

Hướng Dẫn Phát Triển Ứng Dụng Recipe Manager với Python Tkinter

Tác giả: Đặng Kim Thi

1. Mục Tiêu

Dự án này nhằm giúp sinh viên thực hành lập trình GUI (Graphical User Interface) và làm việc với cơ sở dữ liệu thông qua việc phát triển ứng dụng **Recipe Manager** - một ứng dụng quản lý công thức nấu ăn với giao diện hiện đại và thân thiện. Mục tiêu cụ thể bao gồm:


- **Hiểu mô hình MVC (Model-View-Controller):** Học cách tổ chức mã nguồn chuyên nghiệp, tách biệt logic, giao diện, và dữ liệu.
- **Phát triển GUI với Tkinter:** Xây dựng giao diện đa màn hình, sử dụng các thành phần như frame, label, button, text, và scrolledtext.
- **Làm việc với cơ sở dữ liệu MongoDB:** Lưu trữ và truy xuất dữ liệu người dùng (góp ý) bằng MongoDB.
- **Xử lý hình ảnh với PIL (Pillow):** Tải, thay đổi kích thước, và hiển thị ảnh ngẫu nhiên trong giao diện.
- **Thiết kế giao diện hiện đại:** Áp dụng phong cách giao diện từ hình ảnh tham khảo (card-based layout, màu sắc tươi sáng, nút tròn) và cải thiện khả năng đọc.

Kết quả mong muốn là một ứng dụng desktop cho phép người dùng:

- Xem danh sách công thức nấu ăn theo thể loại.
- Xem chi tiết món ăn, bao gồm thời gian, khẩu phần, nguyên liệu, và hướng dẫn.
- Gửi góp ý và xem danh sách góp ý đã có.
- Điều hướng dễ dàng với các nút "Back to Category" và "Back to Home" trên trang chi tiết.
- Giao diện đẹp, màu sắc hài hòa, dễ đọc và sử dụng.

2. Mô Tả Ứng Dụng

Recipe Manager là một ứng dụng desktop giúp người dùng khám phá các công thức nấu ăn, được thiết kế với phong cách hiện đại dựa trên hình ảnh tham khảo:

 Hình ảnh tham khảo giao diện Recipe Manager

Hình ảnh trên mô tả một ứng dụng di động với giao diện thân thiện, gồm các màn hình như:

- Màn hình chào mừng với nút "Start Cooking".
- Màn hình danh mục với danh sách món ăn gần đây (Recent).
- Màn hình chi tiết món ăn với nguyên liệu, công thức, và video hướng dẫn.

Dựa trên hình ảnh và yêu cầu cập nhật, ứng dụng của chúng ta sẽ có các tính năng sau:

- **Màn hình chào mừng:** Hiển thị tiêu đề "Welcome to Recipes." và một ảnh ngẫu nhiên, với nút "Start Cooking" để bắt đầu.

- **Màn hình danh mục:** Hiển thị 4 thể loại (Main Courses, Desserts, Appetizers, Beverages), mỗi thể loại có 10 món ăn.
- **Màn hình danh sách món:** Hiển thị các món ăn thuộc thể loại đã chọn dưới dạng thẻ (card), bao gồm ảnh minh họa và nút "View".
- **Màn hình chi tiết món:** Hiển thị thông tin món (thời gian, khẩu phần, nguyên liệu, công thức), ảnh ngẫu nhiên, ô nhập góp ý, danh sách góp ý đã lưu, cùng với nút "Back to Category" và "Back to Home" để điều hướng.
- **Lưu trữ dữ liệu:** Sử dụng MongoDB để lưu và truy xuất góp ý của người dùng.

Ứng dụng sẽ sử dụng giao diện tiếng Anh và phong cách thiết kế từ hình ảnh tham khảo. Để cải thiện khả năng đọc, nền xám mờ của các nút trên trang chi tiết sẽ được thay bằng màu nền nhẹ nhàng hơn (ví dụ: `#ecf0f1` - xám nhạt), với chữ đen (`#2c3e50`) thay vì trắng, đảm bảo rõ ràng và thân thiện với mắt.

