PY YOUNESS ZAINI

2024

CONTROL ATTANDANCE

POINTEGE AUTO

fatima alaoi

[Tapez l'adresse de la société]

#هنا إضافة المكتبات تا البيتون

import cv2

import tkinter as tk

from tkinter import ttk, messagebox

import qrcode

from PIL import Image, ImageTk

import datetime

import sqlite3

from ttkthemes import ThemedStyle

from reportlab.pdfgen import canvas

from reportlab.lib.pagesizes import letter

from reportlab.lib import colors

from reportlab.platypus import Table, TableStyle

from openpyxl import Workbook

from tkinter import filedialog

#تطبيق التحكم في حضور الفصل الدراسي

class AttendanceControlApp:

def \_\_init\_\_(self, root):

self.root = root

self.root.title("Attendance Control")

# Apply the themed style / تطبيق النمط تحت عنوان

self.style = ThemedStyle(root)

self.style.set\_theme("clam") # Choose a modern theme العديد من الخلفيات المهم هو البحث/ ("equilux")/("clam")/("arc")

# Variables /المتغيرات

self.qr\_code\_data = tk.StringVar()

self.check\_in\_time = None

self.check\_out\_time = None

self.total\_time\_var = tk.StringVar(value="Total Time: Not Available")

self.database\_connection = sqlite3.connect("attendance.db")

self.create\_table\_if\_not\_exists()

# GUI components/ مكونات واجهة المستخدم الرسومية

self.label\_qr\_code = ttk.Label(root, text="رمز الإستجابة السريعة:")

self.entry\_qr\_code = ttk.Entry(root, textvariable=self.qr\_code\_data, width=30)

self.btn\_generate\_qr = ttk.Button(root, text="حفظ رمز الإستجابة السريعة", command=self.generate\_qr\_code)

self.label\_qr\_image = ttk.Label(root)

self.btn\_start\_scanning = ttk.Button(root, text="تشغيل الكاميرا ", command=self.start\_scanning)

self.btn\_check\_in\_out = ttk.Button(root, text="دخول/ خروج", command=self.check\_in\_out)

self.label\_status = ttk.Label(root, text="Status: لا دخول / لاخروج")

self.btn\_generate\_pdf = ttk.Button(root, text="PDF حفظ", command=self.generate\_pdf\_report)

self.btn\_generate\_xlsx = ttk.Button(root, text="XLSX حفظ", command=self.generate\_xlsx\_report)

self.btn\_delete\_record = ttk.Button(root, text="حدف", command=self.delete\_record)

self.label\_total\_time = ttk.Label(root, textvariable=self.total\_time\_var)

self.title\_label = ttk.Label(root, text="برامج يونس لتسيير المقاولات ", font=("Helvetica", 20, "bold"))

# Treeview with modern styling / تريفيو مع التصميم الحديث

self.treeview = ttk.Treeview(root, columns=("ID", "QR Data", "Check-In Time", "Check-Out Time"))

self.treeview.heading("#1", text="ID")

self.treeview.column("ID", width=70,anchor="center")

self.treeview.heading("#2", text="QR Data")

self.treeview.column("QR Data", width=70,anchor="center")

self.treeview.heading("#3", text="Check-In Time")

self.treeview.column("Check-In Time", width=70,anchor="center")

self.treeview.heading("#4", text="Check-Out Time")

self.treeview.column("Check-Out Time", width=70,anchor="center")

self.treeview.column("#1", width=120)

self.treeview.column("#2", width=300)

self.treeview.column("#3", width=150)

self.treeview.column("#4", width=150)

#هاد البوطونة قالب راه كتمسح الجدول بإستمرار HHHHHHHHHH

self.treeview.bind("<ButtonRelease-1>", self.on\_treeview\_click)

# التحديث الذاتي HADI KAT9AD LINA TREVIW WA IDHAR LMA3LOUMAT HHH

self.refresh\_treeview()

#تتقاد الخانات لي في الجدول

self.treeview['show']='headings'

