



Title: **APP Semester Project Report**

Group Number 1

Group Members: Name | Reg. No.

Saad Ali Babar | 402049

M Awais Saleem | 400447

Andaleeb Ghazal | 399697

Course: **AI-853 Advance Programming in Python**

Instructor: **Dr. Muhammad Jawad Khan**

Submitted On: 3-Jan-2023

Robotics and Intelligent Machine Engineering RIME Department
NUST School of Mechanical and Manufacturing Engineering SMME

SHOE STORE INVENTORY MANAGEMENT SYSTEM

Overview

We are making a shoe store inventory management system. This inventory management system will allow the shoe store to search a product with a tag number keep a record of their inventory, update and view product detail, category and count. This system is able to check the particular products using any code used for particular shoe used by the shoe manufacturer company and to locate the product storage information.

In this project we have selected certain features (mentioned below) that our inventory management system will have. As we are at the store side of the management system we have and admin and store man access and features are mainly distributed accordingly.

Selected Project Features:

Valid user or working employee will be able to enter product details, their code, and price and categorize products according to their size and price. This user can also search any particular product using price and size options as well as using product code options to check whether it is available in the Shoe Shop Management System or not.

Following all the selected features define scope of the Inventory Management System

1. User Accounts Registration / Sign Up/In

In order to use Shoe Store inventory management system all its users (Admin and Store man) will have a valid user id and password associated with it.

- Select Admin or Store man
- Login ID
- Password
- Forget Password
- Dashboard for Admin/Store Man

2. Admin Login [CRUD]

Admin will be responsible to provide their user id and password by creating it. Admin can add new user, delete user, modify user, generate bills, and get information on. Check stock, get product details and inventory count or availability.

- Add New Store Man
- Add New Products
- Add/Edit Product Categories and Details
- View Product Details
- Check Inventory Availability/Count
- Receive Alerts

3. Store man Login/ Access

Store man can check inventory update the details and check the availability and count of items in the stock. Also the store man is responsible to update the consume products count, after a certain product is moved from the inventory along the supply chain.

Check Inventory Availability/Count

View/ Update Product Details

Consume Products (Update Inventory Count)

4. Add Products (Name, Category, Image, etc.)

In the add products module the new products can be added to the stock and pre-existing products can be updated when new stock arrives.

- Assign Tag Number/Code to Products
- Article Company
- Specify Color/Size/Image/Type/Collection

5. Add Products Categories

Different product categories can be added to specify the product

- Article Tag/Code (e.g. Metro-30601955 to track inventory) Fixed
- Shoe Color (Black, Brown etc.)
- Shoe Size (8,9,10 etc.)
- Shoe Image(s)
- Shoe Type (Formal Informal)
- Shoe Collection (Men, Women, Kids)

6. Consume Products

This will allow to update the count and status of product if it is sent from the inventory to next stage of supply chain.

7. Search Products

This will allow the user (admin or storeman) to check the product in the inventory by using a tag number to check its count.

Frameworks Used:

Front End: **Tkinter**

Back-end: **Flask** (Optional/If Needed)

Database: **MySQL**

Feature Distribution:

Group Member: Muhammad Awais Saleem

1. User account registration and sign in

Admin and store man can log in using this page. Admin is redirected to admin panel after logging in while store man is redirected to store man's dashboard. The only difference between both dashboards is that the admin dashboard has an additional feature of adding more store man.

This can be done using frontend (Tkinter GUI), backend (flask micro-framework of python) and MYSQL database. GUI will be linked with the backend and all the necessary information is transferred to the backend to proceed with the main logic. The backend, based on logic, decides what should be the response to that specific request and then the GUI acts accordingly.

2. Admin Login [Create(Add) Read(search) Update Delete CRUD]

- **Add New Store Man**

This feature allows the admin to add multiple store man. Usernames and passwords are stored in the database. All the details are fetched from GUI and the backend sends it to the database for storing.

- **Add New Products**

After clicking on this button, add product screen will be opened and the user can enter product details. The product details are fetched by the backend and stored in the database.

- **Update Product Categories and Details**

This allows the user to edit the particular product detail. All the necessary information is provided to the frontend by the backend is fetched from the database and then the user can edit and submit. The submit functionality will behave similarly to adding a new product.

- **View/Search Product Details**

This feature allows the user to check product details of any particular product by entering the article number.