3. Kiến Trúc Tổng Quan

Ứng dụng được thiết kế theo mô hình **MVC (Model-View-Controller)** để đảm bảo mã nguồn sạch, dễ bảo trì, và mở rộng. Cấu trúc tổng quan như sau:

- **Model (`models/`):** Quản lý dữ liệu và logic xử lý dữ liệu.
 - Chứa danh sách thể loại (`categories`) và công thức (`recipes`).
 - Kết nối với MongoDB để lưu và truy xuất góp ý (`feedback`).
 - Xử lý ảnh ngẫu nhiên từ thư mục `images/`.
- **View (`views/`):** Quản lý giao diện người dùng.
 - Mỗi màn hình (welcome, category, recipe list, recipe detail) được tách thành một file riêng.
 - Sử dụng Tkinter để tạo các thành phần giao diện như label, button, text.
- **Controller (`controllers/`):** Điều phối logic giữa Model và View.
 - Xử lý sự kiện người dùng (nhấn nút, gửi góp ý).
 - Chuyển đổi giữa các màn hình, bao gồm các nút mới "Back to Category" và "Back to Home".
- **Main (`main.py`):** File khởi chạy ứng dụng, tạo cửa sổ chính và khởi tạo Controller.

Cấu trúc thư mục

```
recipe-manager/
├── images/
│   ├── food1.jpeg
│   ├── food2.jpeg
│   └── ... (đến food40.jpeg)
├── models/
│   ├── __init__.py
│   └── data.py
├── views/
│   ├── __init__.py
│   ├── welcome_screen.py
│   ├── category_screen.py
│   ├── recipe_list_screen.py
│   └── recipe_detail_screen.py
├── controllers/
│   └── __init__.py
```

```
├── app_controller.py
├── main.py
└── requirements.txt
```

4. Các Kỹ Thuật Sử Dụng

- **Tkinter**: Thư viện GUI của Python để tạo cửa sổ, khung (frame), nhãn (label), nút (button), vùng văn bản (text), và vùng cuộn (scrolledtext).
- **PIL (Pillow)**: Thư viện xử lý hình ảnh để tải, thay đổi kích thước, và hiển thị ảnh trong giao diện Tkinter.
- **MongoDB**: Cơ sở dữ liệu NoSQL để lưu trữ góp ý, sử dụng thư viện **pymongo** để kết nối và thao tác.
- **Mô hình MVC**:
 - **Model**: Quản lý dữ liệu (danh sách món, thể loại, góp ý) và kết nối MongoDB.
 - **View**: Xử lý giao diện người dùng (các màn hình).
 - **Controller**: Điều phối logic, xử lý sự kiện, và kết nối Model với View.
- **Xử lý sự kiện**: Sử dụng các hàm callback để xử lý hành động người dùng (như nhấn nút).
- **Quản lý tài nguyên**: Lưu trữ tham chiếu ảnh để tránh bị thu gom rác trong Tkinter.

5. Hướng Dẫn Thực Hiện

Bước 1: Chuẩn bị Môi Trường

1. Cài đặt Python:

- Tải và cài đặt Python từ python.org.
- Kiểm tra cài đặt bằng lệnh:

```
python --version
```

2. Cài đặt thư viện:

- Tạo file **requirements.txt** với nội dung:

```
pymongo
pillow
```

- Chạy lệnh để cài đặt:

```
pip install -r requirements.txt
```

3. Cài đặt MongoDB:

- Tải MongoDB từ mongodb.com và cài đặt.
- Chạy server MongoDB:

```
mongod
```

- Đảm bảo server chạy trên **localhost:27017**.

4. Chuẩn bị ảnh:

- Tạo thư mục **images/** trong thư mục dự án.
- Đặt 40 file ảnh (**food1.jpeg** đến **food40.jpeg**) vào thư mục này. Bạn có thể tải ảnh món ăn từ internet hoặc tự chụp.

Bước 2: Tạo Cấu Trúc Thư Mục

Tạo cấu trúc thư mục như mô tả ở phần **Kiến trúc tổng quan**. Đảm bảo tạo các file **__init__.py** rỗng trong các thư mục **models/**, **views/**, và **controllers/** để đánh dấu chúng là package.

