ರೈ ತಾಂತ್ರಿಕ ವಿಶ್ವವಿದ್ಯಾಲಯ

Rai Technology University

(Estd. under Karnataka Act. No. 40 of 2013)

PROJECT REPORT

CAFE MANAGEMENT SYSTEM

SUBMITTED BY:

1.RAKESH C	RTU24101CS014
2.DARSHAN D	RTU24101CS015
3.MANISH M	RTU24101CS021
4.G ROSHAN ZAMEER	RTU24101CS035

Program Name: B.TECH CSE AIML

Semester: 3rd Sem

Session: 2025-2026

Submitted To:

Faculty Name: Mr. SATYAM

Department of Computer Science & Engineering College of Engineering

Rai Technology University

ರೈ ತಾಂತ್ರಿಕ ವಿಶ್ವವಿದ್ಯಾಲಯ

Rai Technology University

(Estd. under Karnataka Act. No. 40 of 2013)

College of Engineering & Application

CERTIFICATE

This is to certify that G ROSHAN ZAMEER, MANISH.M, DARSHAN.D & RAKESH.C RTU24101CS035, RTU24101CS021, RTU24101CS015 & RTU24101CS014 a Students of Rai Technology University, Bangalore, has successfully completed the Project work on "CAFÉ MANAGEMENT SYSTEM" in partial fulfilment of the requirements for the 3rd semester B. Tech program at the College of Engineering & Application, Rai Technology University,Bangalore,during the academic year 25-26

Signature of the Subject Faculty **Signature of the HOD**

ABSTRACT

This Project aims to The Cafe Management System is a develop based application developed using Python. This code uses Tkinter library for GUI buttons and to user friendly interface and also it uses SQL Lite for backend data storage. This system is designed to manage the cafe operations, order processing, bill calculations, menu management and time management.

The Primary Goal of this project is to manage the Human error and also to save the time during billing. This records the all customer transactions during billing. Users can modify the menu items, Can check the tax on their orders taken. All data is stored in lightweight database and easy to maintain and also easy to understand.

This project demonstrates the practical use of python in building real world applications for small business automation particularly in the food industry.

This the Information of our Mini Project..

TABLE OF CONTENTS

INTRODUCTION

In Today's world the food industry requires efficient and accurate systems to manage daily operations. Traditional methods of manually recording orders, calculating bills can be time consuming also human errors.

As customer expectations grow and competition increases the need for a reliable and automated management system becomes essential

This project, cafe management system using python is develop to address these challenges by offering complete digital solution without human error for small/medium sized cafes. This system is designed to calculate bills and maintaining the menus, taking orders and generate bills. It helps for customer to process the order early and takes less time than traditional methods.

This application is built using python which is beginner friendly programming language. It uses Tkinter for Graphical User Interface(GUI) and also SQLite to store the menu and also orders.

OBJECTIVE

The Main objectives of the café Management system Model are:

- **To design and develop** an AI-Based Cafe management system which can generate E-Bills Easily
- **To Consume Time** by which the orders can maintain fastly and also easily
- **To Store the data** of orders taken, money collection for each order.
- **To Provide E-Bills** as well as Menu by understanding customers needs in cafe.
- **To lay the foundation** for future enhancemets such as E-bill , E-Data and also storing menu.

FULL CODE

```
import tkinter as tk
from tkinter import messagebox
import sqlite3
from datetime import datetime

# Connect to SQLite database
conn = sqlite3.connect("cafe.db")
cursor = conn.cursor()

# Create table if it doesn't exist
cursor.execute("""
CREATE TABLE IF NOT EXISTS orders (
 id INTEGER PRIMARY KEY
AUTOINCREMENT,
    item TEXT,
    quantity INTEGER,
    total_price REAL,
```

```
    order_time TEXT
)
""")
conn.commit()

# Menu items with prices
menu = {
    "coffee": 50,
    "Tea": 30,
    "Sandwich": 80,
    "Burger": 100,
    "Ginger Tea": 0
}

# Create main application window
root = tk.Tk()
root.title("Cafe Management System")
root.geometry("500x600")
```