- **Check Inventory Availability/Count(Quantity)**

If the user wants to check only the count of products, this feature only shows the inventory count of that specific product by entering the article number.

3. Search Products

This will allow the user (admin or store man) to check the product in the inventory by using a tag number to check its count how many of a particular type of products are there.

Group Member: Saad Ali Babar

4. Store Man Login/Access

This screen will appear when store man will login from the main screen (*User account registration and sign in*). This GUI will be presenting all the functionalities of the store man.

- **Check Inventory Availability/Count(Quantity)**

Enter a product code to check how many of those items are present in the inventory.

When the check inventory button on the GUI of store-man will be pressed a screen will be opened.

- **Update Product Categories and Details**

This allows the user to edit the particular product detail. All the necessary information is provided to the frontend by the backend is fetched from the database and then the user can edit and submit. The submit functionality will behave similarly to adding a new product.

- **View/Search Product Details**

This feature allows the user to check product details of any particular product by entering the article number.

- **Consume(Delete) Products (Update Inventory Count)**

Store man will be able to update the inventory count and mark products being outgoing or incoming.

5. Consume(Delete) Products

This will allow to update the count and status of product if it is sent from the inventory to next stage of supply chain. So when the store man will click on the consume product button on his interface this screen will be opened. It will allow a particular item to be updated in the inventory with respect to its count.

Group Member: Andleeb Ghazal

6. Update products categories:

It contains details such as Shoe's color (black, brown/any color), size (7,8,9 etc.), formal shoes or informal, shoe collection for men woman and kids. Admin may check the stock of specific item.

7. Add products (Name, Category, Image):

Flask will provide all these details to computer and tkinter will create a window and display tag number button. This button contains details of manufacturing company, shoe image, it's all available colors, sizes and when admin enters tag code. Admin may also check that how much pieces of specific item are remaining.

Code:

Login Screen

```
from tkinter import *
import requests

def getvalue():
    print(f"The value of username is {username.get()}")
    print(f"The value of password is {password.get()}")
    # login_screen.destroy()

entered_username = username.get()
entered_password = password.get()

url = 'http://127.0.0.1:5000/login'
myobj = {
    "username" : entered_username,
    "password" : entered_password
}

x = requests.post(url, json = myobj)
x.raise_for_status()
# access JSON content
jsonResponse = x.json()
print("Entire JSON response")
print(jsonResponse)
```

```

if jsonResponse['status'] == 200:
    if entered_username=='admin':
        login_screen.destroy()
        import dashboard_admin
    else:
        # if the user is storeman then open dashboard that does not contain all features.
        login_screen.destroy()
        import storeman_dashboard
elif jsonResponse['status'] == 401:
    print("incorrect username or password")
    Label(login_screen, text="incorrect usrname or password").pack(side=BOTTOM)

```

```

login_screen = Tk()
login_screen.title("Login")
login_screen.geometry("300x210")
Label(login_screen, text="Please enter login details").pack()
Label(login_screen, text="").pack()
Label(login_screen, text="Username").pack()

```

```

username = StringVar()
password = StringVar()

```

```

username_login_entry = Entry(login_screen, textvariable=username)
username_login_entry.pack()
Label(login_screen, text="").pack()
Label(login_screen, text="Password").pack()
password_login_entry = Entry(
    login_screen, textvariable=password, show='*')

```

```
password__login_entry.pack()
Label(login_screen, text="").pack()
Button(login_screen, text="Login", width=10, height=1, command=getvalue).pack()
login_screen.mainloop()
```