Bước 3: Viết Code

3.1. Model (**models/data.py**)

- **Mục đích:** Quản lý dữ liệu (danh sách thể loại, món ăn) và kết nối MongoDB.
- **Nội dung:**

```
from pymongo import MongoClient
from PIL import Image, ImageTk
import random
import os

# Kết nối với MongoDB
client = MongoClient('mongodb://localhost:27017/')
db = client['recipe_manager']
feedback_collection = db['feedback']

# Dữ liệu mẫu cho thể loại và công thức
categories = [
    {"id": 1, "name": "Main Courses"},
    {"id": 2, "name": "Desserts"},
    {"id": 3, "name": "Appetizers"},
    {"id": 4, "name": "Beverages"},
]

recipes = [
    # Main Courses (10 món)
    {"id": 1, "categoryId": 1, "name": "Spiced Bean Tacos",
     "ingredients": "Iceberg lettuce, kidney beans, sour cream, jalapeños...",
     "steps": "Heat oil, add beans, serve in tacos...",
     "time": "25 min", "serves": "2"},
    {"id": 2, "categoryId": 1, "name": "Grilled Chicken",
     "ingredients": "Chicken, olive oil, garlic, rosemary...",
     "steps": "Marinate, grill for 20 min...",
     "time": "30 min", "serves": "4"},
    {"id": 3, "categoryId": 1, "name": "Beef Stir-Fry",
     "ingredients": "Beef, bell peppers, soy sauce...",
     "steps": "Stir-fry beef and
```

```

veggies...", "time": "20 min", "serves": "3"},
  {"id": 4, "categoryId": 1, "name": "Pasta Primavera",
"ingredients": "Pasta, zucchini, tomatoes, cream...", "steps": "Cook
pasta, mix with sauce...", "time": "25 min", "serves": "4"},
  {"id": 5, "categoryId": 1, "name": "Salmon with Lemon",
"ingredients": "Salmon, lemon, butter, herbs...", "steps": "Bake at
180°C for 15 min...", "time": "20 min", "serves": "2"},
  {"id": 6, "categoryId": 1, "name": "Vegetable Curry",
"ingredients": "Potatoes, peas, coconut milk...", "steps": "Simmer for
30 min...", "time": "35 min", "serves": "4"},
  {"id": 7, "categoryId": 1, "name": "Lamb Roast", "ingredients":
"Lamb, garlic, rosemary, potatoes...", "steps": "Roast at 200°C for 1
hour...", "time": "70 min", "serves": "6"},
  {"id": 8, "categoryId": 1, "name": "Shrimp Scampi", "ingredients":
"Shrimp, garlic, butter, pasta...", "steps": "Sauté shrimp, toss with
pasta...", "time": "20 min", "serves": "3"},
  {"id": 9, "categoryId": 1, "name": "Pork Chops", "ingredients":
"Pork, apple sauce, thyme...", "steps": "Grill for 15 min...", "time":
"20 min", "serves": "2"},
  {"id": 10, "categoryId": 1, "name": "Turkey Casserole",
"ingredients": "Turkey, cheese, cream, veggies...", "steps": "Bake at
180°C for 40 min...", "time": "50 min", "serves": "6"},
  # Desserts (10 món)
  {"id": 11, "categoryId": 2, "name": "Chocolate Cake",
"ingredients": "Flour, cocoa, sugar, eggs...", "steps": "Bake at 180°C
for 35 min...", "time": "50 min", "serves": "8"},
  {"id": 12, "categoryId": 2, "name": "Apple Pie", "ingredients":
"Apples, cinnamon, dough...", "steps": "Bake at 200°C for 45 min...",
"time": "60 min", "serves": "6"},
  {"id": 13, "categoryId": 2, "name": "Cheesecake", "ingredients":
"Cream cheese, sugar, graham...", "steps": "Chill for 4 hours...",
"time": "240 min", "serves": "8"},
  {"id": 14, "categoryId": 2, "name": "Lemon Tart", "ingredients":
"Lemon, eggs, sugar, crust...", "steps": "Bake at 160°C for 30
min...", "time": "40 min", "serves": "6"},
  {"id": 15, "categoryId": 2, "name": "Brownie", "ingredients":
"Chocolate, butter, flour...", "steps": "Bake at 180°C for 25 min...",
"time": "35 min", "serves": "8"},
  {"id": 16, "categoryId": 2, "name": "Fruit Sorbet", "ingredients":
"Berries, sugar, water...", "steps": "Freeze for 2 hours...", "time":
"120 min", "serves": "4"},
  {"id": 17, "categoryId": 2, "name": "Caramel Pudding",
"ingredients": "Milk, sugar, eggs...", "steps": "Steam for 20 min...",
"time": "30 min", "serves": "4"},
  {"id": 18, "categoryId": 2, "name": "Pecan Pie", "ingredients":
"Pecans, syrup, pie crust...", "steps": "Bake at 180°C for 50 min...",
"time": "60 min", "serves": "6"},
  {"id": 19, "categoryId": 2, "name": "Tiramisu", "ingredients":
"Coffee, mascarpone, ladyfingers...", "steps": "Chill for 6 hours...",
"time": "360 min", "serves": "8"},
  {"id": 20, "categoryId": 2, "name": "Ice Cream", "ingredients":
"Cream, sugar, vanilla...", "steps": "Churn for 20 min...", "time":
"30 min", "serves": "4"},
  # Appetizers (10 món)

