**LAB PRODUCT MANAGEMENT SYSTM**

**ROLLNO: KRISNA A (CB.EN.P2EBS22003),**

**DHANUSH R (CB.EN.P2EBS22007)**

This document contains the code for lab product management system and screenshots of the both user and admin code.

**Admin App:**

**Main.py***"""  
Author: Krishna, Dhanush  
This application is for admin for managing the Lab Product Management System.  
This file contains the program for main login page  
"""*"""  
import statements includes tkinter, firebase,  
adminPage file   
"""  
from tkinter import \*  
import pyrebase  
from tkinter import messagebox  
import adminPage  
  
"""  
Firebase configuration details  
api keys, database url, application ID  
  
"""  
  
firebaseConfig = {  
 "apiKey": "AIzaSyCeDM5eY6tXZPhQ3BNk-ZYGFh6982BbIgM",  
 "authDomain": "labprodcutmanagementsystem.firebaseapp.com",  
 "databaseURL": "https://labprodcutmanagementsystem-default-rtdb.firebaseio.com",  
 "projectId": "labprodcutmanagementsystem",  
 "storageBucket": "labprodcutmanagementsystem.appspot.com",  
 "messagingSenderId": "808287754951",  
 "appId": "1:808287754951:web:46ad7d2afc5284cae4bce2"  
}  
  
"""  
Initialize firebase configuration and   
database connection  
  
"""  
  
firebase = pyrebase.initialize\_app(firebaseConfig)  
db = firebase.database()  
  
"""  
Intialize the tkinter window using Tk() and size of the window  
  
"""  
  
root = Tk()  
root.geometry("480x240")  
root.title("Sign Up App")  
root.resizable(False, False)  
  
"""  
  
data function for creating user  
  
"""  
  
  
def data():  
 uname = e\_id.get()  
 passwd = e\_password.get()  
  
 if uname == "" or passwd == "":  
 messagebox.showinfo("Insert status", "All fields are required")  
 else:  
 userdata = {'id': uname, 'password': passwd}  
 db.child(uname).set(userdata)  
 e\_id.delete(0, "end")  
 e\_password.delete(0, "end")  
 messagebox.showinfo("user creation", "user created successfully")  
  
  
"""  
  
data function for login user  
  
"""  
  
  
def login():  
 uname = e\_id.get()  
 passwd = e\_password.get()  
  
 users = db.child(uname).get()  
 value = users.val()  
 try:  
 if uname == value['id']:  
 if value['password'] == passwd:  
 root.destroy()  
 print('hi')  
 adminPage.mainprg()  
 else:  
 messagebox.showinfo("Login Error", "Enter the password correctly!...")  
 except:  
 messagebox.showinfo("Login Error", "Enter the user name correctly!...")  
  
  
"""  
  
Label for username and password, and login button  
  
"""  
  
l1 = Label(root, text="UserName", font=("bold", 10))  
l1.place(x=20, y=30)  
  
l1 = Label(root, text="password", font=("bold", 10))  
l1.place(x=20, y=60)  
  
e\_id = Entry()  
e\_id.place(x=150, y=30)  
  
e\_password = Entry(show='\*')  
e\_password.place(x=150, y=60)  
  
create = Button(root, text="Create User", font=("itlaic",10), bg="white", command=data)  
create.place(x=20, y=180)  
  
LogIn = Button(root, text="Log In", font=("itlaic",10), bg="white", command=login)  
LogIn.place(x=150, y=180)  
  
root.mainloop()

**Admin.py:**

*"""  
import statements includes tkinter, firebase,  
adminPage file  
"""*from tkinter import \*  
import pyrebase  
from tkinter import messagebox  
  
