PY YOUNESS ZAINI

CONTROL ATTANDANCE

POINTEGE AUTO

fatima alaoi

2024

[Tapez l'adresse de la societe]

```
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()


#تنقاد 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()
```