```

```
{
  "id": 21, "categoryId": 3, "name": "Caprese Salad",
  "ingredients": "Tomato, mozzarella, basil...", "steps": "Layer and drizzle with oil...", "time": "10 min", "serves": "2"},
  "id": 22, "categoryId": 3, "name": "Bruschetta", "ingredients": "Bread, tomatoes, garlic...", "steps": "Toast and top with mix...", "time": "15 min", "serves": "4"},
  "id": 23, "categoryId": 3, "name": "Stuffed Mushrooms",
  "ingredients": "Mushrooms, cheese, breadcrumbs...", "steps": "Bake at 180°C for 20 min...", "time": "25 min", "serves": "6"},
  "id": 24, "categoryId": 3, "name": "Deviled Eggs", "ingredients": "Eggs, mayonnaise, mustard...", "steps": "Boil, mix, and fill...", "time": "20 min", "serves": "4"},
  "id": 25, "categoryId": 3, "name": "Spring Rolls", "ingredients": "Veggies, wrappers, sauce...", "steps": "Roll and fry for 5 min...", "time": "15 min", "serves": "6"},
  "id": 26, "categoryId": 3, "name": "Cheese Platter",
  "ingredients": "Cheese, grapes, crackers...", "steps": "Arrange on platter...", "time": "10 min", "serves": "4"},
  "id": 27, "categoryId": 3, "name": "Garlic Bread", "ingredients": "Bread, butter, garlic...", "steps": "Bake at 200°C for 10 min...", "time": "15 min", "serves": "4"},
  "id": 28, "categoryId": 3, "name": "Chicken Wings",
  "ingredients": "Wings, sauce, spices...", "steps": "Bake at 200°C for 30 min...", "time": "40 min", "serves": "6"},
  "id": 29, "categoryId": 3, "name": "Hummus with Pita",
  "ingredients": "Chickpeas, tahini, pita...", "steps": "Blend and serve...", "time": "15 min", "serves": "4"},
  "id": 30, "categoryId": 3, "name": "Mini Quiches", "ingredients": "Eggs, cheese, pastry...", "steps": "Bake at 180°C for 20 min...", "time": "30 min", "serves": "6"},
  # Beverages (10 món)
  "id": 31, "categoryId": 4, "name": "Iced Coffee", "ingredients": "Coffee, ice, milk...", "steps": "Brew and chill...", "time": "5 min", "serves": "1"},
  "id": 32, "categoryId": 4, "name": "Lemonade", "ingredients": "Lemon, sugar, water...", "steps": "Mix and serve with ice...", "time": "10 min", "serves": "2"},
  "id": 33, "categoryId": 4, "name": "Green Smoothie",
  "ingredients": "Spinach, banana, yogurt...", "steps": "Blend for 1 min...", "time": "5 min", "serves": "1"},
  "id": 34, "categoryId": 4, "name": "Hot Chocolate",
  "ingredients": "Cocoa, milk, sugar...", "steps": "Heat and stir...", "time": "10 min", "serves": "2"},
  "id": 35, "categoryId": 4, "name": "Fruit Punch", "ingredients": "Juices, fruits, soda...", "steps": "Mix and chill...", "time": "15 min", "serves": "6"},
  "id": 36, "categoryId": 4, "name": "Mint Tea", "ingredients": "Mint, water, honey...", "steps": "Steep for 5 min...", "time": "10 min", "serves": "2"},
  "id": 37, "categoryId": 4, "name": "Mango Lassi", "ingredients": "Mango, yogurt, milk...", "steps": "Blend until smooth...", "time": "5 min", "serves": "2"},
  "id": 38, "categoryId": 4, "name": "Spiced Chai", "ingredients": "Tea, spices, milk...", "steps": "Boil and strain...", "time": "15
```

```

min", "serves": "2"},
    {"id": 39, "categoryId": 4, "name": "Coconut Water",
"ingredients": "Coconut, ice...", "steps": "Serve fresh...", "time":
"5 min", "serves": "1"},
    {"id": 40, "categoryId": 4, "name": "Berry Infused Water",
"ingredients": "Berries, water, mint...", "steps": "Infuse for 1
hour...", "time": "60 min", "serves": "4"},
]

# Hàm lưu góp ý vào MongoDB
def save_feedback(recipe_name, feedback_text):
    feedback_data = {"recipeId": recipe_name, "feedback":
feedback_text}
    feedback_collection.insert_one(feedback_data)

# Hàm lấy danh sách góp ý từ MongoDB
def get_feedbacks(recipe_name):
    return list(feedback_collection.find({"recipeId": recipe_name}))

# Hàm lấy ảnh ngẫu nhiên từ thư mục images
def get_random_image():
    image_files = [f"food{i}.jpeg" for i in range(1, 41)]
    image_file = random.choice(image_files)
    image_path = os.path.join("images", image_file)
    try:
        img = Image.open(image_path).resize((400, 300), Image.LANCZOS)
        return ImageTk.PhotoImage(img)
    except FileNotFoundError:
        print(f"Error: Image file {image_path} not found.")
        return None

