-- phpMyAdmin SQL Dump

-- version 5.0.2

-- https://www.phpmyadmin.net/

--

-- Host: 127.0.0.1

-- Generation Time: Jun 30, 2022 at 11:09 AM

-- Server version: 10.4.14-MariaDB

-- PHP Version: 7.4.10

SET SQL\_MODE = "NO\_AUTO\_VALUE\_ON\_ZERO";

START TRANSACTION;

SET time\_zone = "+00:00";

/\*!40101 SET @OLD\_CHARACTER\_SET\_CLIENT=@@CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET @OLD\_CHARACTER\_SET\_RESULTS=@@CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET @OLD\_COLLATION\_CONNECTION=@@COLLATION\_CONNECTION \*/;

/\*!40101 SET NAMES utf8mb4 \*/;

--

-- Database: `finals\_db`

--

-- --------------------------------------------------------

--

-- Table structure for table `inventory`

--

CREATE TABLE `inventory` (

  `id` int(11) NOT NULL,

  `product` varchar(255) NOT NULL,

  `price` double NOT NULL,

  `quantity` int(11) NOT NULL,

  `description` varchar(255) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--

-- Dumping data for table `inventory`

--

INSERT INTO `inventory` (`id`, `product`, `price`, `quantity`, `description`) VALUES

(2, 'Scuttled', 5, 19, 'Soap'),

(3, 'Tang', 5, 13, 'Powdered Juice'),

(4, 'Lawcourt', 5, 8, 'Law and service'),

(6, 'adobo', 15, 20, 'Food'),

(7, 'nilaga', 5, 5, 'Ulam'),

(10, 'Tang orange', 5, 5, 'Juice'),

(12, 'Pepsi Black', 15, 20, 'GASOLINA'),

(18, 'Pepsi Purple', 12, 5, 'Kerosene');

-- --------------------------------------------------------

--

-- Table structure for table `invoice`

--

CREATE TABLE `invoice` (

  `id` int(11) NOT NULL,

  `name` varchar(255) NOT NULL,

  `mail` varchar(255) NOT NULL,

  `address` varchar(255) NOT NULL,

  `total` double NOT NULL,

  `date` datetime NOT NULL DEFAULT current\_timestamp()

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--

-- Dumping data for table `invoice`

--

INSERT INTO `invoice` (`id`, `name`, `mail`, `address`, `total`, `date`) VALUES

(6, 'Jake Brian Yap', 'yapjakebrian1@gmail.com', 'Lawaan', 87, '2022-06-30 13:25:25'),

(9, 'Bruh', 'bruh@gmail.com', 'bruh city', 12, '2022-06-30 14:02:28'),

(12, 'Johnny', 'JOhnny@gmail.com', 'johncity', 25, '2022-06-30 14:05:18'),

(14, 'Johhny Joestar', 'joestar@joemail.com', 'Lawaan', 75, '2022-06-30 14:17:20'),

(15, 'Jolyne Joestar', 'jolyne@gmail.com', 'joestar', 20, '2022-06-30 14:19:27'),

(69, 'Johnny Boy', 'john@gmail.com', 'john city', 12, '2022-06-30 16:36:16');

--

-- Indexes for dumped tables

--

--

-- Indexes for table `inventory`

--

ALTER TABLE `inventory`

  ADD UNIQUE KEY `id` (`id`);

--

-- Indexes for table `invoice`

--

ALTER TABLE `invoice`

  ADD UNIQUE KEY `id` (`id`);

--

-- AUTO\_INCREMENT for dumped tables

--

--

-- AUTO\_INCREMENT for table `inventory`

--

ALTER TABLE `inventory`

  MODIFY `id` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=20;

--

-- AUTO\_INCREMENT for table `invoice`

--

ALTER TABLE `invoice`

  MODIFY `id` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=81;

COMMIT;

/\*!40101 SET CHARACTER\_SET\_CLIENT=@OLD\_CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET CHARACTER\_SET\_RESULTS=@OLD\_CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET COLLATION\_CONNECTION=@OLD\_COLLATION\_CONNECTION \*/;

from tkinter import \*

from tkinter import ttk

from tkinter import messagebox

import tkinter as tk

import datetime

import pymysql

# Main component

root = Tk()

root.title('Inventory Jake Brian Yap BSIT 3A - PYTHON')

root.geometry("900x700")

dataTree = ttk.Treeview(root)

# Placeholders for inventory

ph1 = StringVar()

ph2 = StringVar()

ph3 = StringVar()

ph4 = StringVar()