"""  
Firebase configuration details  
api keys, database url, application ID  
  
"""  
  
  
productRequestList = {  
 "apiKey": "AIzaSyDG6W2jr4B1wsY\_LZ7fXjs79YEF2PAcM7I",  
 "authDomain": "productrequstlist.firebaseapp.com",  
 "databaseURL": "https://productrequstlist-default-rtdb.firebaseio.com",  
 "projectId": "productrequstlist",  
 "storageBucket": "productrequstlist.appspot.com",  
 "messagingSenderId": "895647350915",  
 "appId": "1:895647350915:web:74c9cc1378525fc996dbb6"  
}  
  
"""  
Initialize firebase configuration and   
database connection  
  
"""  
  
firebaseProductRequestList = pyrebase.initialize\_app(productRequestList)  
productRequestListDB = firebaseProductRequestList.database()  
  
"""  
  
Functions for mainprogram contains insert, delete, update functions  
  
"""  
  
  
def mainprg():  
 mainpage = Tk()  
 mainpage.geometry("480x240")  
 mainpage.title("Admin Login")  
 mainpage.resizable(False, False)  
  
  
  
 mainPageConfig = {  
 "apiKey": "AIzaSyCeDM5eY6tXZPhQ3BNk-ZYGFh6982BbIgM",  
 "authDomain": "labprodcutmanagementsystem.firebaseapp.com",  
 "databaseURL": "https://labprodcutmanagementsystem-default-rtdb.firebaseio.com",  
 "projectId": "labprodcutmanagementsystem",  
 "storageBucket": "labprodcutmanagementsystem.appspot.com",  
 "messagingSenderId": "808287754951",  
 "appId": "1:808287754951:web:c25c5e82a3e83f61e4bce2"  
 }  
  
 firebase = pyrebase.initialize\_app(mainPageConfig)  
 db = firebase.database()  
  
  
 def insert():  
 productid = e\_id.get()  
 productname = e\_name.get()  
 productQty = e\_qty.get()  
 if productname == "" or productid == "" or productQty == "":  
 messagebox.showinfo("Insert status", "Enter all field values")  
 else:  
 userdata = {'name': productname, 'id': productid, 'Qty': productQty}  
 db.child("products").child(productname).set(userdata)  
 e\_id.delete(0, "end")  
 e\_name.delete(0, "end")  
 e\_qty.delete(0, "end")  
 messagebox.showinfo("product creation", "product created successfully")  
  
  
 def delete():  
 if e\_name.get() == "":  
 messagebox.showinfo("Delete status", "Name is compulsory for delete")  
 else:  
 productname = e\_name.get()  
 db.child("products").child(productname).remove()  
 e\_id.delete(0, 'end')  
 e\_name.delete(0, 'end')  
 messagebox.showinfo("Delete status", "Deleted Successfully")  
  
 def update():  
 if e\_name.get() == "" or e\_qty.get() == "":  
 messagebox.showinfo("Delete status", "Name and Quantity is compulsory for delete")  
 else:  
 db.child("products").child(e\_name.get()).update({"Qty": e\_qty.get()})  
 e\_id.delete(0, "end")  
 e\_name.delete(0, "end")  
 messagebox.showinfo("Data update", "Data updated successfully")  
  
  
 def readdatas():  
 searchPage = Tk()  
 searchPage.geometry("480x240")  
 searchPage.title("product page")  
 all\_product = db.child("products").get()  
 i = 1  
 for product in all\_product.each():  
 productId = product.val().get("id")  
 productname = product.val().get("name")  
 productQty = product.val().get("Qty")  
  
 lst = [productId, productname, productQty]  
 e = Label(searchPage, width=15, text="Id", borderwidth=2, relief='ridge', anchor="w")  
 e.grid(row=0, column=1)  
 e = Label(searchPage, width=15, text="Product Name", borderwidth=2, relief='ridge', anchor="w")  
 e.grid(row=0, column=0)  
 e = Label(searchPage, width=15, text="Qty", borderwidth=2, relief='ridge', anchor="w")  
 e.grid(row=0, column=1)  
 for r in range(0, 3):  
 e = Entry(searchPage, width=15, bd=0)  
 e.grid(row=i, column=r)  
 e.insert(END, lst[r])  
 e.config(state="readonly")  
 i = i+1  
 searchPage.mainloop()  
 e\_id.delete(0, "end")  
 e\_name.delete(0, "end")  
  