```

- **Giải thích:**

- `from pymongo import MongoClient`: Nhập thư viện để kết nối với MongoDB.
- `from PIL import Image, ImageTk`: Nhập thư viện Pillow để xử lý ảnh.
- `client = MongoClient('mongodb://localhost:27017/')`: Tạo kết nối đến MongoDB chạy trên localhost, cổng 27017.
- `db = client['recipe_manager']`: Chọn database 'recipe_manager'.
- `feedback_collection = db['feedback']`: Tạo collection 'feedback' để lưu góp ý.
- `categories` và `recipes`: Định nghĩa danh sách thể loại và công thức mẫu.
- `save_feedback(recipe_name, feedback_text)`: Tạo một tài liệu với tên món và góp ý, sau đó lưu vào MongoDB.
- `get_feedbacks(recipe_name)`: Truy vấn MongoDB để lấy tất cả góp ý của một món, chuyển thành danh sách.
- `get_random_image()`: Tạo danh sách file ảnh từ 1 đến 40, chọn ngẫu nhiên một file, mở và thay đổi kích thước ảnh thành 400x300, trả về `PhotoImage`. Nếu file không tồn tại, in thông báo lỗi và trả về `None`.

3.2. View (`views/welcome_screen.py`)

- **Mục đích:** Tạo màn hình chào mừng với giao diện giống hình ảnh tham khảo.

- **Nội dung:**

```
import tkinter as tk

class WelcomeScreen:
    def __init__(self, root, controller):
        self.frame = tk.Frame(root, bg="#fff3e0") # Tạo khung với màu
        nền vàng nhạt
        self.controller = controller

        tk.Label(self.frame, text="Welcome to Recipes.", font=
("Helvetica", 36, "bold"),
                bg="#fff3e0", fg="#f39c12").pack(pady=50) # Tiêu đề
        lớn, màu cam
        self.welcome_image = tk.Label(self.frame, bg="#fff3e0",
borderwidth=2, relief="solid")
        self.welcome_image.pack(pady=20) # Hiển thị ảnh ngẫu nhiên
        image = self.controller.get_random_image()
        if image:
            self.welcome_image.image = image
            self.welcome_image.config(image=image)
        tk.Button(self.frame, text="Start Cooking", font=("Helvetica",
16), bg="#f1c40f", fg="white",
                relief="flat",
        command=self.controller.show_categories, padx=30, pady=15,
borderwidth=0).pack(pady=30)
```

- **Giải thích:**

- `self.frame = tk.Frame(root, bg="#fff3e0")`: Tạo một khung (frame) làm màn hình chào mừng, sử dụng màu nền vàng nhạt `#fff3e0` để tạo cảm giác ấm áp.
- `tk.Label(...)`: Tạo nhãn với tiêu đề "Welcome to Recipes.", sử dụng font Helvetica kích thước 36, in đậm, màu chữ cam `#f39c12` để nổi bật trên nền.
- `self.welcome_image = tk.Label(...)`: Tạo nhãn để hiển thị ảnh, với viền mỏng và kiểu solid.
- `image = self.controller.get_random_image()`: Gọi phương thức từ Controller để lấy ảnh ngẫu nhiên, kiểm tra nếu ảnh hợp lệ thì gán vào nhãn.
- `tk.Button(...)`: Tạo nút "Start Cooking" với màu vàng `#f1c40f`, chữ trắng, không viền (`relief="flat"`), kích thước padding 30x15, khi nhấn gọi `show_categories()` để chuyển màn hình.