Admin Dashboard

```
from tkinter import *
#from tkinter import ttk
import tkinter.messagebox as MessageBox
import mysql.connector as mysql
import requests

dashboard = Tk()
dashboard.title("Admin Dashboard")
storeName = "Shoe Store Inventory Management System"
```

```
def delete():
    print("Inside Delete")
    if(entry_article_code.get() == ""):
        MessageBox.showinfo("ALERT", "Please enter ID to delete row")
    else:
        con = mysql.connect(host="localhost", user="root", database="rest_try")
        cursor = con.cursor()
        cursor.execute("delete from product where article='"+ entry_article_code.get() +"'")
        cursor.execute("commit");

        entry_article_code.delete(0, 'end')
        entry_shoe_colour.delete(0, 'end')
        entry_shoe_size.delete(0, 'end')
```



```
entry_shoe_type.delete(0, 'end')
entry_gender.delete(0, 'end')
entry_qty.delete(0, 'end')
MessageBox.showinfo("Status", "Successfully Deleted")
con.close();
```

```
def update():
```

```
    print("inside update")
    article= entry_article_code.get()
    shoe_colour= entry_shoe_colour.get()
    shoe_size= entry_shoe_size.get()
    shoe_type= entry_shoe_type.get()
    gender= entry_gender.get()
    qty= entry_qty.get()
    if(article == ""):
        MessageBox.showinfo("ALERT", "Please enter ID to find the Product Details!")
    else:
        if(article == ""):
            MessageBox.showinfo("ALERT", "Please enter fields you want to update!")
        else:
            con = mysql.connect(host="localhost", user="root", database="rest_try")
            cursor = con.cursor()

            cursor.execute("update product set shoe_colour = '"+ shoe_colour +"' , shoe_size= '"+
shoe_size+"', shoe_type= '"+ shoe_type+"', gender= '"+ gender+"', qty= '"+ qty+" where article
= '"+ article +"'")

            cursor.execute("commit");

            MessageBox.showinfo("Status", "Successfully Updated")
```

```

        con.close();

def save():

    print("inside save")

    con=mysql.connect(host="localhost",user="root",database="rest_try")

    cursor=con.cursor()

    savequery = "select * from product"

    val1= ("", entry_article_code.get(), entry_shoe_colour.get(), entry_shoe_size.get(),
    entry_shoe_type.get(), entry_gender.get(),entry_qty.get())

    print(type(val1[0]), type(val1[1]),type(val1[2]),type(val1[3]),type(val1[4]),type(val1[5]),
    type(val1[6]))

    query1 = 'INSERT INTO product (id, article, shoe_colour, shoe_size, shoe_type, gender,
    qty) VALUES (%s, %s, %s, %s, %s, %s, %s)'

    val1= ("", entry_article_code.get(), entry_shoe_colour.get(), entry_shoe_size.get(),
    entry_shoe_type.get(), entry_gender.get(),entry_qty.get())

    cursor.execute(query1,val1)

    cursor.execute("commit")

    print("Save executed")

    MessageBox.showinfo("Status", "Product Details Added Successfully")

    con.close();

def searchData():

    url = 'http://127.0.0.1:5000/product/' + articleNumber.get()

    print(url)

    x = requests.get(url)

    x.raise_for_status()

```

```

# access JSON content
jsonResponse = x.json()
print("Entire JSON response")
print(jsonResponse)
size_of_array = len(jsonResponse['shoe_colour'])
if size_of_array==0:
    MessageBox.showinfo("Status", "Product not found")
else:
    for i in range(size_of_array):

        shoe_colour_Label = Label(dashboard, text="Shoe Colour: " +
jsonResponse['shoe_colour'][i])
        shoe_type_Label = Label(dashboard, text="Shoe Type: " + jsonResponse['shoe_type'][i])
        shoe_size_Label = Label(dashboard, text="Shoe Size: " +
str(jsonResponse['shoe_size'][i]))
        gender_Label = Label(dashboard, text="Category: " + str(jsonResponse['gender'][i]))
        qty_Label = Label(dashboard, text="Quantity: " + str(jsonResponse['qty'][
shoe_size_Label.grid(row=9+i, column=2, padx=10, pady=10)I]))

        gender_Label.grid(row=9+i, column=3, padx=10, pady=10)
        qty_Label.grid(row=9+i, column=4, padx=10, pady=10)
shoe_colour_Label.grid(row=9+i, column=0, padx=10, pady=10)
        shoe_type_Label.grid(row=9+i, column=1, padx=10, pady=10)

def add_new_storeman():
    dashboard.destroy()
    import add_storeman