 def acceptRequestFunction(event):  
 print(event)  
 productRequestListDB.child("productsRequest").child(event[3]).child(event[1]).update({"status": "accepted"})  
 product = db.child("products").child(event[1]).get()  
 productQty = product.val().get("Qty")  
 decQty = int(productQty) - int(event[0])  
 product = db.child("products").child(event[1]).update({'Qty': decQty})  
 print(product)  
 print(productQty)  
  
 def rejectRequestFunction(event):  
 productRequestListDB.child("productsRequest").child(event[3]).child(event[1]).update({"status": "rejected"})  
  
 def returnBackComponent(event):  
 productRequestListDB.child("productsRequest").child(event[3]).child(event[1]).update({"status": "Return the product"})  
  
 def changeReturnStatus(event):  
 productRequestListDB.child("productsRequest").child(event[3]).child(event[1]).update({"status": "Returned"})  
  
 def viewRequest():  
 requestPage = Tk()  
 requestPage.geometry("480x240")  
 requestPage.title("product page")  
 requestproductList = productRequestListDB.child("productsRequest").get()  
 j = 1  
 for productList in requestproductList.each():  
 userData = productList.val()  
 e = Label(requestPage, width=20, text="Qty", borderwidth=2, relief='ridge', anchor="w")  
 e.grid(row=0, column=0)  
 e = Label(requestPage, width=20, text="Request Product Name", borderwidth=2, relief='ridge', anchor="w")  
 e.grid(row=0, column=1)  
 e = Label(requestPage, width=20, text="Status", borderwidth=2, relief='ridge', anchor="w")  
 e.grid(row=0, column=2)  
 e = Label(requestPage, width=20, text="User Requested", borderwidth=2, relief='ridge', anchor="w")  
 e.grid(row=0, column=3)  
 for key, val in userData.items():  
 lstVal = []  
 for nested\_key, nested\_val in val.items():  
 lstVal.append(nested\_val)  
 for value in range(0, 4):  
 e = Entry(requestPage, width=20, bd=0)  
 e.grid(row=j, column=value)  
 e.insert(END, lstVal[value])  
 e.config(state="readonly")  
 if lstVal[2] != "Returned":  
 b4 = Button(requestPage, text="change status to returned",  
 command=lambda entry=lstVal: changeReturnStatus(entry))  
 b4.grid(row=j, column=7, sticky=E)  
 if lstVal[2] != "accepted":  
 b1 = Button(requestPage, text="Accept request", command=lambda entry=lstVal: acceptRequestFunction(entry))  
 b1.grid(row=j, column=4, sticky=E)  
 if lstVal[2] != "rejected":  
 b2 = Button(requestPage, text="Reject request", command=lambda entry=lstVal: rejectRequestFunction(entry))  
 b2.grid(row=j, column=5, sticky=E)  
  
 if lstVal[2] == "accepted":  
 b3 = Button(requestPage, text="Ask for return", command=lambda entry=lstVal: returnBackComponent(entry))  
 b3.grid(row=j, column=6, sticky=E)  
 b4 = Button(requestPage, text="change status to returned",  
 command=lambda entry=lstVal: changeReturnStatus(entry))  
 b4.grid(row=j, column=7, sticky=E)  
  
  
  
 j = j + 1  
 requestPage.mainloop()  
  
 """  
  
 Label and input for username and password, and login button  
 insert, update, delete functions  
  
 """  
  
 id = Label(mainpage, text='Enter product ID', font=("bold", 10))  
 id.place(x=20, y=30)  
  
 name = Label(mainpage, text='Enter product name', font=("italic", 10))  
 name.place(x=20, y=60)  
  