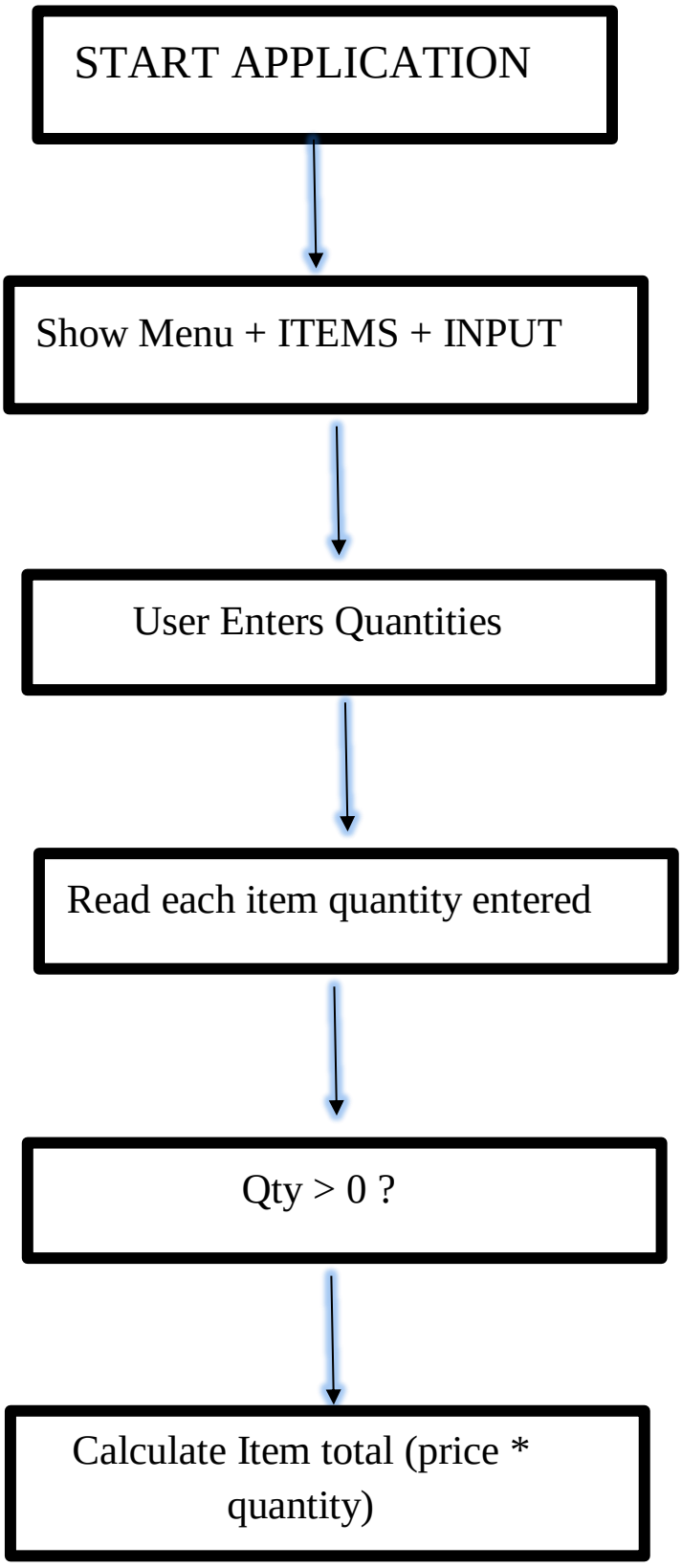
```
# Store quantity input widgets
qty_vars = {}

def calculate_bill():
    total = 0
    order_summary = ""

    for item, price in menu.items():
        qty = qty_vars[item].get()
        if qty.isdigit() and int(qty) > 0:
            item_total = int(qty) * price
            total += item_total
            order_summary += f"{item} x {qty} = ₹{item_total}\n"

            # Store in database
            cursor.execute("INSERT INTO orders (item, quantity, total_price, order_time) VALUES (?, ?, ?, ?)",
```