3.3. View (`views/category_screen.py`)

- **Mục đích:** Hiển thị danh sách thể loại món ăn.
- **Nội dung:**

```
import tkinter as tk

class CategoryScreen:
```



```

def __init__(self, root, controller):
    self.frame = tk.Frame(root, bg="#fff3e0") # Khung với màu nền
    vàng nhạt
    self.controller = controller

    tk.Label(self.frame, text="Categories", font=("Helvetica", 36,
"bold"),
            bg="#fff3e0", fg="#f39c12").pack(pady=50) # Tiêu đề
    lớn, màu cam
    self.category_list = tk.Frame(self.frame, bg="#fff3e0") #
    Khung chứa danh sách thẻ loại
    self.category_list.pack(pady=20)
    for category in self.controller.get_categories():
        btn = tk.Button(self.category_list, text=category["name"],
font=("Helvetica", 18),
                        bg="ffffff", fg="#2c3e50", relief="flat",
borderwidth=0,
                        command=lambda cid=category["id"]:
self.controller.show_recipes(cid),
                        padx=40, pady=20,
activebackground="#ecf0f1")
        btn.pack(fill="x", pady=15, padx=50, ipadx=20) # Nút trái
        dài, padding đều
        btn.config(highlightbackground="#ddd",
highlightthickness=1) # Viền mỏng khi hover
        tk.Button(self.frame, text="Back", font=("Helvetica", 14),
bg="#e74c3c", fg="white",
                    relief="flat", command=self.controller.show_welcome,
padx=20, pady=10).pack(pady=20)

```

• Giải thích:

- `self.frame = tk.Frame(...)`: Tạo khung với màu nền #fff3e0.
- `tk.Label(...)`: Hiển thị tiêu đề "Categories" với font lớn, màu cam #f39c12.
- `self.category_list = tk.Frame(...)`: Tạo khung con để chứa các nút thẻ loại.
- `for category in self.controller.get_categories()`: Lặp qua danh sách thẻ loại từ Controller.
- `btn = tk.Button(...)`: Tạo nút cho mỗi thẻ loại, nền trắng #ffffff, chữ đen #2c3e50, không viền, khi nhấn gọi `show_recipes()` với ID thẻ loại. `activebackground="#ecf0f1"` thay đổi màu khi hover.
- `btn.pack(...)`: Sắp xếp nút trái dài ngang, padding đều.
- `btn.config(...)`: Thêm viền mỏng #ddd khi hover để tăng tương tác.
- `tk.Button(...)`: Nút "Back" với màu đỏ #e74c3c, chữ trắng, quay về màn hình chào mừng.

3.4. View (`views/recipe_list_screen.py`)

- **Mục đích:** Hiển thị danh sách món ăn dưới dạng thẻ (card) với ảnh.
- **Nội dung:**

```

import tkinter as tk

class RecipeListScreen:
    def __init__(self, root, controller):
        self.frame = tk.Frame(root, bg="#fff3e0") # Khung với màu nền
        vàng nhạt
        self.controller = controller

        tk.Label(self.frame, text="Recipes", font=("Helvetica", 36,
"bold"),
                bg="#fff3e0", fg="#f39c12").pack(pady=50) # Tiêu đề
        lớn, màu cam
        self.recipe_list = tk.Frame(self.frame, bg="#fff3e0") # Khung
        chứa danh sách món
        self.recipe_list.pack(pady=20)
        tk.Button(self.frame, text="Back", font=("Helvetica", 14),
        bg="#e74c3c", fg="white",
                relief="flat",
        command=self.controller.show_categories, padx=20,
        pady=10).pack(pady=20)

    def update_recipes(self, recipes):
        for widget in self.recipe_list.winfo_children():
            widget.destroy() # Xóa các widget cũ
        for recipe in recipes:
            frame = tk.Frame(self.recipe_list, bg="ffffff",
        borderwidth=1, relief="solid") # Thẻ món
            frame.pack(fill="x", pady=10, padx=50)
            image = self.controller.get_random_image()
            if image:
                img_label = tk.Label(frame, image=image, bg="ffffff")
                img_label.image = image # Giữ tham chiếu ảnh
                img_label.pack(side="left", padx=10)
                tk.Label(frame, text=recipe["name"], font=("Helvetica",
        16, "bold"), fg="#2c3e50", bg="ffffff",
                        wraplength=500).pack(side="left", padx=10,
        pady=5) # Tên món
                tk.Button(frame, text="View", font=("Helvetica", 12),
        bg="#f1c40f", fg="white", relief="flat",
                        command=lambda rid=recipe["id"]:
        self.controller.show_recipe_detail(rid), padx=15,
        pady=5).pack(side="right", padx=10)