# Placeholders for customers

ph5 = StringVar()

ph6 = StringVar()

ph7 = StringVar()

ph8 = StringVar()

ph9 = StringVar()

# Functions

def connection():

    conn = pymysql.connect(

        host='localhost',

        user='root',

        password='',

        db='finals\_db'

    )

    return conn

    print('con yoo')

def populateInventory():

    #Clear treeview

    for rows in myTree.get\_children():

        myTree.delete(rows)

        myTree2.delete(rows)

    conn = connection()

    cursor = conn.cursor()

    cursor.execute("SELECT \* FROM inventory")

    results = cursor.fetchall()

    conn.commit()

    conn.close()

    #print(type(results))

    for result in results:

        myTree.insert(parent='', index=len(results), iid=result, text='', values=(result[0], result[1], result[2], result[3], result[4]))

        myTree2.insert(parent='', index=len(results), iid=result, text='', values=(result[0], result[1], result[2], result[3], result[4]))

def addProduct():

    product = ph1.get()

    price = ph2.get()

    quantity = ph3.get()

    description = ph4.get()

    if (product == "" or product == " ") or (price == "" or price == " ") or (quantity == "" or quantity == " ") or (description == "" or description == " "):

        messagebox.showinfo("Error", "Please fill up the blank entry")

        return

    else:

        try:

            conn = connection()

            cursor = conn.cursor()

            cursor.execute("INSERT INTO inventory (product, price, quantity, description) VALUES ('{}', '{}', '{}', '{}')".format(product, price, quantity, description))

            conn.commit()

            conn.close()

            # Clear entries

            productEntry.delete(0, END)

            priceEntry.delete(0, END)

            quantityEntry.delete(0, END)

            descriptionEntry.delete(0, END)

            messagebox.showinfo("Success","Added successfully")

            populateInventory()

        except:

            messagebox.showinfo("Error", "skrrt error")

            return

    populateInventory()

def deleteProduct():

    id = myTree2.selection()[0]

    try:

        conn = connection()

        cursor = conn.cursor()

        cursor.execute("DELETE FROM inventory WHERE id='{}'".format(id))

        conn.commit()

        conn.close()

        populateInventory()

        messagebox.showinfo("Success","Deleted successfully")

    except:

        messagebox.showinfo("Error", "Skrrt error")

        return

def calculate():

    total = 0

    for row in myTree3.get\_children():

        # [1] is price, [2] is quantity, check treeview.

        total += float(myTree3.item(row)['values'][1]) \* float(myTree3.item(row)['values'][2])

    totalLabel.config(text = "Total: ₱{}".format(total))

    return total

def createInvoice(receiptNumber):

    total = calculate()

    def clearCart():

        for rows in myTree3.get\_children():

            myTree3.delete(rows)

        totalLabel.config(text = "Total: 0")

        # Clear Entries

        customerNameEntry.delete(0, END)

        customerMailEntry.delete(0, END)

        customerAddressEntry.delete(0, END)

        itemIdEntry.delete(0, END)

        customerQuantityEntry.delete(0, END)

        top.destroy()

    #Tkinter

    top = Toplevel(root)

    top.geometry("700x350")

    top.title("INVOICE")

    invoiceName = Label(top, text="Customer Name: {}".format(ph5.get()), pady=10).pack()

    invoiceMail = Label(top, text="Customer Email: {}".format(ph6.get()), pady=10).pack()

    invoiceAddress = Label(top, text="Customer Address: {}".format(ph7.get()), pady=10).pack()

    invoiceTime = Label(top, text="Date and time of purchase: {}".format(datetime.datetime.now().strftime("%c")), pady=10).pack()

    #TreeView

    myTree4 = ttk.Treeview(top, show='headings', height=5, padding=10)

    myTree4['columns']=('Product', 'Price', 'Quantity', 'Description')

    myTree4.column('#0', width=0, stretch=NO)

    myTree4.column('Product', anchor=CENTER, width=200)

    myTree4.column('Price', anchor=CENTER, width=150)

    myTree4.column('Quantity', anchor=CENTER, width=150)

    myTree4.column('Description', anchor=CENTER, width=150)

    myTree4.heading('Product', text='Product', anchor=CENTER)

    myTree4.heading('Price', text='Price', anchor=CENTER)

    myTree4.heading('Quantity', text='Quantity', anchor=CENTER)

    myTree4.heading('Description', text='Description', anchor=CENTER)

    myTree4.pack(expand=1)

    totalLabel2 = Label(top, text="Total: ₱{}".format(total))

    totalLabel2.pack()

    try:

        for row in myTree3.get\_children():