```

```
def logout():  
    url = 'http://127.0.0.1:5000/logout'  
    print(  
        # access JSON contentURL)  
  
    jsonResponse = x.json()  
    print("Entire JSON response")  
    print(jsonResponse)  
    dashboard.destroy()  
    import login
```

```
titleLabel = Label(dashboard, text=storeName, font=('Tahoma', 30), bd=2)  
titleLabel.grid(row=0, column=0, columnspan=8, padx=20, pady=20)
```

```
article_code = Label(dashboard, text="Article Code", font=('Tahoma', 15))  
shoe_colour = Label(dashboard, text="Shoe Colour", font=('Tahoma', 15))  
shoe_size = Label(dashboard, text="Shoe Size", font=('Tahoma', 15))  
shoe_type = Label(dashboard, text="Shoe Type", font=('Tahoma', 15))  
gender = Label(dashboard, text="Category", font=('Tahoma', 15))  
qty = Label(dashboard, text="Quantity", font=('Tahoma', 15))  
article_code.grid(row=1, column=0, padx=10, pady=10)  
shoe_colour.grid(row=2, column=0, padx=10, pady=10)  
shoe_size.grid(row=3, column=0, padx=10, pady=10)  
shoe_type.grid(row=4, column=0, padx=10, pady=10)  
gender.grid(row=5, column=0, padx=10, pady=10)  
qty.grid(row=6, column=0, padx=10, pady=10)
```

```

entry_article_code = Entry(dashboard, width=25, bd=5, font=('Tahoma', 15))
entry_shoe_colour = Entry(dashboard, width=25, bd=5, font=('Tahoma', 15))
#shoesize = IntVar()
#entrySearch = Entry(dashboard, width=25, bd=5, font=(
# 'Arial bold', 15), bg="#ffffff", textvariable=shoesize)
entry_shoe_size = Entry(dashboard, width=25, bd=5, font=('Tahoma', 15))
entry_shoe_type = Entry(dashboard, width=25, bd=5, font=('Tahoma', 15))
entry_gender = Entry(dashboard, width=25, bd=5, font=('Tahoma', 15))
entry_qty = Entry(dashboard, width=25, bd=5, font=('Tahoma', 15))
entry_article_code.grid(row=1, column=1, columnspan=3, padx=5, pady=5)
entry_shoe_colour.grid(row=2, column=1, columnspan=3, padx=5, pady=5)
entry_shoe_size.grid(row=3, column=1, columnspan=3, padx=5, pady=5)
entry_shoe_type.grid(row=4, column=1, columnspan=3, padx=5, pady=5)
entry_gender.grid(row=5, column=1, columnspan=3, padx=5, pady=5)
entry_qty.grid(row=6, column=1, columnspan=3, padx=5, pady=5)

```

```

buttonEnter = Button(
    dashboard, text="Enter", padx=5, pady=5, width=5,
    bd=3, font=('Tahoma', 15), bg="#ffffff", command=save)
buttonEnter.grid(row=7, column=1, columnspan=1)

```

```

buttonUpdate = Button(
    dashboard, text="Update", padx=5, pady=5, width=5,
    bd=3, font=('Tahoma', 15), bg="#ffffff", command=update)
buttonUpdate.grid(row=7, column=2, columnspan=1)

```

```

buttonDelete = Button(
    dashboard, text="Delete", padx=5, pady=5, width=5,

