_w,  height=1,width=10,bg='white')
   CompCustPhone.grid(row=4,column=2)
   b1 = tkinter.Button(my_w,  text='Complete Order', width=40,command=lambda: comp_the_order())
   b1.grid(row=7,column=2)
def comp_the_order() :
   CustName = CompCustName.get('1.0', END)
   CustPhone = CompCustPhone.get('1.0', END)
   query = "DELETE FROM OrderTable WHERE Phone = %s"
   myData = (CustPhone,)
   myCursor.execute(query, myData)
   db_connection.commit()
   CompCustName.delete('1.0', END)
   CompCustPhone.delete('1.0', END)
```

Python program

```
def GiveProject() :
   global tGiveProj, tGiveName, tGiveNum, tGiveVend, tGiveItem,tGiveDetails
   my_w = tkinter.Tk()
   my_w.geometry("1366x700+0+0")
   my_w.title("Give Project")

   l0 = tkinter.Label(my_w,  text='Give Project',font=('Helvetica', 16), width=50,anchor="c" )
   l0.grid(row=1,column=1,columnspan=4)

   l1 = tkinter.Label(my_w,  text='Project Number : ',width=40,anchor="c" )
   l1.grid(row=3,column=1)

   tGiveProj = tkinter.Text(my_w,  height=1, width=40,bg='white')
   tGiveProj.grid(row=3,column=2)

   l2 = tkinter.Label(my_w,  text='Customer Name : ',width=40,anchor="c" )
   l2.grid(row=4,column=1)

   tGiveName = tkinter.Text(my_w,  height=1, width=10,bg='white')
   tGiveName.grid(row=4,column=2)

   l3 = tkinter.Label(my_w,  text=' Customer Number : ',width=40,anchor="c" )
   l3.grid(row=5,column=1)

   tGiveNum = tkinter.Text(my_w,  height=1, width=10,bg='white')
   tGiveNum.grid(row=5,column=2)

   l4 = tkinter.Label(my_w,  text=' Vendor Number : ',width=40,anchor="c" )
   l4.grid(row=6,column=1)

   tGiveVend = tkinter.Text(my_w,  height=1, width=10,bg='white')
   tGiveVend.grid(row=6,column=2)

   l5 = tkinter.Label(my_w,  text='Item : ',width=10,anchor="c" )
   l5.grid(row=7,column=1)

   tGiveItem = tkinter.Text(my_w,  height=1, width=10,bg='white')
   tGiveItem.grid(row=7,column=2)

   l6 = tkinter.Label(my_w,  text='Details : ',width=10,anchor="c" )
   l6.grid(row=8,column=1)

   tGiveDetails = tkinter.Text(my_w,  height=1,width=10,bg='white')
   tGiveDetails.grid(row=8,column=2)

   b1 = tkinter.Button(my_w,  text='Give Project', width=10,command=lambda: giveTheProject())
   b1.grid(row=10,column=2)

   print('Give project')
def giveTheProject() :
   my_name = tGiveProj.get("1.0",END)
   my_class = tGiveName.get("1.0",END)
   my_mark = tGiveNum.get("1.0",END)
   my_gender = tGiveVend.get("1.0",END)
   my_item = tGiveItem.get("1.0",END)
   my_details = tGiveDetails.get("1.0",END)
   query=("INSERT INTO  `Project` (`ProjectNumber` ,`CustomerName`,`CustomerNumber` ,`VendorNumber`, `Item`, `Details`) VALUES (%s,%s,%s,%s,%s,%s)")
   my_data=(my_name,my_class,my_mark,my_gender, my_item,my_details)
   myCursor.execute(query,my_data)
   db_connection.commit()
   tGiveProj.delete('1.0',END)
   tGiveName.delete('1.0',END)
   tGiveNum.delete('1.0',END)
   tGiveVend.delete('1.0',END)
   tGiveItem.delete('1.0',END)
   tGiveDetails.delete('1.0',END)
   print("Query executed")