 Qty = Label(mainpage, text='Enter product Qty', font=("italic", 10))  
 Qty.place(x=20, y=90)  
  
 e\_id = Entry()  
 e\_id.place(x=150, y=30)  
  
 e\_name = Entry()  
 e\_name.place(x=150, y=60)  
  
 e\_qty = Entry()  
 e\_qty.place(x=150, y=90)  
  
 insert = Button(mainpage, text="Insert", font=("italic", 10), bg="white", command=insert)  
 insert.place(x=20, y=140)  
 delete = Button(mainpage, text="Delete", font=("italic", 10), bg="white", command=delete)  
 delete.place(x=80, y=140)  
 update = Button(mainpage, text="Update", font=("italic", 10), bg="white", command=update)  
 update.place(x=20, y=180)  
 getdataValue = Button(mainpage, text="Get Product List", font=("italic", 10), bg="white", command=readdatas)  
 getdataValue.place(x=80, y=180)  
 viewRequest = Button(mainpage, text="View Request", font=("italic", 10), bg="white", command=viewRequest)  
 viewRequest.place(x=210, y=180)  
  
 mainpage.mainloop()

**User App:**

**Main.py:**

*"""  
Author: Krishna, Dhanush  
This application is for admin for managing the Lab Product Management System.  
This file contains the program for main login page  
"""*"""  
import statements includes tkinter, firebase,  
adminPage file   
"""  
from tkinter import \*  
import pyrebase  
from tkinter import messagebox  
import userApp  
  
firebaseConfig = {  
 "apiKey": "AIzaSyALabaTvbp4bedauYSGNHvlA9mdoi53x4c",  
 "authDomain": "labproductrequestlist.firebaseapp.com",  
 "databaseURL": "https://labproductrequestlist-default-rtdb.firebaseio.com",  
 "projectId": "labproductrequestlist",  
 "storageBucket": "labproductrequestlist.appspot.com",  
 "messagingSenderId": "507388215851",  
 "appId": "1:507388215851:web:4c772fb2a7d3f72d00e06e"  
}  
  
"""  
Initialize firebase configuration and   
database connection  
  
"""  
  
userFireBase = pyrebase.initialize\_app(firebaseConfig)  
userFireBaseDB = userFireBase.database()  
  
"""  
Intialize the tkinter window using Tk() and size of the window  
  
"""  
  
root = Tk()  
root.geometry("480x240")  
root.title("User Sign Up App")  
root.resizable(False, False)  
  
  
"""  
  
data function for creating user  
  
"""  
  
def data():  
 uname = e\_id.get()  
 passwd = e\_password.get()  
  
 if uname == "" or passwd == "":  
 messagebox.showinfo("Insert status", "All fields are required")  
 else:  
 userdata = {'id': uname, 'password': passwd}  
 userFireBaseDB.child(uname).set(userdata)  
 e\_id.delete(0, "end")  
 e\_password.delete(0, "end")  
 messagebox.showinfo("user creation", "user created successfully")  
  
  
"""  
  
data function for login user  
  
"""  
  
def login():  
 uname = e\_id.get()  
 passwd = e\_password.get()  
  
 users = userFireBaseDB.child(uname).get()  
 value = users.val()  
 try:  
 if uname == value['id']:  
 if value['password'] == passwd:  
 root.destroy()  
 userApp.productreqList(uname)  
 else:  
 messagebox.showinfo("Login Error", "Enter the password correctly!...")  
 except:  
 messagebox.showinfo("Login Error", "Enter the user name correctly!...")  
  
"""  
  
Label for username and password, and login button  
  
"""  
  
  
l1 = Label(root, text="UserName", font=("bold", 10))  
l1.place(x=20, y=30)  
  
l1 = Label(root, text="password", font=("bold", 10))  
l1.place(x=20, y=60)  
  
e\_id = Entry()  
e\_id.place(x=150, y=30)  
  
e\_password = Entry(show='\*')  
e\_password.place(x=150, y=60)  
  
create = Button(root, text="Create User", font=("italic", 10), bg="white", command=data)  
create.place(x=20, y=180)  
  