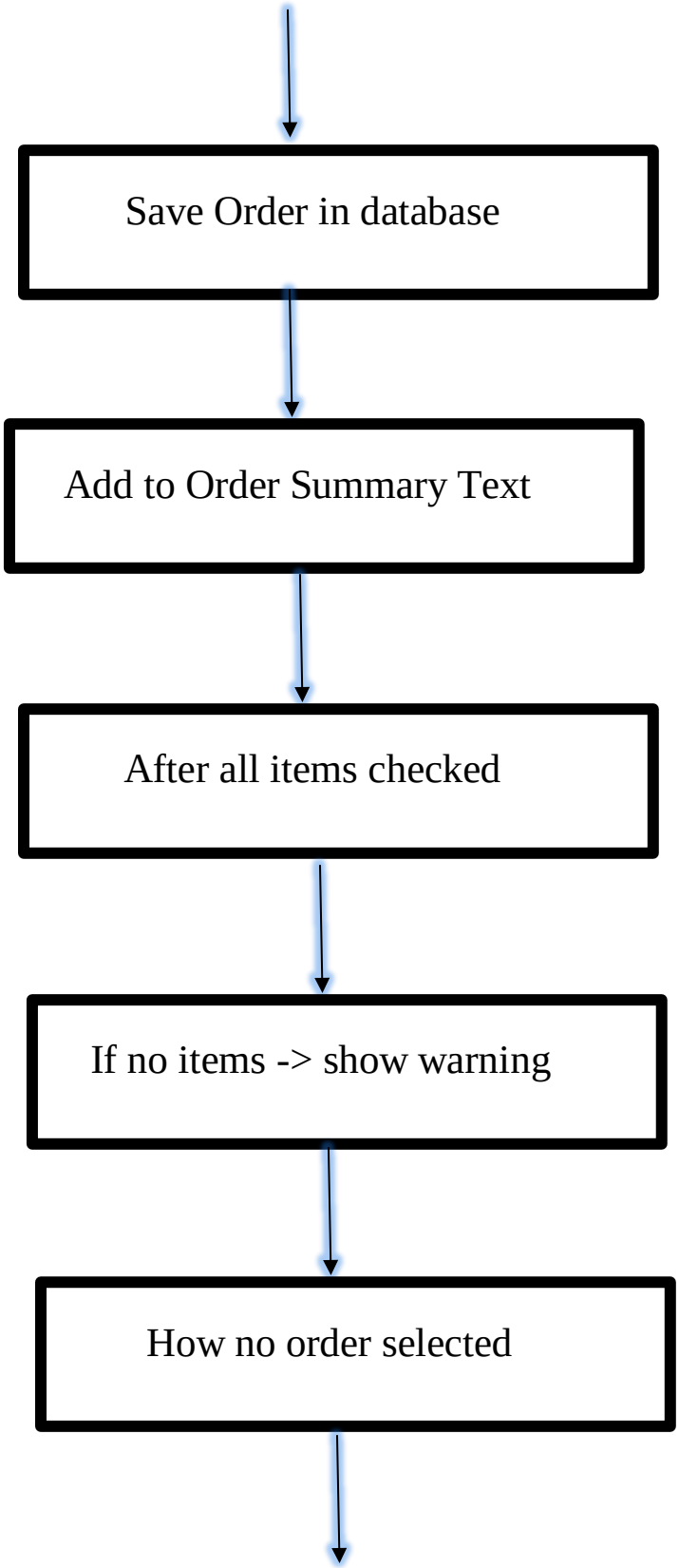
```
                    (item, int(qty), item_total,
datetime.now().strftime("%Y-%m-%d
%H:%M:%S")))
            conn.commit()


    if total == 0:
        messagebox.showwarning("No Order",
"Please select at least one item.")
        return


    tax = total * 0.05
    grand_total = total + tax
    order_summary += f"\nSubtotal:
₹{total:.2f}\nTax (5%): ₹{tax:.2f}\nTotal:
₹{grand_total:.2f}"


    messagebox.showinfo("Bill", order_summary)


def clear_fields():
    for var in qty_vars.values():
```