def TakesProjects() :
   global tTakeProj, tTakeName, tTakeDetails
   myCursor.execute("SELECT * FROM Project")
   my_wo = tkinter.Tk()
   my_wo.title("Requested projects ")
   my_wo.geometry("1366x700+0+0")
   i=0
   for Project in myCursor:
      for j in range(len(Project)):
         e = Entry(my_wo, width=10, fg='blue')
         e.grid(row=i, column=j)
         e.insert(END, Project[j])
      i=i+1
   my_w = tkinter.Tk()
   my_w.geometry("1366x700+0+0")
   my_w.title("Take Project")
   l0 = tkinter.Label(my_w,  text='Take Project',font=('Helvetica', 16), width=30,anchor="c" )
   l0.grid(row=1,column=1,columnspan=4)
   l1 = tkinter.Label(my_w,  text='Project Number : ',width=10,anchor="c" )
   l1.grid(row=3,column=1)
   tTakeProj = tkinter.Text(my_w,  height=1, width=10,bg='white')
   tTakeProj.grid(row=3,column=2)
   l2 = tkinter.Label(my_w,  text='Vendor Name : ',width=10,anchor="c" )
   l2.grid(row=4,column=1)
   tTakeName = tkinter.Text(my_w,  height=1, width=10,bg='white')
   tTakeName.grid(row=4,column=2)
   l3 = tkinter.Label(my_w,  text='Give details : ',width=10,anchor="c" )
   l3.grid(row=4,column=1)
   tTakeDetails = tkinter.Text(my_w,  height=1,width=10,bg='white')
   tTakeDetails.grid(row=4,column=2)
   b1 = tkinter.Button(my_w,  text='Take the project', width=25,command=lambda: taketheProj())
   b1.grid(row=6,column=2)
```

Python program

```
def taketheProj() :
   my_class = tTakeProj.get("1.0",END)
   my_mark = tTakeDetails.get("1.0",END)
   query = """UPDATE Project SET Details = %s WHERE ProjectNumber= %s"""
   my_data = (my_mark,my_class,)
   myCursor.execute(query,my_data)
   db_connection.commit()
   tTakeName.delete('1.0',END)
   tTakeDetails.delete('1.0',END)
   tTakeProj.delete('1.0',END)
   print("Query executed")
def CompletesProjects() :
   global tCompProj, tCompName, tCompDetails
   myCursor.execute("SELECT * FROM Project")
   my_wo = tkinter.Tk()
   my_wo.title("Requested projects ")
   my_wo.geometry("1366x700+0+0")
   i=0
   for Project in myCursor:
      for j in range(len(Project)):
         e = Entry(my_wo, width=10, fg='blue')
         e.grid(row=i, column=j)
         e.insert(END, Project[j])
      i=i+1
   my_w = tkinter.Tk()
   my_w.geometry("1366x700+0+0")
   my_w.title("Take Project")
   l0 = tkinter.Label(my_w,  text='Project Completed!',font=('Helvetica', 16), width=30,anchor="c" )
   l0.grid(row=1,column=1,columnspan=4)
   l1 = tkinter.Label(my_w,  text='Project Number : ',width=30,anchor="c" )
   l1.grid(row=3,column=1)
   tCompProj = tkinter.Text(my_w,  height=1, width=30,bg='white')
   tCompProj.grid(row=3,column=2)
   l2 = tkinter.Label(my_w,  text='Vendor Name : ',width=30,anchor="c" )
   l2.grid(row=4,column=1)
   tCompName = tkinter.Text(my_w,  height=1, width=30,bg='white')
   tCompName.grid(row=4,column=2)
   l3 = tkinter.Label(my_w,  text='Give details : ',width=30,anchor="c" )
   l3.grid(row=4,column=1)
   tCompDetails = tkinter.Text(my_w,  height=1,width=30,bg='white')
   tCompDetails.grid(row=4,column=2)
   b1 = tkinter.Button(my_w,  text='Completed', width=30,command=lambda: compTheProj())
   b1.grid(row=6,column=2)
def compTheProj() :
   my_class = tCompProj.get("1.0",END)
   query = """DELETE FROM Project WHERE ProjectNumber = %s"""
   my_data = (my_class,)
   myCursor.execute(query,my_data)
   db_connection.commit()
   tCompName.delete('1.0',END)
   tCompDetails.delete('1.0',END)
   tCompProj.delete('1.0',END)
   print("Query executed")
def HelloCallBack() :
   C = tkinter.Button(main, text = "Customer", command = CustomerDial, compound = "c",height=10,width=50)
   V = tkinter.Button(main, text = "Vendor", command = VendorDial, compound = "c",height=10,width=50)
   C.pack(padx=20, pady=20)
   V.pack(padx=20, pady=20)
HelloCallBack()
main.mainloop()
```

Python was the language of choice for the frontend as the project creates a desktop application and Python's library Tkinter eases the creation of all the kinds of UI needed for this project. The mysql connector for Python provides with seamless connection to the
database and also fast retrieval times. The cursor created with the help of the connector executes all the queries and also passes the data. The data is taken as input from the user using Tkinter TextBoxes and the get function. The data is then stored into variables and passed into the queries as arguments. The commit function of the mysql connector makes real changes to the connected database. Before running the software, the local mysql server needs to be started. The project has the following modules :

- Database creation • Table creation
- Core code
- Items : Needs{Consumer} and Sells{Vendor}
- Payments : Makes{Consumer} and Takes{Vendor}
- Orders : Places{Consumer} and Completes{Vendor}
- Projects : Gives{Consumer}, Takes{Vendor} and Completes{Vendor}

Python

program

https://github.com/Sedulous-logorrhea/Construction_Company_Management_System