# Layout/ تَخطِيط

self.title\_label.grid(row=0, column=0, columnspan=4, pady=10)

self.label\_qr\_code.grid(row=1, column=0, padx=10, pady=10)

self.entry\_qr\_code.grid(row=1, column=1, padx=10, pady=10)

self.btn\_generate\_qr.grid(row=1, column=2, padx=10, pady=10)

self.label\_qr\_image.grid(row=2, column=0, columnspan=3, padx=10, pady=10)

self.btn\_start\_scanning.grid(row=3, column=1, pady=10)

self.btn\_check\_in\_out.grid(row=4, column=1, pady=10)

self.label\_status.grid(row=5, column=0, columnspan=3, pady=5)

self.label\_total\_time.grid(row=6, column=0, columnspan=3, pady=5)

self.btn\_generate\_pdf.grid(row=7, column=0, pady=10)

self.btn\_generate\_xlsx.grid(row=7, column=2, pady=10)

self.treeview.grid(row=8, column=0, columnspan=4, pady=10)

self.btn\_delete\_record.grid(row=9, column=1, pady=10)

# create database SQLITE3 / إنشاء قاعدة بيانات SQLITE3

def create\_table\_if\_not\_exists(self):

cursor = self.database\_connection.cursor()

cursor.execute('''CREATE TABLE IF NOT EXISTS attendance (

id INTEGER PRIMARY KEY AUTOINCREMENT,

qr\_data TEXT,

check\_in\_time TEXT,

check\_out\_time TEXT)''')

self.database\_connection.commit()

# create generate\_qr\_code make\_image/ إنشاء رمز الاستجابة السريعة جعل الصورة

def generate\_qr\_code(self):

data = self.qr\_code\_data.get()

if data:

qr = qrcode.QRCode(

version=1,

error\_correction=qrcode.constants.ERROR\_CORRECT\_L,

box\_size=10,

border=4,

)

qr.add\_data(data)

qr.make(fit=True)

qr\_code\_image = qr.make\_image(fill\_color="black", back\_color="white")

qr\_code\_image.save("generated\_qr.png")

img = Image.open("generated\_qr.png")

img = img.resize((200, 200), Image)

img = ImageTk.PhotoImage(img)

self.label\_qr\_image.config(image=img)

self.label\_qr\_image.image = img

else:

messagebox.showinfo("Error", "Please enter data for the QR code.")

# start\_scanning / بدء المسح

def start\_scanning(self):

cap = cv2.VideoCapture(0)

while True:

ret, frame = cap.read()

if not ret:

messagebox.showinfo("Error", "Failed to capture video.")

break

detector = cv2.QRCodeDetector()

data, vertices, qr\_code = detector.detectAndDecode(frame)

if data:

messagebox.showinfo("QR Code Scanned", f"Data: {data}")

self.qr\_code\_data.set(data)

self.check\_in\_out()

break

cv2.imshow("QR Code Scanner", frame)

if cv2.waitKey(1) & 0xFF == 27: # Press 'Esc' to exit MERCI LIK AZEN

break

cap.release()

cv2.destroyAllWindows()

self.refresh\_treeview()

#check\_in\_out//تسجيل الدخول والخروج

def check\_in\_out(self):

data = self.qr\_code\_data.get()

if data:

cursor = self.database\_connection.cursor()

cursor.execute("SELECT \* FROM attendance WHERE qr\_data = ? ORDER BY id DESC LIMIT 1", (data,))

result = cursor.fetchone()

if result is None or result[3] is not None:

self.check\_in()

else:

self.check\_out()

#check\_in\_out//تسجيل الدخول

def check\_in(self):

data = self.qr\_code\_data.get()

self.check\_in\_time = datetime.datetime.now().strftime('%Y-%m-%d %H:%M:%S')

cursor = self.database\_connection.cursor()

cursor.execute("INSERT INTO attendance (qr\_data, check\_in\_time) VALUES (?, ?)", (data, self.check\_in\_time))

self.database\_connection.commit()

self.label\_status.config(text=f"Status: وقت بداية العمل {self.check\_in\_time}")

self.calculate\_total\_time()

self.refresh\_treeview()

#check\_in\_out//تسجيل الخروج

def check\_out(self):

data = self.qr\_code\_data.get()

self.check\_out\_time = datetime.datetime.now().strftime('%Y-%m-%d %H:%M:%S')

cursor = self.database\_connection.cursor()

cursor.execute("UPDATE attendance SET check\_out\_time = ? WHERE qr\_data = ? AND check\_out\_time IS NULL", (self.check\_out\_time, data))

self.database\_connection.commit()

self.label\_status.config(text=f"Status: وقت المغادرة {self.check\_out\_time}")

self.calculate\_total\_time()

self.refresh\_treeview()