```

- **Giải thích:**

- `self.frame = tk.Frame(...)`: Tạo khung với màu nền #fff3e0.
- `tk.Label(...)`: Hiển thị tiêu đề "Recipes" với font lớn, màu cam #f39c12.
- `self.recipe_list = tk.Frame(...)`: Khung con chứa danh sách món.
- `tk.Button(...)`: Nút "Back" với màu đỏ #e74c3c, quay về màn hình danh mục.
- `def update_recipes(self, recipes)`: Hàm cập nhật danh sách món.
- `for widget in self.recipe_list.winfo_children()`: Xóa các widget cũ để làm mới danh sách.

- `frame = tk.Frame(...)`: Tạo thẻ (card) cho mỗi món, nền trắng với viền mỏng.
- `image = self.controller.get_random_image()`: Lấy ảnh ngẫu nhiên, kiểm tra và hiển thị nếu hợp lệ.
- `tk.Label(...)`: Hiển thị tên món với font đậm, màu đen `#2c3e50`, giới hạn chiều rộng 500 pixel.
- `tk.Button(...)`: Nút "View" với màu vàng `#f1c40f`, gọi `show_recipe_detail()` khi nhấn.

3.5. View (`views/recipe_detail_screen.py`)

- **Mục đích:** Hiển thị chi tiết món ăn, quản lý góp ý, và thêm nút "Back to Category" và "Back to Home".
- **Nội dung:**

```
import tkinter as tk
from tkinter import scrolledtext, messagebox

class RecipeDetailScreen:
    def __init__(self, root, controller):
        self.frame = tk.Frame(root, bg="#fff3e0") # Khung với màu nền
        vàng nhạt
        self.controller = controller

        self.detail_name = tk.Label(self.frame, font=("Helvetica", 28,
        "bold"),
                                   bg="#fff3e0", fg="#f39c12") #
        Tiêu đề món
        self.detail_name.pack(pady=20)

        self.detail_image = tk.Label(self.frame, bg="#fff3e0",
        borderwidth=2, relief="solid") # Hiển thị ảnh
        self.detail_image.pack(pady=10)

        self.detail_info_frame = tk.Frame(self.frame, bg="#fff3e0") #
        Khung thông tin
        self.detail_info_frame.pack(pady=10, padx=50)
        tk.Label(self.detail_info_frame, text="Time: ", font=
        ("Helvetica", 14), fg="#2c3e50", bg="#fff3e0").pack(side="left")
        self.detail_time = tk.Label(self.detail_info_frame, font=
        ("Helvetica", 14), fg="#2c3e50", bg="#fff3e0")
        self.detail_time.pack(side="left", padx=5)
        tk.Label(self.detail_info_frame, text=" | Serves: ", font=
        ("Helvetica", 14), fg="#2c3e50", bg="#fff3e0").pack(side="left")
        self.detail_serves = tk.Label(self.detail_info_frame, font=
        ("Helvetica", 14), fg="#2c3e50", bg="#fff3e0")
        self.detail_serves.pack(side="left", padx=5)

        self.detail_ingredients = tk.Label(self.frame,
        text="Ingredients: ", font=("Helvetica", 16, "bold"),
                                   fg="#2c3e50", bg="#fff3e0",
        justify="left") # Tiêu đề nguyên liệu
        self.detail_ingredients.pack(pady=5)
```

```

        self.ingredients_text = tk.Label(self.frame, font=
("Helvetica", 14), fg="#2c3e50",
                                bg="#ffffff", wraplength=800,
padx=20, pady=10, justify="left") # Nội dung nguyên liệu
        self.ingredients_text.pack(fill="x", pady=5)

        self.detail_steps = tk.Label(self.frame, text="Steps: ", font=
("Helvetica", 16, "bold"),
                                fg="#2c3e50", bg="#fff3e0",
justify="left") # Tiêu đề hướng dẫn
        self.detail_steps.pack(pady=5)
        self.steps_text = tk.Label(self.frame, font=("Helvetica", 14),
fg="#2c3e50",
                                bg="#ffffff", wraplength=800,
padx=20, pady=10, justify="left") # Nội dung hướng dẫn
