VISVESVARAYA TECHNOLOGICAL UNIVERSITY
BELGAUM - 590014

A DBMS Mini-Project Report
On

"Recipe Management System"

A Mini-project report submitted in partial fulfilment of the requirements for the award of the Bachelor ***of Engineering in Computer Science and Engineering*** *of Visvesvaraya Technological University, Belgaum.*

Submitted by:
ARYAMAN. M(1DT20AI010)
AND
MOHAMMED ZABIULLAH .C (1DT20AI026)

Under the Guidance of:
Dr. Sandhya N (Assoc. Prof. Dept of CSE)
and
Mr. Manjunath D R (Asst. Prof. Dept of CSE)

Department of Artificial Intelligence and Machine Learning
DAYANANDA SAGAR ACADEMY OF TECHNOLOGY AND MANAGEMENT
Kanakapura Road, Udayapura, Bangalore 2019-2020
(Accredited by NBA, New Delhi for 3 years validity: 26-07-2018 to 30-06-2021)

DAYANANDA SAGAR ACADEMY OF TECHNOLOGY AND MANAGEMENT,

Kanakapura Road, Udayapura, Bangalore

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

CERTIFICATE

This is to certify that the Mini-Project on Database Management System (DBMS) titled **"RECIPE MANAGEMENT SYSTEM"** has been successfully carried out by **ARYAMAN .M (1DT20AI010) and MOHAMMED ZABIULLAH .C(1DT20AI026),** bonafide students of **Dayananda Sagar Academy of Technology and Management** in partial fulfilment of the requirements for the award of degree in **Bachelor of Engineering in Artificial Intelligence and Machine Learning** of **Visvesvaraya Technological University, Belgaum** during academic year 2022-2023. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The mini project report has been approved as it satisfies the academic requirements with respect to the project work for said degree.

GUIDES:

1. **Prof. Raghava M.S**
2. **Dr. Shivaprasad A C**

Dr. Sandhya .N
(HoD Artificial Intelligence and Machine Learning)

Examiners: **Signature with Date**

1.
2.

ACKNOWLEDGEMENT

We present before you our project titled **"RECIPE MANAGEMENT SYSTEM USING PYTHON AND MYSQL".** We express our gratitude towards our institution, **DAYANANDA SAGAR ACADEMY OF TECHNOLOGY AND MANAGEMENT** for providing us with the knowledge and support required for completing the project.

We wish to express a sincere thanks to our respected principal **Dr. M. Ravishankar** for his support.

We express our deepest gratitude and special thanks to **Dr. Sandhya .N, H.O.D, Dept. Of Artificial Intelligence and Machine Learning** for her guidance and encouragement.

We acknowledge the guidance and constant encouragement, and express our gratitude to our mini- project guides, **Prof. Raghava .M.S (Dept. of AIML) and Dr. Shivaprasad A C (Prof. Department of AIML)**

Aryaman .M(1DT20AI010)
AND
Mohammed Zabiullah .C (1DT20AI026)

ABSTRACT

The project, “RECIPE MANAGEMENT SYSTEM” is a computerised system used to add, store, view and retrieve information about the different types of recipes available in the database.

The user needs to sign up or log in. Once logged in, the user can search for a particular recipe using the search bar or can choose to explore different recipes available based on their preferences. Once the user has found what he/she was looking for, they can view the ingredients required, procedure, rating of the recipe and viewing comments posted by other users. Users can also add their own rating and/or comment for selected recipes.

The system aims to provide an easy to use recipe management system for users of all ages. Applicable in both casual and professional environments like household applications and restaurants.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Background

A recipe management system is a tool that allows users to store, organise, and access their recipe collections. These systems can be installed on a computer and typically include features such as the ability to add, delete recipes as well as view recipes added by other users categorising them by various criteria such as type of recipe(Vegetarian/Non-Vegetarian) and time of preparation(Breakfast, Lunch or Dinner).

1.2 Problem Definition

This project is aimed to reduce the inconvenience that comes with the process of using physical cookbooks for reference during cooking, some of these may be unavailability of recipes or a complicated layout and difficult instructions to follow. This project is developed to simplify the process by providing a user-friendly user interface and creating a singular application for all the recipe needs. The effective purpose of the project is to computerise the primitive use of cookbooks, make it user friendly and accessible to everyone.

1.3 Motivation

The development of recipe management systems was motivated by the desire to have access to a multitude of recipes at one's convenience**.**

- **Manual System:** The existing system is mundane and difficult to understand for beginners.
- **Technical System:** Leverage technology to make the process of cooking easy and accessible to everyone.

1.4 Objective

1. To digitise the process of accessing recipes and mitigating cookbooks.
2. To design a system that is user friendly and easy to navigate.
3. To develop a repository of recipes which is accessible at any time.
4. To allow users to explore new recipes and cuisines.

1.5 Scope of the project

The project provides an effortless system which contains a diverse variety of recipes available to the user. The upfront cost for cookbooks is extreme, detering beginners from venturing into the delightful world of cooking. We would like to provide an easy to use system which is economical as well as user friendly.

CHAPTER 2

REQUIREMENTS

The requirements for the project are broken down into two major categories, namely hardware and software requirements.

The hardware requirements specifies the minimum hardware requirements for a system running our project. The software requirements specifies the essential software needed to build and run the project.

2.1 Hardware Requirements

The system is designed to run light and is capable of running on the most basic hardware.

- Processor - Intel® Pentium® Silver N5030 Processor or equivalent.
- Processor Speed - Base frequency 1.10 max frequency up to 3.10 GHz
- System Storage - 100 GB or greater
- RAM - 4GB or greater

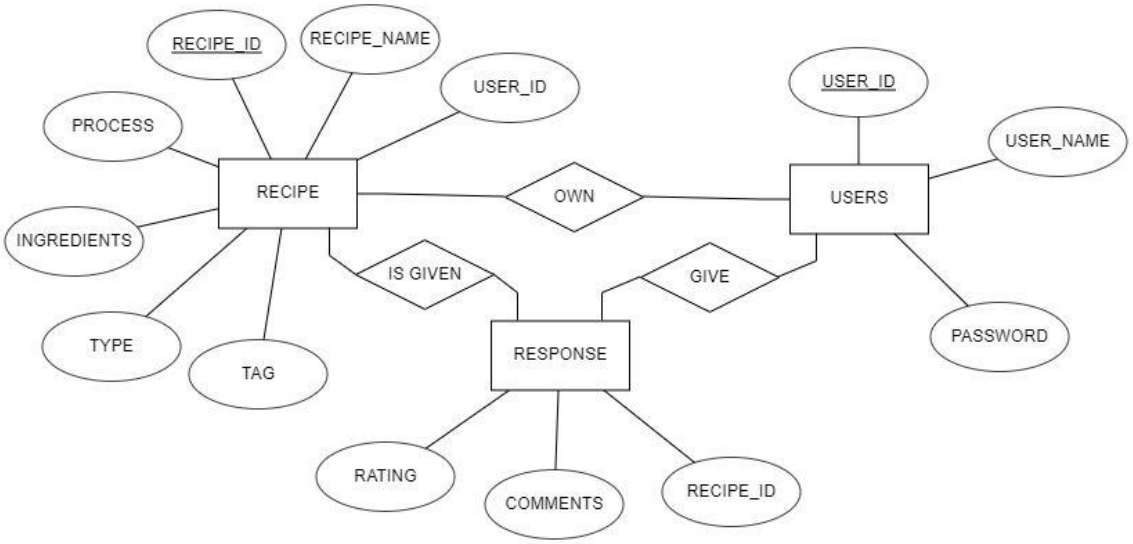
2.2 Software Requirements

- Operating System : Windows 7 or greater
- Language Used : Python
- Database : MySQL
- User Interface Design : Tkinter and CustomTkinter