LogIn = Button(root, text="Log In", font=("italic", 10), bg="white", command=login)  
LogIn.place(x=150, y=180)  
  
root.mainloop()

**userApp.py:**

*"""  
import statements includes tkinter, firebase,  
adminPage file  
"""*from tkinter import \*  
import pyrebase  
from tkinter import messagebox  
  
"""  
Firebase configuration details  
api keys, database url, application ID  
  
"""  
  
"""  
Initialize firebase configuration and   
database connection  
  
"""  
  
searchPageConfig = {  
 "apiKey": "AIzaSyCeDM5eY6tXZPhQ3BNk-ZYGFh6982BbIgM",  
 "authDomain": "labprodcutmanagementsystem.firebaseapp.com",  
 "databaseURL": "https://labprodcutmanagementsystem-default-rtdb.firebaseio.com",  
 "projectId": "labprodcutmanagementsystem",  
 "storageBucket": "labprodcutmanagementsystem.appspot.com",  
 "messagingSenderId": "808287754951",  
 "appId": "1:808287754951:web:c25c5e82a3e83f61e4bce2"  
 }  
  
firebaseSearchRequestList = pyrebase.initialize\_app(searchPageConfig)  
db = firebaseSearchRequestList.database()  
  
productRequestList = {  
 "apiKey": "AIzaSyDG6W2jr4B1wsY\_LZ7fXjs79YEF2PAcM7I",  
 "authDomain": "productrequstlist.firebaseapp.com",  
 "databaseURL": "https://productrequstlist-default-rtdb.firebaseio.com",  
 "projectId": "productrequstlist",  
 "storageBucket": "productrequstlist.appspot.com",  
 "messagingSenderId": "895647350915",  
 "appId": "1:895647350915:web:74c9cc1378525fc996dbb6"  
}  
  
firebaseProductRequestList = pyrebase.initialize\_app(productRequestList)  
productRequestListDB = firebaseProductRequestList.database()  
  
  
"""  
  
Functions for mainprogram contains insert, delete, update functions  
  
"""  
  
def productreqList(username):  
  
 productRequestList = Tk()  
 productRequestList.geometry("480x240")  
 productRequestList.title("Product Request List")  
  
 def readdatas():  
 print(username)  
 searchPage = Tk()  
 searchPage.geometry("480x240")  
 searchPage.title("product page")  
 all\_product = db.child("products").get()  
 i = 1  
 for product in all\_product.each():  
 print(product.key())  
 productId = product.val().get("id")  
 productname = product.val().get("name")  
 productQty = product.val().get("Qty")  
  
 lst = [productname, productQty]  
 e = Label(searchPage, width=15, text="Product Name", borderwidth=2, relief='ridge', anchor="w")  
 e.grid(row=0, column=0)  
 e = Label(searchPage, width=15, text="Qty", borderwidth=2, relief='ridge', anchor="w")  
 e.grid(row=0, column=1)  
 for r in range(0, 2):  
 e = Entry(searchPage, width=15, bd=0)  
 e.grid(row=i, column=r)  
 e.insert(END, lst[r])  
 e.config(state="readonly")  
 print(productId)  
 i = i + 1  
 searchPage.mainloop()  
 e\_name.delete(0, "end")  
  
  
 def raiseProductRequest():  
 productName = e\_name.get()  
 productQty = e\_qty.get()  
 if productName == "" or productQty == "":  
 messagebox.showinfo("Insert status", "Enter all field values")  
 else:  
 requestData = {'name': productName, 'Qty': productQty, 'username': username, 'status': 'pending'}  
 productRequestListDB.child("productsRequest").child(username).child(productName).set(requestData)  
 e\_name.delete(0, "end")  
 e\_qty.delete(0, "end")  
 messagebox.showinfo("Request creation status", "Product request created successfully")  
  
 def viewMyProductRequestList():  
  