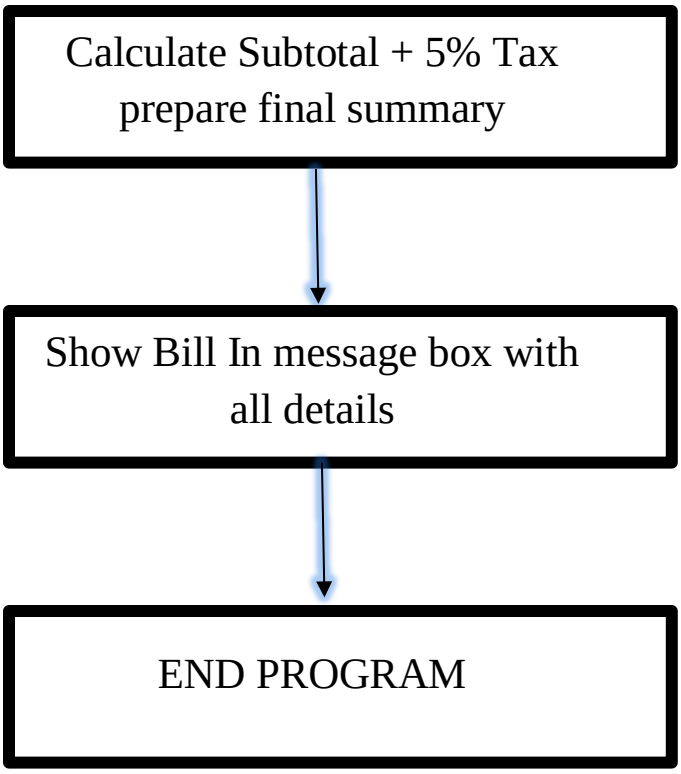
```
        var.set("0")


# UI Layout
tk.Label(root, text="Cafe Menu",
font=("Helvetica", 16, "bold")).pack(pady=10)


menu_frame = tk.Frame(root)
menu_frame.pack(pady=10)


# Generate menu rows
for i, (item, price) in enumerate(menu.items()):
    tk.Label(menu_frame, text=f"{item}
(₹{price})", font=("Helvetica", 12)).grid(row=i,
column=0, padx=10, pady=5, sticky='w')
    qty_var = tk.StringVar(value="0")
    qty_vars[item] = qty_var
    tk.Entry(menu_frame, textvariable=qty_var,
width=5).grid(row=i, column=1)
```

```
# Buttons
tk.Button(root, text="Calculate Bill",
command=calculate_bill, bg="green", fg="white",
font=("Helvetica", 12)).pack(pady=10)
tk.Button(root, text="Clear",
command=clear_fields, bg="orange", fg="white",
font=("Helvetica", 12)).pack(pady=5)
tk.Button(root, text="Exit",
command=root.destroy, bg="red", fg="white",
font=("Helvetica", 12)).pack(pady=10)


root.mainloop()
```

FLOW OF CODE


START APPLICATION
Show Menu + ITEMS + INPUT
User Enters Quantities
Read each item quantity entered
Qty > 0 ?
Calculate Item total (price * quantity)


Save Order in database
Add to Order Summary Text
After all items checked
If no items -> show warning
How no order selected



THIS SHOWS HOW THE CODE RUNS AND USE OF CODE

DESIGN AND IMPLEMENTATION

The Cafe Management System is designed using a modular and user- centric approach. The goal is to create a simple, efficient, and robust application that automates the key functionalities of a cafe, including order processing, billing, and record-keeping. The implementation utilizes the Python programming language, Tkinter for GUI design, and SQLite for local database management.