CHAPTER 3

DATABASE DESIGN

3.1.1 E-R Diagram



3.1.2 Database Schema

Database:

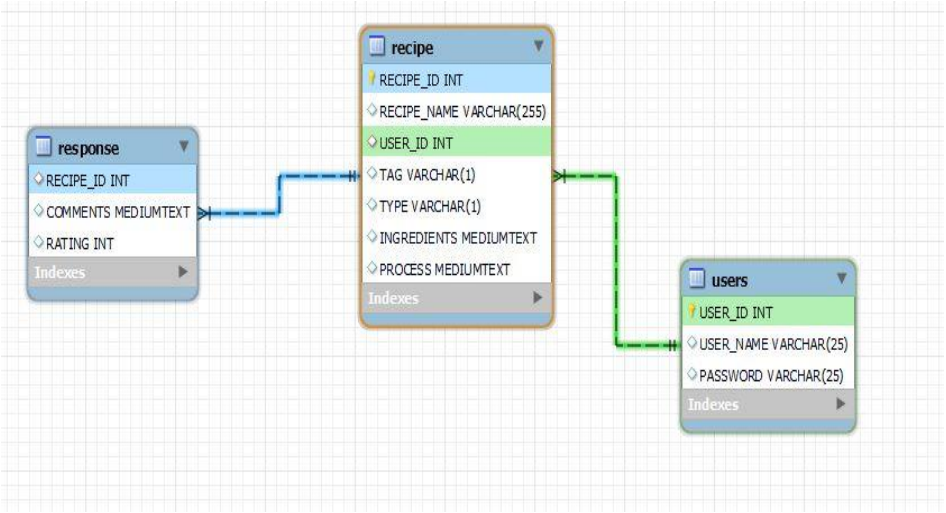


Table:Recipe

Column	Type	Default Value	Nulla
INGREDIENTS	mediumtext		YES
PROCESS	mediumtext		YES
RECIPE_ID	int		NO
RECIPE_NAME	varchar(255)		YES
TAG	varchar(1)		YES
TYPE	varchar(1)		YES
USER_ID	int		YES

Table:User

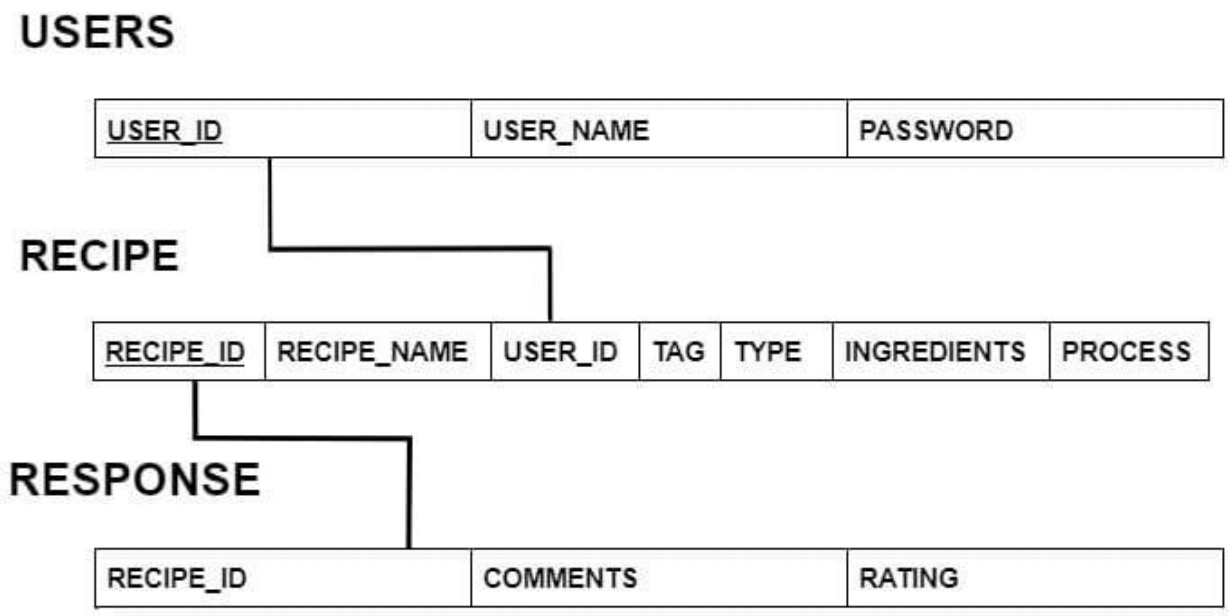
Column	Type	Default Value	Nullable
PASSWORD	varchar(25)		YES
USER_ID	int		NO
USER_NAME	varchar(25)		YES

Table : Reviews and Comments

Column	Type	Default Value	Nullable
COMMENTS	mediumtext		YES
RATING	int		YES
RECIPE_ID	int		YES

3.1.3 Relational Schema

Database:



3.2 Database Normalisation

3.2.1 First Normal Form

All the Relations are designed in such a way that it has no repeating groups. Hence all tables are in 1st Normal Form.

3.2.2 Second Normal Form

A relation is said to be in second normal form if it is already in first normal form and it has no partial dependency. All the tables in the database are designed in such a way that there is no partial dependency. Hence all tables are in 2nd Normal Form.

3.2.3 Third Normal Form

A relation is said to be in third normal form if it is already in 1st and 2nd Normal Form and has no transitive dependency. All the tables in the database are designed in such a way that there is no transitive dependency. Hence all tables are in 3rd Normal Norm.

3.3 User Interface

The User Interface of the System is divided into:

- User Module - For the users registered to post complaints
- Admin Module – For the admin/owner(users) of the CMS