#calculate\_total\_time///// حساب الوقت الإجمالي

def calculate\_total\_time(self):

if self.check\_in\_time and self.check\_out\_time:

check\_in\_datetime = datetime.datetime.strptime(self.check\_in\_time, '%Y-%m-%d %H:%M:%S')

check\_out\_datetime = datetime.datetime.strptime(self.check\_out\_time, '%Y-%m-%d %H:%M:%S')

total\_time = check\_out\_datetime - check\_in\_datetime

self.total\_time\_var.set(f"Total Time: {str(total\_time)}")

def generate\_pdf\_report(self):

filename = f"attendance\_report\_{datetime.datetime.now().strftime('%Y%m%d\_%H%M%S')}.pdf"

c = canvas.Canvas(filename, pagesize=letter)

c.setFont("Helvetica", 12)

title\_text = "Attendance Report"

c.drawCentredString(letter[0] / 2, 750, title\_text)

cursor = self.database\_connection.cursor()

cursor.execute("SELECT \* FROM attendance ORDER BY id")

rows = cursor.fetchall()

table\_data = [["ID", "QR Data", "Check-In Time", "Check-Out Time"]]

for row in rows:

table\_data.append([str(row[0]), row[1], row[2], row[3]])

table = Table(table\_data)

table.setStyle(TableStyle([('BACKGROUND', (0, 0), (-1, 0), colors.grey),

('TEXTCOLOR', (0, 0), (-1, 0), colors.whitesmoke),

('ALIGN', (0, 0), (-1, -1), 'CENTER'),

('FONTNAME', (0, 0), (-1, 0), 'Helvetica-Bold'),

('BOTTOMPADDING', (0, 0), (-1, 0), 12),

('BACKGROUND', (0, 1), (-1, -1), colors.beige),

('GRID', (0, 0), (-1, -1), 1, colors.black)]))

table.wrapOn(c, 200, 400)

table.drawOn(c, 72, 600)

c.save()

messagebox.showinfo("PDF Report Generated", f"PDF Report saved as {filename}")

def generate\_xlsx\_report(self):

filename = f"attendance\_report\_{datetime.datetime.now().strftime('%Y%m%d\_%H%M%S')}.xlsx"

workbook = Workbook()

sheet = workbook.active

sheet.title = "Attendance Data"

header = ["ID", "QR Data", "Check-In Time", "Check-Out Time"]

sheet.append(header)

cursor = self.database\_connection.cursor()

cursor.execute("SELECT \* FROM attendance ORDER BY id")

rows = cursor.fetchall()

for row in rows:

sheet.append(row)

workbook.save(filename)

messagebox.showinfo("XLSX Report Generated", f"XLSX Report saved as {filename}")

def refresh\_treeview(self):

cursor = self.database\_connection.cursor()

cursor.execute("SELECT \* FROM attendance ORDER BY id")

rows = cursor.fetchall()

# Clear existing data in Treeview

for item in self.treeview.get\_children():

self.treeview.delete(item)

for row in rows:

self.treeview.insert("", "end", values=row)

def on\_treeview\_click(self, event):

selected\_item = self.treeview.selection()[0]

qr\_data = self.treeview.item(selected\_item, "values")[1]

confirmation = messagebox.askyesno("Delete Record", f"Do you want to delete the record with QR Data: {qr\_data}?")

if confirmation:

self.delete\_record()

def delete\_record(self):

selected\_item = self.treeview.selection()[0]

record\_id = self.treeview.item(selected\_item, "values")[0]

cursor = self.database\_connection.cursor()

cursor.execute("DELETE FROM attendance WHERE id = ?", (record\_id,))

self.database\_connection.commit()

messagebox.showinfo("Record Deleted", f"Record with ID {record\_id} deleted.")

self.refresh\_treeview()

if \_\_name\_\_ == "\_\_main\_\_":

root = tk.Tk()

app = AttendanceControlApp(root)

root.mainloop()