```

```
        bd=3, font=('Tahoma', 15), bg="#ffffff", command=delete)
buttonDelete.grid(row=7, column=3, columnspan=1)

buttonAddStoreman = Button(
    dashboard, text="Add Storeman", padx=5, pady=5, width=15,
    bd=3, font=('Tahoma', 15), bg="#ffffff", command=add_new_storeman)
buttonAddStoreman.grid(row=7, column=0, columnspan=1)

searchLabel = Label(
    dashboard, text="Article Number", font=('Arial bold', 15))
searchLabel.grid(row=8, column=0, padx=10, pady=10)

articleNumber = StringVar()
entrySearch = Entry(dashboard, width=25, bd=5, font=(
    'Tahoma', 15), bg="#ffffff", textvariable=articleNumber)
entrySearch.grid(row=8, column=1, columnspan=3, padx=5, pady=5)

buttonSearch = Button(
    dashboard, text="Search", padx=5, pady=5, width=5,
    bd=3, font=('Arial', 15), bg="#ffffff", command=searchData)
buttonSearch.grid(row=8, column=4, columnspan=1)

buttonLogout = Button(
    dashboard, text="Log Out", padx=5, pady=5, width=5,
    bd=3, font=('Tahoma', 15), bg="#ffffff", command=logout)
buttonLogout.grid(row=7, column=4, columnspan=1)

dashboard.mainloop()
```

```
x = requests.get(url)
x.raise_for_status()    shoe_type_Label.grid(row=9+i, column=1, padx=10, pady=10)
    shoe_size_Label.grid(row=9+i, column=2, padx=10, pady=10)
```

```
### Add Store Man
```

```
from tkinter import *
```

```
# import json
```

```
import requests
```

```
# from flask import jsonify
```

```
def getvalue():
```

```
    print(f"The value of username is {username.get()}")
```

```
    print(f"The value of password is {password.get()}")
```

```
    # login_screen.destroy()
```

```
    entered_username = username.get()
```

```
    entered_password = password.get()
```

```
    entered_re_password = re_password.get()
```

```
if entered_password != entered_re_password:
```

```
    Label(login_screen, text="password does not match").pack()
```

```
else:
```

```
    url = 'http://127.0.0.1:5000/add_storeman'
```

```
    myobj = {
```

```
        "username" : entered_username,
```

```
        "password" : entered_password,
```

```
        "cnfrm_password": entered_re_password
```

```
    }
```

```
x = requests.post(url, json = myobj)
x.raise_for_status()
# access JSON content
jsonResponse = x.json()
print("Entire JSON response")
print(jsonResponse)
```

```
if jsonResponse['status'] == 200:
    login_screen.destroy()
    import dashboard2
```

```
login_screen = Tk()
login_screen.title("Sign Up")
login_screen.geometry("300x250")
Label(login_screen, text="Please enter details").pack()
Label(login_screen, text="").pack()
Label(login_screen, text="Username").pack()
```

```
username = StringVar()
password = StringVar()
re_password = StringVar()
```

```
username_signup_entry = Entry(login_screen, textvariable=username)
username_signup_entry.pack()
Label(login_screen, text="").pack()
Label(login_screen, text="Password").pack()
```



```

password__signup_entry = Entry(
    login_screen, textvariable=password, show='*')
password__signup_entry.pack()
Label(login_screen, text="retype Password").pack()