Design

The design of the Cafe Management System focuses on simplicity, ease of use, and functionality. The user interface is built using Tkinter, which provides a clean layout for menu items, quantity input fields, and billing options.

The design includes:

- A title header and welcome message
- Menu items with entry boxes to input quantity

- Buttons for generating the bill and clearing inputs
- A text area to display the bill summary
- Backend database design with a single table to store orders

Implementation

The system is implemented in Python with three main modules:

1. **GUI Module (Tkinter):** Handles the user interface, including buttons, labels, entry fields, and the layout of widgets.
2. **Billing Logic:** Calculates total price based on item quantities and predefined prices. Adds tax (e.g., 5%) and formats the bill summary.
3. **Database Integration (SQLite3):** Every transaction is recorded in a local database with the item name, quantity, total price, and timestamp.

Results and Discussion Results

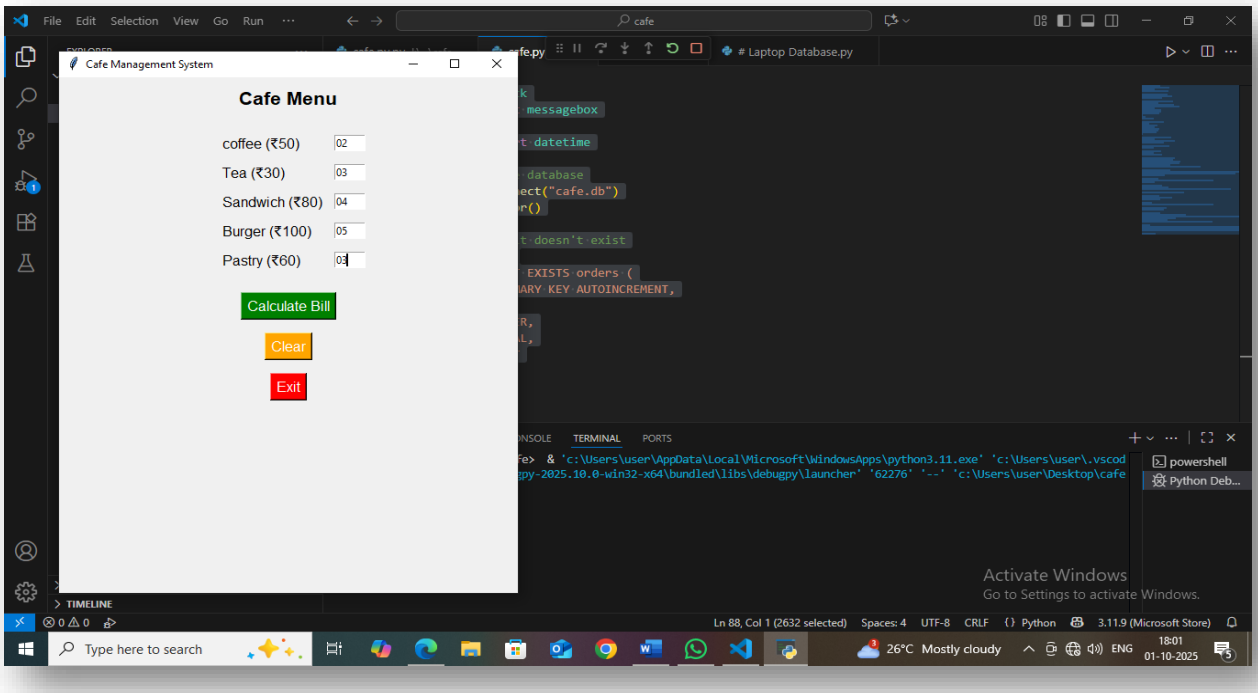
The Cafe Management System successfully performs all intended functions:

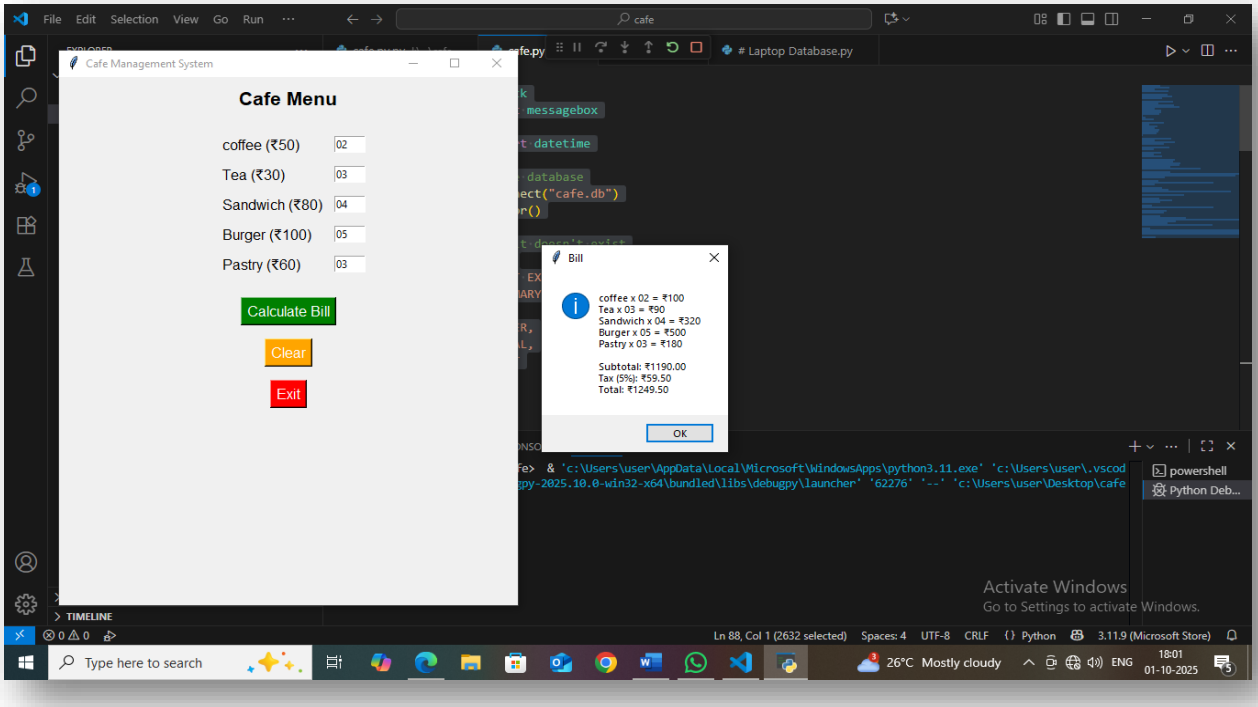
- Takes orders through a graphical user interface (GUI).
- Accurately calculates the total bill, including tax.
- Displays a clear bill summary for the customer.
- Saves order details (items, quantities, total, timestamp) in a local SQLite database.
- Prevents invalid inputs through basic error checking. The system was tested on multiple devices and produced correct outputs consistently with fast response times. The application worked smoothly with minimal resource usage.

Discussion

- The interface is simple and suitable for small cafes and food stalls.
- Using Tkinter ensures that the system remains lightweight and cross- platform.
- Storing data in SQLite allows basic record-keeping without the need for a server.
- Although effective, the system currently lacks advanced features like user login, printed receipts, and reporting tools.
- The modular code design allows easy future upgrades.

SCREENSHOTS OF OUTPUT





Conclusion

The Cafe Management System developed using Python, Tkinter, and SQLite successfully streamlines the daily operations of a small cafe. It simplifies the process of taking orders, calculating bills, and storing transaction data in a secure and efficient manner.

The system is user-friendly, cost-effective, and easy to maintain. It eliminates manual billing errors and reduces the workload on staff. With a modular and scalable design, it also provides a solid foundation for future enhancements such as inventory management, user login, and advanced reporting features.

In summary, the project meets its objectives and demonstrates how Python can be effectively used to build real-world business applications.

THANK YOU