 viewMyProductRequestList = Tk()  
 viewMyProductRequestList.geometry("480x240")  
 viewMyProductRequestList.title("My request List")  
 requestproductList = productRequestListDB.child("productsRequest").child(username).get()  
 j = 1  
 for productList in requestproductList.each():  
 userData = productList.val()  
 print(userData.items())  
 e = Label(viewMyProductRequestList, width=20, text="Qty", borderwidth=2, relief='ridge', anchor="w")  
 e.grid(row=0, column=0)  
 e = Label(viewMyProductRequestList, width=20, text="Request Product Name", borderwidth=2, relief='ridge', anchor="w")  
 e.grid(row=0, column=1)  
 e = Label(viewMyProductRequestList, width=20, text="Status", borderwidth=2, relief='ridge', anchor="w")  
 e.grid(row=0, column=2)  
 e = Label(viewMyProductRequestList, width=20, text="User Requested", borderwidth=2, relief='ridge', anchor="w")  
 e.grid(row=0, column=3)  
 lstVal = []  
 for key, val in userData.items():  
 lstVal.append(val)  
 for value in range(0, 4):  
 e = Entry(viewMyProductRequestList, width=20, bd=0)  
 e.grid(row=j, column=value)  
 e.insert(END, lstVal[value])  
 e.config(state="readonly")  
 j = j + 1  
  
 viewMyProductRequestList.mainloop()  
  
 def returnRequest():  
  
 returnRequestList = Tk()  
 returnRequestList.geometry("480x240")  
 returnRequestList.title("My request List")  
 returnRequestListValues = productRequestListDB.child("productsRequest").child(username).get()  
 j = 1  
 for productList in returnRequestListValues.each():  
 userData = productList.val()  
 e = Label(returnRequestList, width=20, text="Qty", borderwidth=2, relief='ridge', anchor="w")  
 e.grid(row=0, column=0)  
 e = Label(returnRequestList, width=20, text="Request Product Name", borderwidth=2, relief='ridge',  
 anchor="w")  
 e.grid(row=0, column=1)  
 e = Label(returnRequestList, width=20, text="Status", borderwidth=2, relief='ridge', anchor="w")  
 e.grid(row=0, column=2)  
 e = Label(returnRequestList, width=20, text="User Requested", borderwidth=2, relief='ridge',  
 anchor="w")  
 e.grid(row=0, column=3)  
 lstVal = []  
 if userData['status'] == 'Return the product':  
 for key, val in userData.items():  
 lstVal.append(val)  
 for value in range(0, 4):  
 e = Entry(returnRequestList, width=20, bd=0)  
 e.grid(row=j, column=value)  
 e.insert(END, lstVal[value])  
 e.config(state="readonly")  
 j = j + 1  
  
 returnRequestList.mainloop()  
  
 """  
  
 Label and input for username and password, and login button  
 insert, update, delete functions  
  
 """  
  
 name = Label(productRequestList, text='Enter product name', font=("italic", 10))  
 name.place(x=20, y=60)  
  
 Qty = Label(productRequestList, text='Enter product Qty', font=("italic", 10))  
 Qty.place(x=20, y=90)  
  
 e\_name = Entry()  
 e\_name.place(x=150, y=60)  
  
 e\_qty = Entry()  
 e\_qty.place(x=150, y=90)  
  
 getdataValue = Button(productRequestList, text="Get Product List", font=("italic", 10), bg="white", command=readdatas)  
 getdataValue.place(x=80, y=180)  
  
 rasieRequestButton = Button(productRequestList, text="Raise Request", font=("italic", 10), bg="white", command=raiseProductRequest)  
 rasieRequestButton.place(x=210, y=180)  
  
 viewMyRequestList = Button(productRequestList, text="My Product Requests", font=("italic", 10), bg="white", command=viewMyProductRequestList)  
 viewMyRequestList.place(x=320, y=180)  
  
 viewMyRequestList = Button(productRequestList, text="My Return Requests", font=("italic", 10), bg="white",  
 command=returnRequest)  
 viewMyRequestList.place(x=80, y=215)  
  
 productRequestList.mainloop()