re_password__signup_entry = Entry(
    login_screen, textvariable=re_password, show='*')
re_password__signup_entry.pack()
Label(login_screen, text="").pack()
Button(login_screen, text="Sign Up", width=10, height=1, command=getvalue).pack()
login_screen.mainloop()

```

Store Man Login

```

from tkinter import *

import tkinter.messagebox as MessageBox
import mysql.connector as mysql
import requests

dashboard = Tk()
dashboard.title("Storeman Dashboard")
storeName = "Shoe Store Inventory Management System"

def delete():
    print("Inside Delete")
    if(entry_article_code.get() == ""):
        MessageBox.showinfo("ALERT", "Please enter ID to delete row")
    else:
        con = mysql.connect(host="localhost", user="root", database="rest_try")
        cursor = con.cursor()

```

```
cursor.execute("delete from product where article='"+ entry_article_code.get() +"'")
cursor.execute("commit");
```

```
entry_article_code.delete(0, 'end')
entry_shoe_colour.delete(0, 'end')
entry_shoe_size.delete(0, 'end')
entry_shoe_type.delete(0, 'end')
entry_gender.delete(0, 'end')
entry_qty.delete(0, 'end')
MessageBox.showinfo("Status", "Successfully Deleted")
con.close();
```

```
def update():
```

```
    print("inside update")
    article= entry_article_code.get()
    shoe_colour= entry_shoe_colour.get()
    shoe_size= entry_shoe_size.get()
    shoe_type= entry_shoe_type.get()
    gender= entry_gender.get()
    qty= entry_qty.get()
    if(article == ""):
        MessageBox.showinfo("ALERT", "Please enter ID to find the Product Details!")
    else:
        if(article == ""):
            MessageBox.showinfo("ALERT", "Please enter fields you want to update!")
        else:
            con = mysql.connect(host="localhost", user="root", database="rest_try")
```

```

        cursor = con.cursor()

        cursor.execute("update product set shoe_colour = '"+ shoe_colour +"', shoe_size= '"+
shoe_size+"', shoe_type= '"+ shoe_type+"', gender= '"+ gender+"', qty= '"+ qty+" where article
= '"+ article +"'")

        cursor.execute("commit");

        MessageBox.showinfo("Status", "Successfully Updated")

        con.close();

```

def save():

```

    print("inside save")

    con=mysql.connect(host="localhost",user="root",database="rest_try")

    cursor=con.cursor()

    savequery = "select * from product"

    val1 = ("", entry_article_code.get(), entry_shoe_colour.get(), entry_shoe_size.get(),
entry_shoe_type.get(), entry_gender.get(),entry_qty.get())

    print(type(val1[0]), type(val1[1]),type(val1[2]),type(val1[3]),type(val1[4]),type(val1[5]),
type(val1[6]))

    query1 = 'INSERT INTO product (id, article, shoe_colour, shoe_size, shoe_type, gender,
qty) VALUES (%s, %s, %s, %s, %s, %s, %s)'

    val1 = ("", entry_article_code.get(), entry_shoe_colour.get(), entry_shoe_size.get(),
entry_shoe_type.get(), entry_gender.get(),entry_qty.get())

    cursor.execute(query1,val1)

    cursor.execute("commit")

    print("Save executed")

    MessageBox.showinfo("Status", "Product Details Added Successfully")