3.3.1 USER REGISTRATION MODULE

3.3.1.1 User Registration

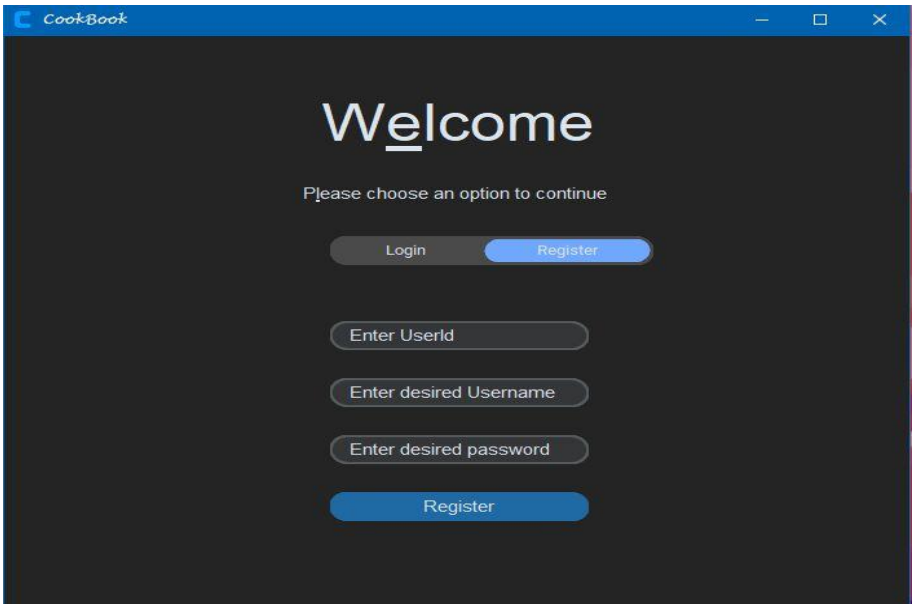


Figure 3.3.1.1: User registration module

3.3.1.2 User Login

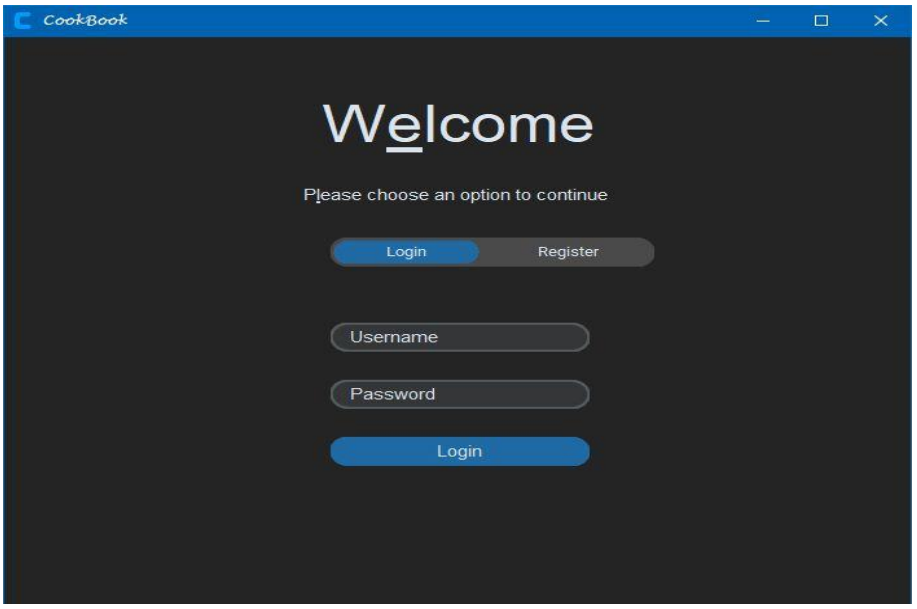


Figure 3.3.1.2: User Login

3.3.2 USER OPERATIONS MODULE

3.3.2.1 Dashboard

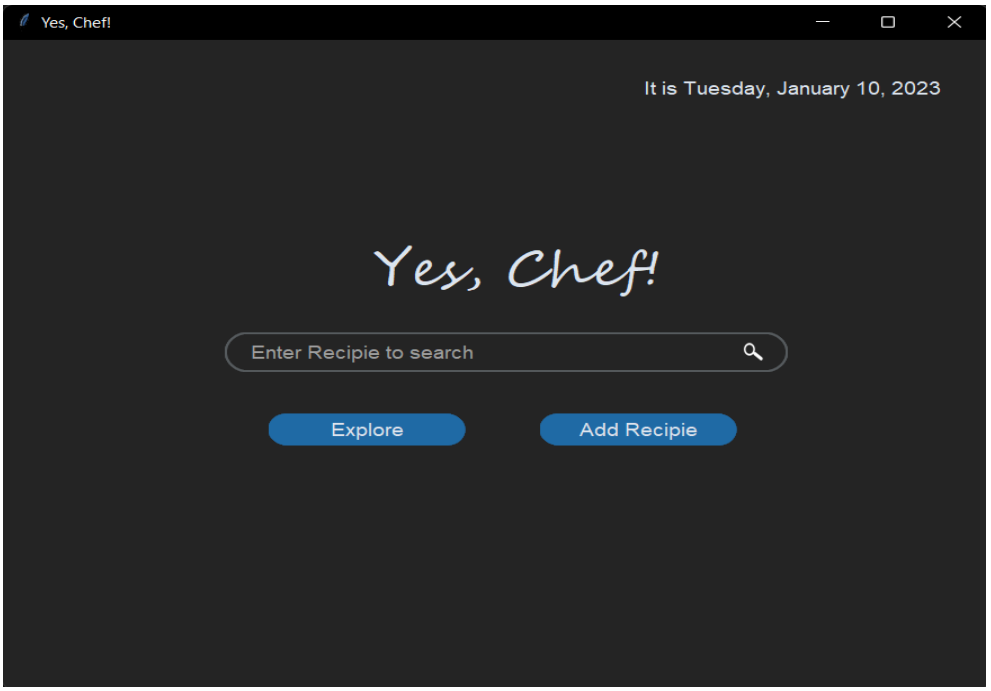


Figure 3.3.2.1: Dashboard

3.3.2.2 Explore

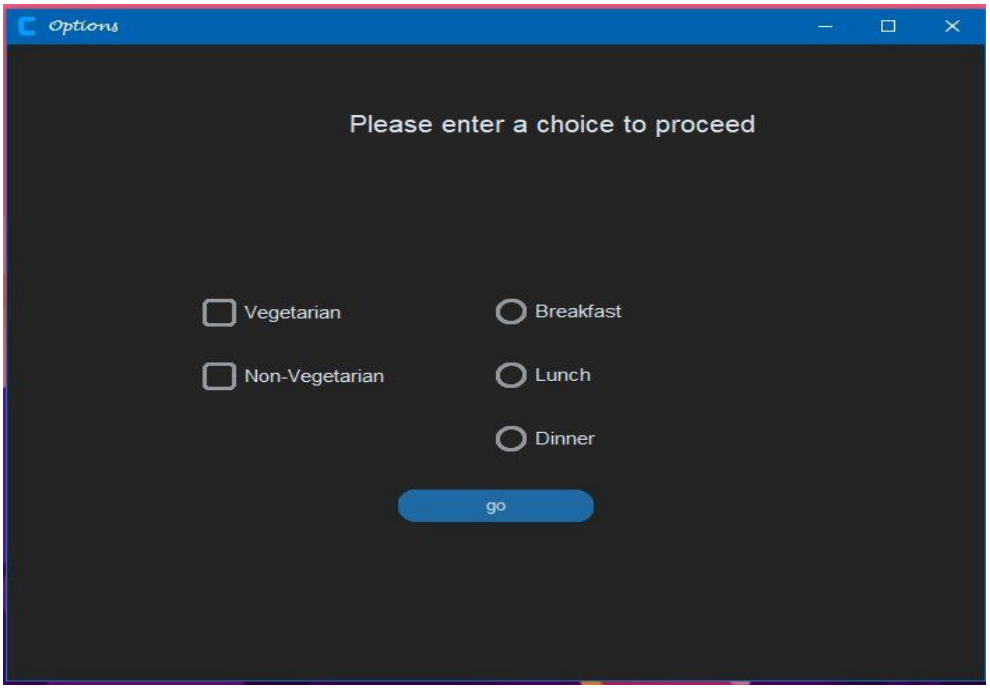


Figure 3.3.2.2: Explore

3.3.2.3 Add Recipe

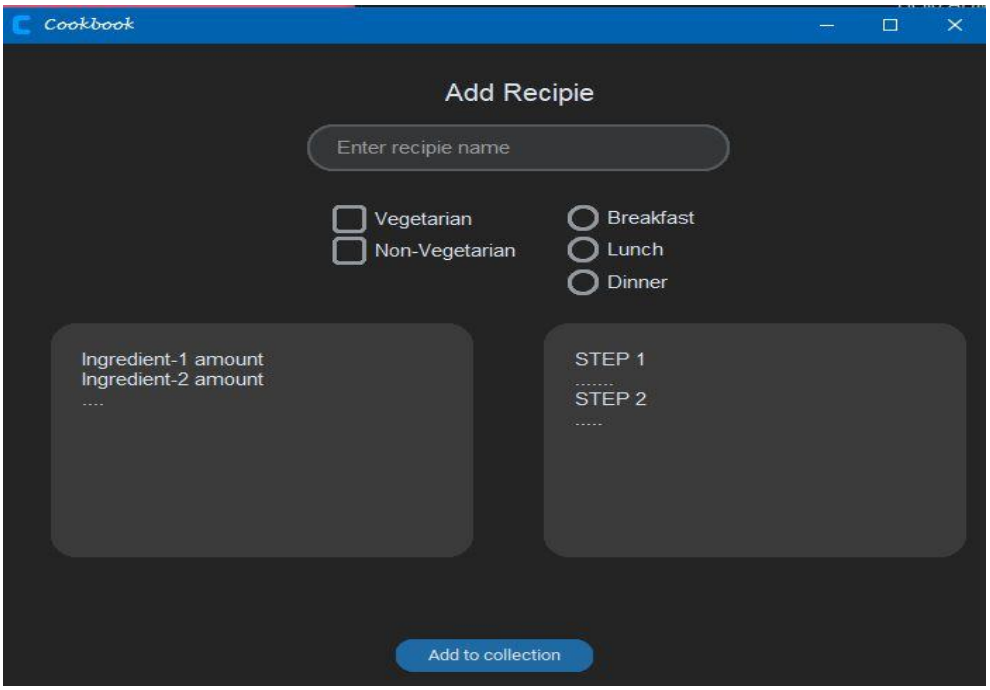


Figure 3.3.2.3: Add Recipe

3.3.2.4 Displaying Selected recipe

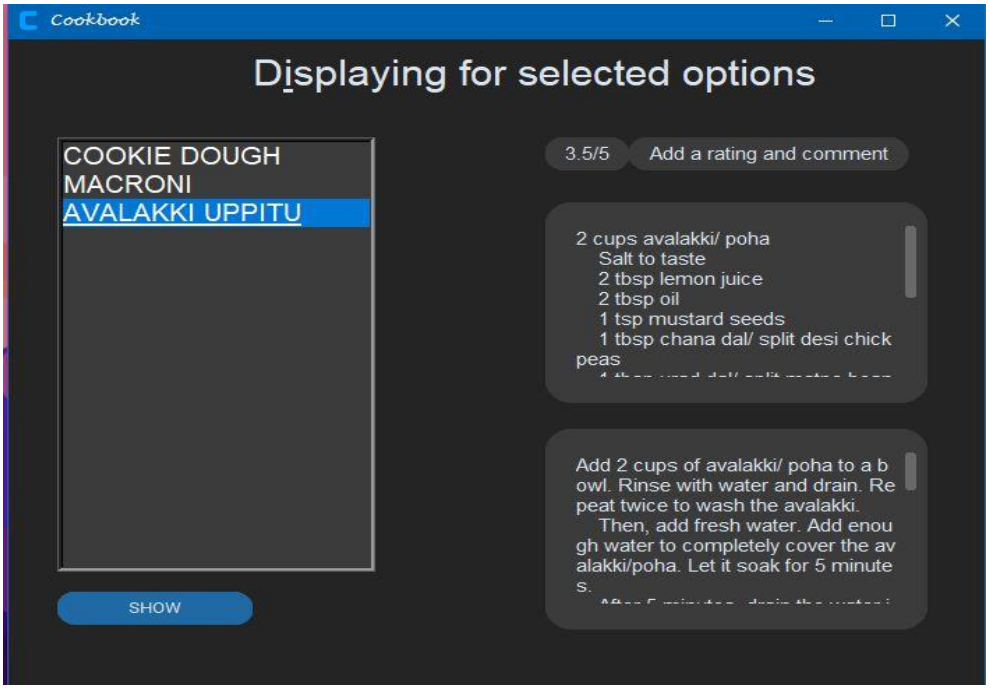


Figure 3.3.2.4: Displaying Recipe

3.3.2.5 Adding Rating and Comments

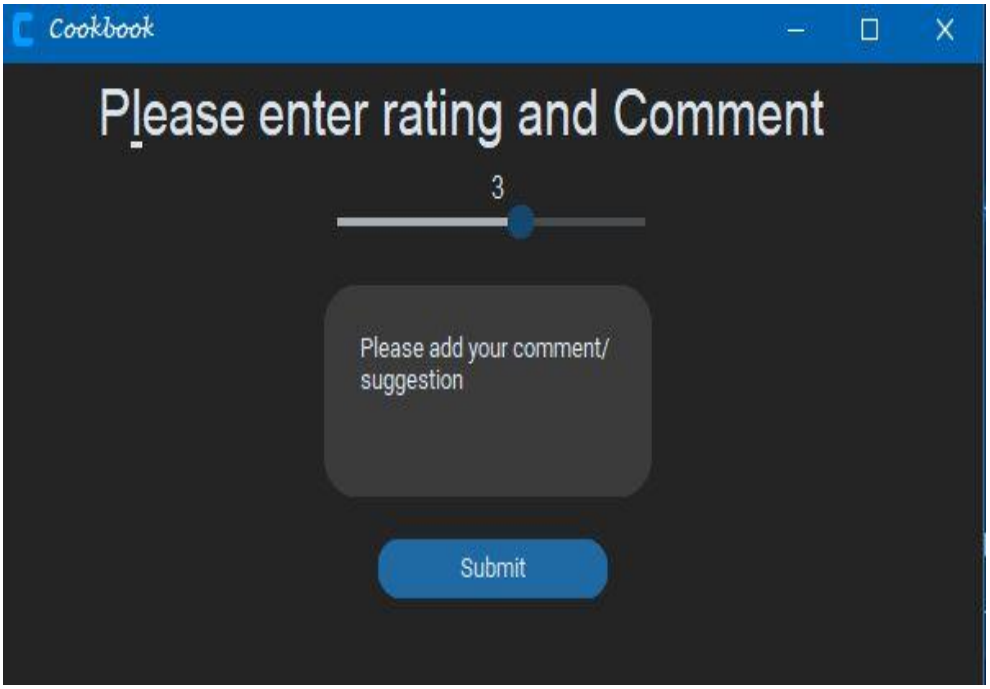


Figure 3.3.2.5: Adding rating and Comment

CHAPTER 4

IMPLEMENTATION

4.1.1 User Registration Module

Process Name	• User Registration
Process Number	• 1.1
Input	• User_id • User_name • User_password
Output	• Successfully created user profile
Error Condition	• User_name already exists • All fields are mandatory

4.1.2 User Login Module

Process Name	• User Login
Process Number	• 1.2
Input	• User_name • User_password
Output	• Successfully Logged In
Error Condition	• User_name or User_password is incorrect • All fields are mandatory

4.1.3 User Dashboard

Process Name	• Dashboard Display
Process Number	• 2.1
Input	• Search • Explore • Add Recipe
Error Condition	• Recipe not available. Contact admin to add recipe

4.1.4 Explore

Process Name	• Explore
Process Number	• 2.2
Input	• Vegetarian • Non_vegetarian • Breakfast • Lunch • Dinner
Output	• Displays the Recipes fulfilling the criteria
Error Condition	• Please select one field at least