            myTree4.insert(parent='', index="end", iid=row, text='', values=(myTree3.item(row)['values'][0], myTree3.item(row)['values'][1], myTree3.item(row)['values'][2], myTree3.item(row)['values'][3]))

    except Exception as e:

        print(e)

    top.protocol("WM\_DELETE\_WINDOW", clearCart)

def invoice():

    name = ph5.get()

    mail = ph6.get()

    address = ph7.get()

    if (name == "" or name == " ") or (mail == "" or mail == " ") or (address == "" or address == " ") or (len(myTree3.get\_children()) <= 0):

        messagebox.showinfo("Error", "Please fill up the blank entry")

        return

    else:

        try:

            conn = connection()

            cursor = conn.cursor()

            total = calculate()

            cursor.execute("INSERT INTO invoice (name, mail, address, total) VALUES ('{}', '{}', '{}', '{}')".format(name, mail, address, total))

            conn.commit()

            conn.close()

            createInvoice(cursor.lastrowid)

        except:

            messagebox.showinfo("Error", "skrrt error")

            return

def removeProduct():

    id = myTree3.selection()[0]

    myTree3.delete(id)

def addToCart():

    itemId = ph8.get()

    quantity = ph9.get()

    if (itemId == "" or itemId == " ") or (quantity == "" or quantity == " "):

        messagebox.showinfo("Error", "Please fill up the blank entry")

        return

    else:

        try:

            conn = connection()

            cursor = conn.cursor()

            cursor.execute("SELECT \* FROM inventory")

            results = cursor.fetchall()

            conn.commit()

            conn.close()

            for result in results:

                # Convert to INT!

                if result[0] == int(itemId):

                    myTree3.insert(parent='', index="end", values=(result[1], result[2], quantity, result[4]))

            calculate()

        except:

            messagebox.showinfo("Error", "Skrrt error")

            return

# Tabs

# Tab parent

tabParent = ttk.Notebook(root)

tabParent.pack(expand=1, fill='both')

# Tab children

inventoryTab = Frame(tabParent)

manageTab = Frame(tabParent)

customerTab = Frame(tabParent)

tabParent.add(inventoryTab, text="Inventory")

tabParent.add(manageTab, text="Manage Inventory")

tabParent.add(customerTab, text="Customer")

# INVENTORY WIDGETS (TAB 1)

# inventoryWidget = ttk.Frame(inventoryTab)

# inventoryWidget.place(relx=.5, rely=.5, anchor="center")

myTree = ttk.Treeview(inventoryTab, show='headings', height=10, padding=10)

myTree['columns']=('Id', 'Product', 'Price', 'Quantity', 'Description')

myTree.column('#0', width=0, stretch=NO)

myTree.column('Id', anchor=CENTER, width=50)

myTree.column('Product', anchor=CENTER, width=200)

myTree.column('Price', anchor=CENTER, width=150)

myTree.column('Quantity', anchor=CENTER, width=150)

myTree.column('Description', anchor=CENTER, width=150)

myTree.heading('Id', text='Id', anchor=CENTER)

myTree.heading('Product', text='product', anchor=CENTER)

myTree.heading('Price', text='Price', anchor=CENTER)

myTree.heading('Quantity', text='Quantity', anchor=CENTER)

myTree.heading('Description', text='Description', anchor=CENTER)

myTree.pack(expand=1)

# Button

refreshButton = Button(inventoryTab, text="Refresh", width = '10', height = '2', pady=5, command=populateInventory)

refreshButton.pack(expand=1)

# MANAGE INVENTORY WIDGETS (TAB 2)

# Textboxes, call grid on ANOTHER line so it will have a delete method.

manageFrame = Frame(manageTab)

productLabel = Label(manageFrame, text='Product name: ').grid(row=0, column=0, sticky='e')

productEntry = Entry(manageFrame, textvariable=ph1)

productEntry.grid(row=0, column=1, pady=10)

priceLabel = Label(manageFrame, text='Price: ').grid(row=1, column=0, sticky='e')

priceEntry = Entry(manageFrame, textvariable=ph2)

priceEntry.grid(row=1, column=1, pady=10)

quantityLabel = Label(manageFrame, text='Product quantity: ').grid(row=2, column=0, sticky='e')

quantityEntry = Entry(manageFrame, textvariable=ph3)

quantityEntry.grid(row=2, column=1, pady=10)

descriptionLabel = Label(manageFrame, text='Product description: ').grid(row=3, column=0, sticky='e')

descriptionEntry = Entry(manageFrame, textvariable=ph4)

descriptionEntry.grid(row=3, column=1, pady=10)

manageFrame.pack()