```

```
con.close();
```

```
def searchData():
```

```
    url = 'http://127.0.0.1:5000/product/' + articleNumber.get()
```

```
    print(url)
```

```
    x = requests.get(url)
```

```
    x.raise_for_status()
```

```
    # access JSON content
```

```
    jsonResponse = x.json()
```

```
    print("Entire JSON response")
```

```
    print(jsonResponse)
```

```
    size_of_array = len(jsonResponse['shoe_colour'])
```

```
    if size_of_array==0:
```

```
        MessageBox.showinfo("Status", "Product not found")
```

```
    else:
```

```
        for i in range(size_of_array):
```

```
            shoe_colour_Label = Label(dashboard, text="Shoe Colour: " +  
jsonResponse['shoe_colour'][i])
```

```
            shoe_type_Label = Label(dashboard, text="Shoe Type: " + jsonResponse['shoe_type'][i])
```

```
            shoe_size_Label = Label(dashboard, text="Shoe Size: " +  
str(jsonResponse['shoe_size'][i]))
```

```
            gender_Label = Label(dashboard, text="Category: " + str(jsonResponse['gender'][i]))
```

```
            qty_Label = Label(dashboard, text="Quantity: " + str(jsonResponse['qty']
```

```
shoe_size_Label.grid(row=9+i, column=2, padx=10, pady=10))
```

```
            gender_Label.grid(row=9+i, column=3, padx=10, pady=10)
```

```
        qty_Label.grid(row=9+i, column=4, padx=10, pady=10)
shoe_colour_Label.grid(row=9+i, column=0, padx=10, pady=10)

        shoe_type_Label.grid(row=9+i, column=1, padx=10, pady=10)
```

```
def logout():
```

```
    url = 'http://127.0.0.1:5000/logout'
```

```
    print(
```

```
    # access JSON contentURL)
```

```
    jsonResponse = x.json()
```

```
    print("Entire JSON response")
```

```
    print(jsonResponse)
```

```
    dashboard.destroy()
```

```
    import login
```

```
titleLabel = Label(dashboard, text=storeName, font=('Tahoma', 30), bd=2)
```

```
titleLabel.grid(row=0, column=0, columnspan=8, padx=20, pady=20)
```

```
article_code = Label(dashboard, text="Article Code", font=('Tahoma', 15))
```

```
shoe_colour = Label(dashboard, text="Shoe Colour", font=('Tahoma', 15))
```

```
shoe_size = Label(dashboard, text="Shoe Size", font=('Tahoma', 15))
```

```
shoe_type = Label(dashboard, text="Shoe Type", font=('Tahoma', 15))
```

```
gender = Label(dashboard, text="Category", font=('Tahoma', 15))
```

```
qty = Label(dashboard, text="Quantity", font=('Tahoma', 15))
```

```
article_code.grid(row=1, column=0, padx=10, pady=10)
```

```
shoe_colour.grid(row=2, column=0, padx=10, pady=10)
```

```
shoe_size.grid(row=3, column=0, padx=10, pady=10)
```

```
shoe_type.grid(row=4, column=0, padx=10, pady=10)
```

```
gender.grid(row=5, column=0, padx=10, pady=10)
```

```
qty.grid(row=6, column=0, padx=10, pady=10)
```

```
entry_article_code = Entry(dashboard, width=25, bd=5, font=('Tahoma', 15))
```

```
entry_shoe_colour = Entry(dashboard, width=25, bd=5, font=('Tahoma', 15))
```

```
entry_shoe_size = Entry(dashboard, width=25, bd=5, font=('Tahoma', 15))
```

```
entry_shoe_type = Entry(dashboard, width=25, bd=5, font=('Tahoma', 15))
```

```
entry_gender = Entry(dashboard, width=25, bd=5, font=('Tahoma', 15))
```

```
entry_qty = Entry(dashboard, width=25, bd=5, font=('Tahoma', 15))
```

```
entry_article_code.grid(row=1, column=1, columnspan=3, padx=5, pady=5)
```

```
entry_shoe_colour.grid(row=2, column=1, columnspan=3, padx=5, pady=5)
```

```
entry_shoe_size.grid(row=3, column=1, columnspan=3, padx=5, pady=5)
```

```
entry_shoe_type.grid(row=4, column=1, columnspan=3, padx=5, pady=5)
```

```
entry_gender.grid(row=5, column=1, columnspan=3, padx=5, pady=5)
```

```
entry_qty.grid(row=6, column=1, columnspan=3, padx=5, pady=5)
```

```
buttonEnter = Button(
```

```
    dashboard, text="Enter", padx=5, pady=5, width=5,
```

```
    bd=3, font=('Tahoma', 15), bg="ffffff", command=save)
```

```
buttonEnter.grid(row=7, column=1, columnspan=1)
```

```
buttonUpdate = Button(
```

```
    dashboard, text="Update", padx=5, pady=5, width=5,
```

```
    bd=3, font=('Tahoma', 15), bg="ffffff", command=update)
```

```
buttonUpdate.grid(row=7, column=2, columnspan=1)
```

```
buttonDelete = Button(
```

```
    dashboard, text="Delete", padx=5, pady=5, width=5,
```

```
    bd=3, font=('Tahoma', 15), bg="ffffff", command=delete)
```

```
buttonDelete.grid(row=7, column=3, columnspan=1)
```

```
searchLabel = Label(
```

```
    dashboard, text="Article Number", font=('Arial bold', 15))
```

```
searchLabel.grid(row=8, column=0, padx=10, pady=10)
```

```
articleNumber = StringVar()
```

```
entrySearch = Entry(dashboard, width=25, bd=5, font=(
```

```
    'Tahoma', 15), bg="#ffffff", textvariable=articleNumber)
```

```
entrySearch.grid(row=8, column=1, columnspan=3, padx=5, pady=5)
```

```
buttonSearch = Button(
```

```
    dashboard, text="Search", padx=5, pady=5, width=5,
```

```
    bd=3, font=('Tahoma', 15), bg="#ffffff", command=searchData)
```

```
buttonSearch.grid(row=8, column=4, columnspan=1)
```

```
buttonLogout = Button(
```

```
    dashboard, text="Log Out", padx=5, pady=5, width=5,
```

```
    bd=3, font=('Tahoma', 15), bg="#ffffff", command=logout)
```

```
buttonLogout.grid(row=7, column=4, columnspan=1)
```

```
dashboard.mainloop()
```