4.1.5 Add Recipe

Process Name	• Add Recipe
Process Number	• 2.3
Input	• Recipe_name • Vegetarian/Non-Vegetarian • Breakfast/Lunch/Dinner • Ingredients • Procedure

Output	● Successfully created recipe. Available to view.
Error Condition	● Please enter all the fields

4.1.6 Displaying Selected recipe

Process Name	● Displaying Selected Recipe
Process Number	● 3.1
Input	● Add comment and rating
Output	● Recipe name ● Ingredients ● Procedure ● Rating

4.1.7 Adding Rating and Comment

Process Name	● Adding Rating and Comment
Process Number	● 3.2
Input	● Rating ● Comment
Output	● Thank you for your input

4.2 SOURCE CODE

Sample Source code to view food items:

```
import datetime as dt
from PIL import Image
import mysql.connector
import tkinter as tk
from customtkinter import *
# DATABASE CONNECTION AND INTERFACING

# Establishing initioal connection
db = mysql.connector.connect(
    host='localhost',
    user='sqluser',
    password='password',
)

# Creating database
mc = db.cursor()
mc.execute("DROP DATABASE IF EXISTS RECIPES")
mc.execute("CREATE DATABASE IF NOT EXISTS RECIPES")
mc.close()
db.disconnect()

# Connecting to database
db = mysql.connector.connect(
    host='localhost',
    user='sqluser',
    password='password',
    database="RECIPES"
)

mc = db.cursor()


def database():

    # DATABASE CONNECTION AND INTERFACING

    # Establishing initioal connection
    db = mysql.connector.connect(
        host='localhost',
        user='sqluser',
        password='password',
    )

    # Creating database
    mc = db.cursor()
    mc.execute("DROP DATABASE IF EXISTS RECIPES")
    mc.execute("CREATE DATABASE IF NOT EXISTS RECIPES")
    mc.close()
    db.disconnect()

    # Connecting to database
    db = mysql.connector.connect(
        host='localhost',
        user='sqluser',
        password='password',
        database="RECIPES"
    )

    mc = db.cursor()
    mc.execute(
        "CREATE TABLE IF NOT EXISTS USERS(USER_ID INT PRIMARY KEY,USER_NAME VARCHAR(25),PASSWORD
VARCHAR(25)) ")
```

```
    mc.execute("CREATE TABLE IF NOT EXISTS RECIPE(RECIPE_ID INT AUTO_INCREMENT PRIMARY KEY,RECIPE_NAME
VARCHAR(255),USER_ID INT,TAG VARCHAR(1),TYPE VARCHAR(1),INGREDIENTS MEDIUMTEXT ,PROCESS
MEDIUMTEXT,FOREIGN KEY(USER_ID) REFERENCES USERS(USER_ID))")

    mc.execute("CREATE TABLE IF NOT EXISTS RESPONSE(RECIPE_ID INT, COMMENTS MEDIUMTEXT, RATING INT,
FOREIGN KEY(RECIPE_ID) REFERENCES RECIPE(RECIPE_ID))")

    mc.execute(
        "INSERT INTO USERS(USER_ID,USER_NAME,PASSWORD) VALUES(0,'ADMIN','1234')")

    print("inserting admin")
    """
    1       COOKIE DOUGH
    2       MACRONI
    3       CHITRANA
    4       CURD RICE
    5       RAGI MUDDE
    6       Meen Pollichathu
    7       Avalakki Uppittu
    """

    mc.execute(
        "INSERT INTO RESPONSE(RECIPE_ID,COMMENTS,RATING) VALUES(1,'YUMMM',5)")
    mc.execute(
        "INSERT INTO RESPONSE(RECIPE_ID,COMMENTS,RATING) VALUES(1,'NOICE',4)")

    mc.execute(
        "INSERT INTO RESPONSE(RECIPE_ID,COMMENTS,RATING) VALUES(2,'NOICE',5)")
    mc.execute(
        "INSERT INTO RESPONSE(RECIPE_ID,COMMENTS,RATING) VALUES(2,'TASTYYYY',4)")

    mc.execute(
        "INSERT INTO RESPONSE(RECIPE_ID,COMMENTS,RATING) VALUES(3,'SUPPERB',4)")
    mc.execute(
        "INSERT INTO RESPONSE(RECIPE_ID,COMMENTS,RATING) VALUES(3,'YUMMMM',3)")

    mc.execute(
        "INSERT INTO RESPONSE(RECIPE_ID,COMMENTS,RATING) VALUES(4,'BESSSTT',3)")
    mc.execute(
        "INSERT INTO RESPONSE(RECIPE_ID,COMMENTS,RATING) VALUES(4,'TASTYYYY',4)")

    mc.execute(
        "INSERT INTO RESPONSE(RECIPE_ID,COMMENTS,RATING) VALUES(5,'NOICE',3)")
    mc.execute(
        "INSERT INTO RESPONSE(RECIPE_ID,COMMENTS,RATING) VALUES(5,'SUPPERB',4)")

    mc.execute(
        "INSERT INTO RESPONSE(RECIPE_ID,COMMENTS,RATING) VALUES(6,'DELICIOUSSS',3)")
    mc.execute(
        "INSERT INTO RESPONSE(RECIPE_ID,COMMENTS,RATING) VALUES(6,'YUMMMMM',4)")

    mc.execute(
        "INSERT INTO RESPONSE(RECIPE_ID,COMMENTS,RATING) VALUES(7,'BESSTT',3)")
    mc.execute(
        "INSERT INTO RESPONSE(RECIPE_ID,COMMENTS,RATING) VALUES(7,'SUPPERB',4)")

    db.commit()


database()


def login(name, password):
    p, ID = '', 0
    mc.execute(f"SELECT PASSWORD,USER_ID FROM USERS WHERE USER_NAME='{name}'")
    for i in mc:
        p, ID = i[0], i[1]
    if p == password:
        return True, ID
    else:
```

```
        return False, ID


def register(i, name, password):
    q = "INSERT INTO USERS(USER_ID,USER_NAME,PASSWORD) VALUES(%s,%s,%s)"
    val = (i, name, password)
    try:
        mc.execute(q, val)
    except mysql.connector.errors.IntegrityError:
        return "user id already taken"
    except:
        return "Something went wrong"
    else:
        db.commit()
        return "User registered successfully"


def segemented_button_callback(value):
    if (value == "Login"):
        label.configure(text="")
        return login_select()
    else:
        label.configure(text="")
        return register_select()


def login_clicked():
    val, i = login(str(login_user_name.get()).upper(),
                   login_password.get())
    if val:
        label.configure(text="Logged in succesfully")
        LOGIN.destroy()
        explore(i)
    else:
        label.configure(text="Incorrect data")


def register_clicked():
    val = register(register_user_id.get(), register_user_name.get(
    ).upper(), register_user_password.get())
    label.configure(text=val)


def login_select():
    global login_user_name
    login_user_name = StringVar(value="Username")
    global login_password
    login_password = StringVar(value="Password")

    lu = CTkEntry(
        LOGIN, font=general_font, corner_radius=20, width=200, textvariable=login_user_name)
    lu.bind("<FocusIn>", lambda e: login_user_name.set(""))
    lu.place(relx=.36, rely=.5)

    lp = CTkEntry(
        LOGIN, font=general_font, corner_radius=20, width=200, textvariable=login_password)
    lp.place(relx=.36, rely=.6)
    lp.bind("<FocusIn>", lambda e: login_password.set(""))

    CTkButton(
        LOGIN, text="Login", font=general_font, corner_radius=20, width=200, command=login_clicked).place(relx=.36, rely=.7)


def register_select():
    global register_user_id
    register_user_id = StringVar(value="Enter UserId")
    global register_user_name
    register_user_name = StringVar(value="Enter desired Username")
    global register_user_password
    register_user_password = StringVar(value="Enter desired password")
```

```
    rui = CTkEntry(
        LOGIN, font=general_font, corner_radius=20, width=200, textvariable=register_user_id)
    rui.place(relx=.36, rely=.5)