# Treeview

myTree2 = ttk.Treeview(manageTab, show='headings', height=10, padding=10)

myTree2['columns']=('Id', 'Product', 'Price', 'Quantity', 'Description')

myTree2.column('#0', width=0, stretch=NO)

myTree2.column('Id', anchor=CENTER, width=50)

myTree2.column('Product', anchor=CENTER, width=200)

myTree2.column('Price', anchor=CENTER, width=150)

myTree2.column('Quantity', anchor=CENTER, width=150)

myTree2.column('Description', anchor=CENTER, width=150)

myTree2.heading('Id', text='Id', anchor=CENTER)

myTree2.heading('Product', text='product', anchor=CENTER)

myTree2.heading('Price', text='Price', anchor=CENTER)

myTree2.heading('Quantity', text='Quantity', anchor=CENTER)

myTree2.heading('Description', text='Description', anchor=CENTER)

myTree2.pack(expand=1)

# Buttons

addButton = Button(manageTab, text="Add product", width = '10', height = '2', pady=1, command=addProduct).pack(side=LEFT, expand=1)

deleteButton = Button(manageTab, text="Delete product", width = '10', height = '2', pady=1, command=deleteProduct).pack(side=RIGHT, expand=1)

refreshButton2 = Button(manageTab, text="Refresh", width = '10', height = '2', pady=5, command=populateInventory).pack(expand=1)

# CUSTOMER WIDGETS (TAB 3)

# Upper

customerFrame = LabelFrame(customerTab, text='Personal Informations')

logoLabel = Label(customerFrame, text= "Customers Order Management Panel", font='Helvetica 15 bold').grid(row=0, column=0, columnspan=2, pady=10, padx=100)

customerNameLabel = Label(customerFrame, text="Customer's name: ").grid(row=1, column=0, pady=5)

customerNameEntry = Entry(customerFrame, textvariable=ph5, width=50)

customerNameEntry.grid(row=1, column=1)

customerMailLabel = Label(customerFrame, text="Customer's email: ").grid(row=2, column=0, pady=5)

customerMailEntry = Entry(customerFrame, textvariable=ph6, width=50)

customerMailEntry.grid(row=2, column=1)

customerAddressLabel = Label(customerFrame, text="Customer's address: ").grid(row=3, column=0, pady=5)

customerAddressEntry = Entry(customerFrame, textvariable=ph7, width=50)

customerAddressEntry.grid(row=3, column=1)

# Lower, emulate the groupbox effect

bottomLabelFrame = LabelFrame(customerTab, text='Order Informations')

itemIdLabel = Label(bottomLabelFrame, text="Item ID: ").grid(row=0, column=0, pady=10, padx=20, sticky=NS)

itemIdEntry = Entry(bottomLabelFrame, textvariable=ph8)

itemIdEntry.grid(row=0, column=1)

customerQuantityLabel = Label(bottomLabelFrame, text="Quantity: ").grid(row=0, column=3, pady=10, padx=20)

customerQuantityEntry = Entry(bottomLabelFrame, textvariable=ph9)

customerQuantityEntry.grid(row=0, column=4)

#Buttons

addToCart = Button(bottomLabelFrame, text="Add to cart", pady=1, command=addToCart).grid(row=0, column=5)

removeButton = Button(bottomLabelFrame, text="Remove from cart", pady=1, command=removeProduct).grid(row=0, column=6)

invoiceButton = Button(bottomLabelFrame, text="Invoice", pady=1, command=invoice).grid(row=0, column=7)

customerFrame.pack()

bottomLabelFrame.pack(fill=X)

#TreeView

myTree3 = ttk.Treeview(customerTab, show='headings', height=10, padding=10)

myTree3['columns']=('Product', 'Price', 'Quantity', 'Description')

myTree3.column('#0', width=0, stretch=NO)

myTree3.column('Product', anchor=CENTER, width=200)

myTree3.column('Price', anchor=CENTER, width=150)

myTree3.column('Quantity', anchor=CENTER, width=150)

myTree3.column('Description', anchor=CENTER, width=150)

myTree3.heading('Product', text='Product', anchor=CENTER)

myTree3.heading('Price', text='Price', anchor=CENTER)

myTree3.heading('Quantity', text='Quantity', anchor=CENTER)

myTree3.heading('Description', text='Description', anchor=CENTER)

myTree3.pack(expand=1)

totalLabel = Label(customerTab, text="Total: 0", pady=10)

totalLabel.pack(side=BOTTOM)

# Populate data on load

populateInventory()

# Mainloop

root.mainloop()