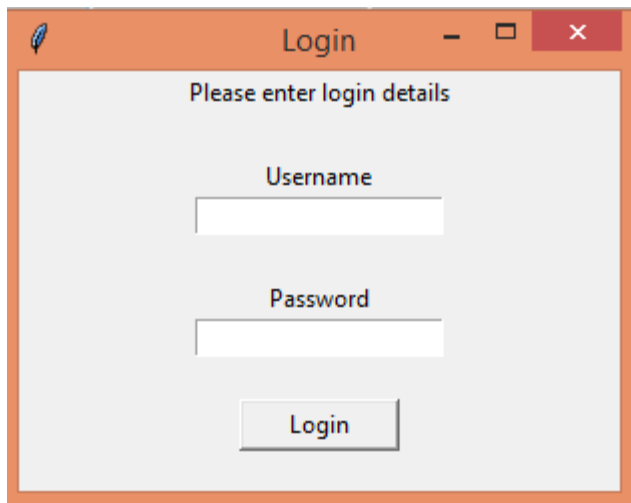
```
x = requests.get(url)
```

```
x.raise_for_status()    shoe_colour_Label.grid(row=9+i, column=0, padx=10, pady=10)
```

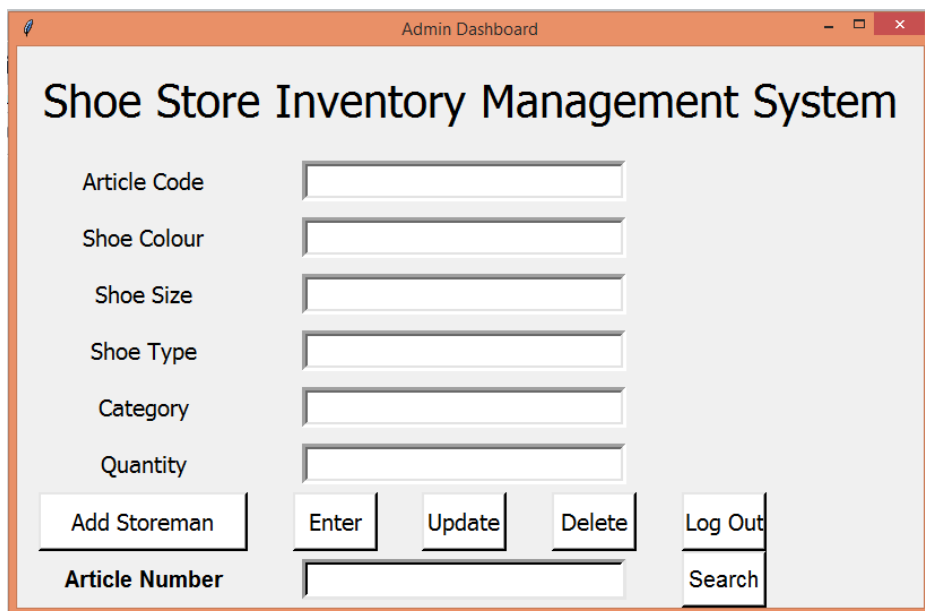
```
    shoe_type_Label.grid(row=9+i, column=1, padx=10, pady=10)
```

```
+++=====+++
```


Screens:



A screenshot of a 'Login' window. The window has a title bar with the text 'Login' and standard minimize, maximize, and close buttons. The main content area has a light gray background and contains the text 'Please enter login details' at the top. Below this text are two input fields: the first is labeled 'Username' and the second is labeled 'Password'. At the bottom of the form is a button labeled 'Login'.



A screenshot of an 'Admin Dashboard' window. The window has a title bar with the text 'Admin Dashboard' and standard minimize, maximize, and close buttons. The main content area has a light gray background and features the title 'Shoe Store Inventory Management System' at the top. Below the title are six input fields with labels to their left: 'Article Code', 'Shoe Colour', 'Shoe Size', 'Shoe Type', 'Category', and 'Quantity'. At the bottom of the form are five buttons: 'Add Storeman', 'Enter', 'Update', 'Delete', and 'Log Out'. Below the 'Add Storeman' button is a label 'Article Number' followed by an input field. To the right of this input field is a button labeled 'Search'.

Sign Up-□×


Please enter details

Username

Password

retype Password

Sign Up

Storeman Dashboard-□×

Shoe Store Inventory Management System

Article Code

Shoe Colour

Shoe Size

Shoe Type

Category

Quantity

Enter

Update

Delete

Log Out

Article Number

Search

Database:

phpMyAdmin

Recent Favorites

- New
- college
- information_schema
- inventory
- mysql
- performance_schema
- phpmyadmin
- project_inventory
 - New
 - inventoryusers
 - product
- test

Server: 127.0.0.1 » Database: project_inventory » Table: inventoryusers

Browse Structure SQL Search Insert Export Import Privileges Operations

☐ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

Extra options

			id	username	password
<input type="checkbox"/>	Edit	Copy	Delete	1	admin
<input type="checkbox"/>	Edit	Copy	Delete	2	shadow
<input type="checkbox"/>	Edit	Copy	Delete	3	new storeman
<input type="checkbox"/>	Edit	Copy	Delete	4	saad
<input type="checkbox"/>	Edit	Copy	Delete	7	Merlin

↑ ☐ Check all | With selected: Edit Copy Delete Export

☐ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

phpMyAdmin

Recent Favorites

- New
- college
- information_schema
- inventory
- mysql
- performance_schema
- phpmyadmin
- project_inventory
 - New
 - inventoryusers
 - product
- test

Server: 127.0.0.1 » Database: project_inventory » Table: inventoryusers

Browse Structure SQL Search Insert Export Import Privileges Operations

☐ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

Extra options

			id	username	password
<input type="checkbox"/>	Edit	Copy	Delete	1	admin
<input type="checkbox"/>	Edit	Copy	Delete	2	shadow
<input type="checkbox"/>	Edit	Copy	Delete	3	new storeman
<input type="checkbox"/>	Edit	Copy	Delete	4	saad
<input type="checkbox"/>	Edit	Copy	Delete	7	Merlin

↑ ☐ Check all | With selected: Edit Copy Delete Export

☐ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None