    rui.bind("<FocusIn>", lambda e: register_user_id.set(""))

    ru = CTkEntry(
        LOGIN, font=general_font, corner_radius=20, width=200, textvariable=register_user_name)
    ru.place(relx=.36, rely=.6)
    ru.bind("<FocusIn>", lambda e: register_user_name.set(""))

    rp = CTkEntry(
        LOGIN, font=general_font, corner_radius=20, width=200, textvariable=register_user_password)
    rp.place(relx=.36, rely=.7)
    rp.bind("<FocusIn>", lambda e: register_user_password.set(""))
    CTkButton(
        LOGIN, text="Register", font=general_font, corner_radius=20, width=200, command=register_clicked).place(relx=.36, rely=.8)


def explore_button_click():
    Explore_choice = CTk()
    Explore_choice.geometry("700x550")
    Explore_choice.title("Options")
    veg = BooleanVar(Explore_choice)
    nonveg = BooleanVar(Explore_choice)
    radio_var = IntVar(Explore_choice)
    general_font = CTkFont(family="Sans-serif", size=15)
    display_font = CTkFont(family="Sans-serif", size=20)
    CTkLabel(Explore_choice, text="Please enter a choice to proceed",
             font=display_font).place(relx=.35, rely=.10)
    CTkCheckBox(Explore_choice, text="Vegetarian", font=general_font,
                variable=veg, onvalue=True).place(relx=.2, rely=.4)
    CTkCheckBox(Explore_choice, text="Non-Vegetarian", font=general_font,
                variable=nonveg, onvalue=True).place(relx=.2, rely=.5)
    CTkRadioButton(Explore_choice, text="Breakfast", variable=radio_var,
                   value=1, font=general_font).place(relx=.50, rely=.40)
    CTkRadioButton(Explore_choice, text="Lunch", variable=radio_var,
                   value=2, font=general_font).place(relx=.50, rely=.50)
    CTkRadioButton(Explore_choice, text="Dinner", variable=radio_var,
                   value=3, font=general_font).place(relx=.50, rely=.60)

    def submit_button_click():
        v1, v2, v3 = veg.get(), nonveg.get(), radio_var.get()
        if not v1 and not v2:
            pass
        elif v3 not in [1, 2, 3]:
            pass
        Display_recipe = CTk()
        Display_recipe.geometry("700x550")
        Display_recipe.title("Cookbook")
        general_font = CTkFont(family="Sans-serif", size=15)
        display_font = CTkFont(family="Sans-serif", size=30)
        greeting = CTkLabel(
            Display_recipe, text="Displaying for selected options", font=display_font, underline=True).place(relx=0.25, rely=0.02)

        def show_clicked():
            mc.execute(
                f"SELECT INGREDIENTS,PROCESS,RECIPE_ID FROM RECIPE WHERE RECIPE_NAME = '{check.get(ANCHOR)}'")
            for i in mc:
                ingredients, process, recipe_id = i
            mc.execute(
                f"SELECT RATING FROM RESPONSE WHERE RECIPE_ID = '{recipe_id}'")
            avg_rating = []
            for i in mc:
                avg_rating.append(i[0])
            avg_rating = sum(avg_rating)/len(avg_rating)
            recipi_ingridients.configure(state='normal')
            recipi_process.configure(state="normal")
            recipi_ingridients.delete("0.0", END)
            recipi_process.delete("0.0", END)
            rating.configure(text=f"{avg_rating}/5")
```

```
    recipi_process.insert("0.0", process)
    recipi_ingridients.insert("0.0", ingredients)
    recipi_ingridients.configure(state='disabled')
    recipi_process.configure(state="disabled")

def comment_rating():
    response = CTk()
    response.geometry("600x300")
    response.title("Cookbook")
    welcome_font = CTkFont(family="Sans-serif", size=50)
    general_font = CTkFont(family="Sans-serif", size=15)
    display_font = CTkFont(family="Sans_serif", size=30)
    greeting = CTkLabel(
        response, text="Please enter rating and Comment", font=display_font, underline=True).place(relx=0.1, rely=0.02)

    def slider_event(value):
        rating = CTkLabel(response, text=int(
            value), font=general_font).place(relx=.50, rely=.15)

    def submit_clicked():
        comment = textbox.get("0.0", END)
        rating = int(rating_slider.get())
        recipe_name = check.get(ANCHOR)
        mc.execute(
            f"SELECT RECIPE_ID FROM RECIPE WHERE RECIPE_NAME = '{recipe_name}'")
        for i in mc:
            recipe_id = i[0]
        print(recipe_id)
        mc.execute(
            f"INSERT INTO RESPONSE(RECIPE_ID,COMMENTS,RATING) VALUES({recipe_id},'{comment}',{rating})")
        db.commit()
        response.destroy()

    rating_slider = CTkSlider(
        response, from_=0, to=5, number_of_steps=5, command=slider_event)
    rating_slider.place(relx=0.5, rely=0.25, anchor=tk.CENTER)
    submit = CTkButton(
        response, text="Submit", corner_radius=20, command=submit_clicked).place(relx=.385, rely=.75)
    textbox = CTkTextbox(
        response, width=200, height=100, corner_radius=20)
    textbox.place(relx=.33, rely=.35)
    textbox.bind("<FocusIn>", lambda e: textbox.delete("0.0", "end"))
    # insert at line 0 character 0
    textbox.insert("0.0", "Please add your comment/suggestion")
    # get text from line 0 character 0 till the end
    text = textbox.get("0.0", "end")
    response.mainloop()

show = CTkButton(
    Display_recipe, text="SHOW", corner_radius=20, command=show_clicked)
show.place(relx=.05, rely=.85)

comment_rate = CTkButton(Display_recipe, text="Add a rating and comment", corner_radius=20,
                         fg_color='#3b3b3b', font=general_font, command=comment_rating, width=10).place(relx=.635, rely=.15)
rating = CTkLabel(Display_recipe, text=f"-/5", corner_radius=20, width=50,
                  fg_color='#3b3b3b', font=general_font)
rating.place(relx=.55, rely=.15)
recipi_ingridients = CTkTextbox(
    Display_recipe, width=275, height=170, corner_radius=20, font=general_font)
recipi_ingridients.place(relx=.55, rely=.25)
recipi_ingridients.insert("0.0", "Ingredients....")
recipi_ingridients.configure(state="disabled")

recipi_process = CTkTextbox(
    Display_recipe, width=275, height=170, corner_radius=20, font=general_font)
recipi_process.place(relx=.55, rely=.60)
recipi_process.insert("0.0", "Process....")
recipi_process.configure(state="disabled")

check = tk.Listbox(Display_recipe, bg='#3b3b3b', fg='white', height=15, font=(
```

```
        'Sans-serif', 15), bd=3)
    check.place(relx=.05, rely=.15)

    def add_check_list_filter(tag, type):
        if len(tag) == 2:
            mc.execute(
                f"SELECT RECIPE_NAME FROM RECIPE WHERE (TAG = '{tag[0]}' OR TAG = '{tag[1]}') AND TYPE = '{type}'")
            result_list = []
            for i in mc:
                result_list.append(i[0])
            for i in result_list:
                check.insert(END, i)
        else:
            mc.execute(
                f"SELECT RECIPE_NAME FROM RECIPE WHERE TAG ='{tag[0]}' AND TYPE = '{type}'")
            result_list = []
            for i in mc:
                result_list.append(i[0])
            for i in result_list:
                check.insert(END, i)

    if v1 and v2 and v3 == 1:
        add_check_list_filter(['V', 'N'], 'B')
    elif v1 and v2 and v3 == 2:
        add_check_list_filter(['V', 'N'], 'L')
    elif v1 and v2 and v3 == 3:
        add_check_list_filter(['V', 'N'], 'D')
    elif v1 and v3 == 1:
        add_check_list_filter(['V'], 'B')
    elif v1 and v3 == 2:
        add_check_list_filter(['V'], 'L')
    elif v1 and v3 == 3:
        add_check_list_filter(['V'], 'D')
    elif v2 and v3 == 1:
        add_check_list_filter(['N'], 'B')
    elif v2 and v3 == 2:
        add_check_list_filter(['N'], 'L')
    elif v2 and v3 == 3:
        add_check_list_filter(['N'], 'D')
    else:
        pass
    Explore_choice.destroy()
    Display_recipe.mainloop()
CTkButton(Explore_choice, text="go", corner_radius=30,
          command=submit_button_click).place(relx=.40, rely=.70)
Explore_choice.mainloop()


def add_button_clicked(user_id):
    Display_recipe_search = CTk()
    v = BooleanVar()
    nv = BooleanVar()
    rv = IntVar()

    def add_to_collection(recipe_name, tag, type, ingredients, process):
        mc.execute(
            f"INSERT INTO RECIPE(USER_ID,RECIPE_NAME,TAG,TYPE,INGREDIENTS,PROCESS)
VALUES({user_id},'{recipe_name}','{tag}','{type}','{ingredients}','{process}' )")
        mc.execute(
            f"SELECT RECIPE_ID FROM RECIPE WHERE RECIPE_NAME = '{recipe_name}'")
        for i in mc:
            recipe_id = i[0]
        mc.execute(
            f"INSERT INTO RESPONSE(RECIPE_ID,COMMENTS,RATING) VALUES({recipe_id},'NOICE',3)")
        db.commit()
        Display_recipe_search.destroy()

    def gobutton_clicked():
        v1, v2, v3, ingredients, process, recipe_name = v.get(), nv.get(), rv.get(
        ), ingridients_name.get("0.0", END), procedure.get("0.0", END), recipi_name.get()
```

```
        if v1 and v3 == 1:
            add_to_collection(recipe_name, 'V', 'B', ingredients, process)
        elif v1 and v3 == 2:
            add_to_collection(recipe_name, 'V', 'L', ingredients, process)
        elif v1 and v3 == 3:
            add_to_collection(recipe_name, 'V', 'D', ingredients, process)
        elif v2 and v3 == 1:
            add_to_collection(recipe_name, 'N', 'B', ingredients, process)
        elif v2 and v3 == 2:
            add_to_collection(recipe_name, 'N', 'L', ingredients, process)
        elif v2 and v3 == 3:
            add_to_collection(recipe_name, 'N', 'D', ingredients, process)

        print(f"{v1}\n{v2}\n{v3}\n{ingredients}\n{process}\n{recipe_name} ")

    general_font = CTkFont(family="Sans-serif", size=15)
    display_font = CTkFont(family="Sans-serif", size=20)
    Add_display = CTkLabel(
        Display_recipe_search, text="Add Recipie", font=display_font).place(relx=.45, rely=.05)
    recipi_name = CTkEntry(Display_recipe_search, placeholder_text="Enter recipie name",
                           font=general_font, corner_radius=30, width=300, height=40)
    recipi_name.place(relx=.31, rely=.125)
    vegetarian = CTkCheckBox(Display_recipe_search, text="Vegetarian", font=general_font,
                             variable=v, onvalue=True, hover_color="green").place(relx=.335, rely=.25)
    non_vegetarian = CTkCheckBox(Display_recipe_search, text="Non-Vegetarian", font=general_font,
                                 variable=nv, onvalue=True, hover_color="red").place(relx=.335, rely=.30)
    Breakfast = CTkRadioButton(
        Display_recipe_search, text="Breakfast", variable=rv, value=1, font=general_font).place(relx=.575, rely=.25)
    lunch = CTkRadioButton(
        Display_recipe_search, text="Lunch", variable=rv, value=2, font=general_font).place(relx=.575, rely=.3)
    dinner = CTkRadioButton(
        Display_recipe_search, text="Dinner", variable=rv, value=3, font=general_font).place(relx=.575, rely=.35)
    ingridients_name = CTkTextbox(
        Display_recipe_search, width=300, height=200, corner_radius=20, font=general_font)
    ingridients_name.place(relx=.05, rely=.435)
    ingridients_name.insert(
        "0.0", "Ingredient-1 amount\nIngredient-2 amount\n....")
    ingridients_name.bind(
        "<FocusIn>", lambda e: ingridients_name.delete("0.0", "end"))
    procedure = CTkTextbox(
        Display_recipe_search, width=300, height=200, corner_radius=20, font=general_font)
    procedure.place(relx=.55, rely=.435)
    procedure.insert("0.0", "STEP 1\n.......\nSTEP 2\n.....")
    procedure.bind("<FocusIn>", lambda e: procedure.delete("0.0", "end"))
    go_button = CTkButton(
        Display_recipe_search, text="Add to collection", corner_radius=30, command=gobutton_clicked).place(relx=.40, rely=.925)
    Display_recipe_search.geometry("700x550")
    Display_recipe_search.title("Cookbook")
    Display_recipe_search.mainloop()


def display_recipe_search():
    Display_recipe_search = CTk()
    Display_recipe_search.geometry("700x550")
    Display_recipe_search.title("Cookbook")
    general_font = CTkFont(family="Sans-serif", size=15)
    display_font = CTkFont(family="Sans-serif", size=30)

    rating = CTkLabel(Display_recipe_search, text=f"-/5", corner_radius=20, width=50,
                      fg_color='#3b3b3b', font=general_font)
    rating.place(relx=.55, rely=.15)
    recipi_ingridients = CTkTextbox(
        Display_recipe_search, width=265, height=150, corner_radius=20, font=general_font)
    recipi_ingridients.place(relx=.55, rely=.25)
    recipi_ingridients.insert("0.0", "Ingredients....")
    recipi_ingridients.configure(state="disabled")

    recipi_process = CTkTextbox(
        Display_recipe_search, width=265, height=150, corner_radius=20, font=general_font)
    recipi_process.place(relx=.55, rely=.60)
```

```
recipi_process.insert("0.0", "Process....")
recipi_process.configure(state="disabled")

check = tk.Listbox(Display_recipe_search, bg='#3b3b3b', fg='white', height=15, font=(
    'Sans-serif', 15), bd=3)
check.place(relx=.05, rely=.15)
recipe_list = []

mc.execute(
    f"SELECT RECIPE_NAME FROM RECIPE WHERE RECIPE_NAME LIKE '%{str(search_bar.get()).upper()}%'")
for i in mc:
    recipe_list.append(i[0])
if len(recipe_list) == 0:
    suggestion.configure(
        text="No such recipe found, Click on add recipe to your recipe")
    return
print(recipe_list)
for i in recipe_list:
    check.insert(END, i)

greeting = CTkLabel(
    Display_recipe_search, text="Displaying matched recipes", font=display_font, underline=True).place(relx=0.25, rely=0.02)

def show_clicked():
    mc.execute(
        f"SELECT INGREDIENTS,PROCESS,RECIPE_ID FROM RECIPE WHERE RECIPE_NAME = '{check.get(ANCHOR)}'")
    for i in mc:
        ingredients, process, recipe_id = i
    mc.execute(
        f"SELECT RATING FROM RESPONSE WHERE RECIPE_ID = '{recipe_id}'")
    avg_rating = []
    for i in mc:
        avg_rating.append(i[0])
    avg_rating = sum(avg_rating)/len(avg_rating)
    recipi_ingridients.configure(state='normal')
    recipi_process.configure(state="normal")
    recipi_ingridients.delete("0.0", END)
    recipi_process.delete("0.0", END)
    rating.configure(text=f"{avg_rating}/5")
    recipi_process.insert("0.0", process)
    recipi_ingridients.insert("0.0", ingredients)
    recipi_ingridients.configure(state='disabled')
    recipi_process.configure(state="disabled")

show = CTkButton(
    Display_recipe_search, text="SHOW", corner_radius=20, command=show_clicked)
show.place(relx=.05, rely=.85)

def comment_rating():
    response = CTk()
    response.geometry("600x300")
    response.title("Cookbook")
    welcome_font = CTkFont(family="Sans-serif", size=50)
    general_font = CTkFont(family="Sans-serif", size=15)
    display_font = CTkFont(family="Sans_serif", size=30)
    greeting = CTkLabel(
        response, text="Please enter rating and Comment", font=display_font, underline=True).place(relx=0.1, rely=0.02)

    def slider_event(value):
        rating = CTkLabel(response, text=int(
            value), font=general_font).place(relx=.50, rely=.15)

    def submit_clicked():
        comment = textbox.get("0.0", END)
        rating = int(rating_slider.get())
        recipe_name = check.get(ANCHOR)
        mc.execute(
            f"SELECT RECIPE_ID FROM RECIPE WHERE RECIPE_NAME = '{recipe_name}'")
        for i in mc:
            recipe_id = i[0]
```

```
        print(recipe_id)
        mc.execute(
            f"INSERT INTO RESPONSE(RECIPE_ID,COMMENTS,RATING) VALUES({recipe_id},'{comment}',{rating})")
        db.commit()
        response.destroy()

    rating_slider = CTkSlider(
        response, from_=0, to=5, number_of_steps=5, command=slider_event)
    rating_slider.place(relx=0.5, rely=0.25, anchor=tk.CENTER)
    submit = CTkButton(
        response, text="Submit", corner_radius=20, command=submit_clicked).place(relx=.385, rely=.75)
    textbox = CTkTextbox(
        response, width=200, height=100, corner_radius=20) textbox.place(relx=.33, rely=.35)
    textbox.bind("<FocusIn>", lambda e: textbox.delete("0.0", "end"))
    # insert at line 0 character 0
    textbox.insert("0.0", "Please add your comment/suggestion")
    # get text from line 0 character 0 till the end
    text = textbox.get("0.0", "end")
    response.mainloop()
  comment_rate = CTkButton(Display_recipe_search, text="Add a rating and comment", corner_radius=20,
                    fg_color='#3b3b3b', font=general_font, command=comment_rating, width=10).place(relx=.635, rely=.15)

  Display_recipe_search.mainloop()


def explore(user_id):

  mc.execute(f"SELECT USER_NAME FROM USERS WHERE USER_ID = {user_id}")
  for i in mc:
    un = i[0]
  print(un)
  Explore = CTk()
  Explore.geometry("700x550")
  Explore.title("Cookbook")
  welcome_font = CTkFont(family="Sans-serif", size=50)
  general_font = CTkFont(family="Sans-serif", size=15)
  display_font = CTkFont(family="Segoe Script", size=40)
  global search_bar
  search_bar = StringVar(value="Enter Recipe to search")
  sb = CTkEntry(Explore, font=general_font, corner_radius=30, height=35,
            width=400, fg_color="transparent", textvariable=search_bar)
  sb.place(relx=.225, rely=.45)
  s = CTkTextbox(Explore)
  sb.bind("<FocusIn>", lambda e: search_bar.set(""))
  name_display = CTkLabel(Explore, text="Name Cookbook",
                  font=display_font).place(relx=.25, rely=.30)
  explore_button = CTkButton(master=Explore, text="Explore", font=general_font,
                    corner_radius=30, command=explore_button_click).place(relx=.27, rely=.575)
  add_button = CTkButton(master=Explore, text="Add Recipe",
                  font=general_font, corner_radius=30, command=lambda: add_button_clicked(user_id)).place(relx=.545, rely=.575)
  my_image = CTkImage(dark_image=Image.open(
    "search-icon-png-29.png"), size=(15, 15))

  search_button = CTkButton(Explore, image=my_image, text="", fg_color="transparent",
                    width=20, height=5, corner_radius=50, command=display_recipe_search).place(relx=.745, rely=.46)
  date = dt.datetime.now()
  time_display = CTkLabel(
    Explore, text=f"Hello {un} It is {date:%A, %B %d, %Y}", font=general_font).place(relx=.55, rely=.05)
  global suggestion
  suggestion = CTkLabel(Explore, text="",
                  font=general_font)
  suggestion.place(relx=.25, rely=.20)
  Explore.mainloop()


# Login window
LOGIN = CTk()
LOGIN.geometry("700x550")
LOGIN.title("CookBook")
welcome_font = CTkFont(family="Sans-serif", size=50)
```

```
general_font = CTkFont(family="Sans-serif", size=15)
label = CTkLabel(
    LOGIN, text="", font=general_font, underline=True)
label.place(relx=0.35, rely=0.05)

CTkLabel(
    LOGIN, text="Welcome", font=welcome_font, underline=True).place(relx=0.35, rely=0.1)
CTkLabel(LOGIN, text="Please choose an option to continue",
         font=general_font, underline=True).place(relx=.33, rely=.25)

CTkSegmentedButton(LOGIN, values=["Login", "Register"], command=segemented_button_callback,
                   corner_radius=20, selected_hover_color="#70a7ff").place(relx=.36, rely=.35, width=250)
login_register_choose_var = StringVar(value="Login")

LOGIN.mainloop()
```

CONCLUSION

The Recipe Management System designed by us is an improvement over the current system of using traditional cookbooks. We have leveraged technology and developed a user-friendly system which is useful for cooks of all measures.
The system is thoroughly checked and tested and is found to be reliable for users of all categories. Therefore, we have implemented a system which has a modern user interface, requires minimal system requirements, running on the lowest grade of hardware along with being user friendly and approachable.

BIBLIOGRAPHY

WEBSITE REFERENCES:

Tkinrer/CustomTkinter:

- https://github.com/TomSchimansky/CustomTkinter
- https://docs.python.org/3/library/tkinter.html
- https://www.geeksforgeeks.org/python-gui-tkinter/

MySQL/MySQL Connector:

- https://www.w3schools.com/python/python_mysql_getstarted.asp

Contact Details :

Name : Aryaman .M (1DT20AI010)
E-mail : 1dt20ai010@dsatm.edu.in
Name : Mohammed Zabiullah .C (1DT20AI026)
EMAIL : 1dt20ai026@dsatm.edu.in