        self.steps_text.pack(fill="x", pady=5)

        tk.Label(self.frame, text="Your Feedback:", font=("Helvetica",
16, "bold"),
                                fg="#2c3e50", bg="#fff3e0").pack(pady=10) # Tiêu đề
góp ý
        self.feedback_text = tk.Text(self.frame, height=4, width=60,
font=("Helvetica", 12),
                                bg="#ffffff", borderwidth=1,
relief="flat") # Ô nhập góp ý
        self.feedback_text.pack(pady=10)
        tk.Button(self.frame, text="Submit Feedback", font=
("Helvetica", 12), bg="#2ecc71", fg="white",
                                relief="flat", command=self.submit_feedback,
padx=20, pady=10).pack(pady=10) # Nút gửi góp ý

        tk.Label(self.frame, text="Feedbacks:", font=("Helvetica", 16,
"bold"),
                                fg="#2c3e50", bg="#fff3e0").pack(pady=10) # Tiêu đề
danh sách góp ý
        self.feedback_list = scrolledtext.ScrolledText(self.frame,
height=8, width=60,
                                font=
("Helvetica", 12), bg="#ffffff", relief="flat",
                                borderwidth=1,
wrap=tk.WORD) # Danh sách góp ý
        self.feedback_list.pack(pady=10)

        # Nút Back to Category và Back to Home
        button_frame = tk.Frame(self.frame, bg="#fff3e0")
        button_frame.pack(pady=20)
        tk.Button(button_frame, text="Back to Category", font=
("Helvetica", 14), bg="#ecf0f1", fg="#2c3e50",
                                relief="flat", command=self.controller.show_recipes,
padx=20, pady=10).pack(side="left", padx=10)
        tk.Button(button_frame, text="Back to Home", font=
("Helvetica", 14), bg="#ecf0f1", fg="#2c3e50",
                                relief="flat", command=self.controller.show_welcome,
padx=20, pady=10).pack(side="left", padx=10)

```

```

def update_details(self, recipe):
    self.detail_name.config(text=recipe["name"]) # Cập nhật tên
    món
    image = self.controller.get_random_image()
    if image:
        self.detail_image.image = image
        self.detail_image.config(image=image) # Cập nhật ảnh
    self.detail_time.config(text=recipe["time"]) # Cập nhật thời
    gian
    self.detail_serves.config(text=recipe["serves"]) # Cập nhật
    khẩu phần
    self.ingredients_text.config(text=recipe["ingredients"]) #
    Cập nhật nguyên liệu
    self.steps_text.config(text=recipe["steps"]) # Cập nhật hướng
    dẫn
    self.feedback_text.delete(1.0, tk.END) # Xóa nội dung cũ
    trong ô góp ý
    self.feedback_list.delete(1.0, tk.END) # Xóa danh sách góp ý
    cũ
    feedbacks = self.controller.get_feedbacks(recipe["name"]) #
    Lấy danh sách góp ý
    for fb in feedbacks:
        self.feedback_list.insert(tk.END, f"- {fb['feedback']}\n")
    # Thêm góp ý vào danh sách

def submit_feedback(self):
    feedback = self.feedback_text.get(1.0, tk.END).strip() # Lấy
    nội dung góp ý
    recipe_name = self.detail_name.cget("text") # Lấy tên món
    if not feedback:
        messagebox.showwarning("Error", "Please enter your
    feedback!") # Cảnh báo nếu trống
        return
    self.controller.save_feedback(recipe_name, feedback) # Lưu
    góp ý
    messagebox.showinfo("Success", "Feedback submitted!") # Thông
    báo thành công
    self.feedback_text.delete(1.0, tk.END) # Xóa ô góp ý

self.update_details(self.controller.get_recipe_by_id(self.controller.c
urrent_recipe_id)) # Cập nhật lại giao diện

```

- **Giải thích:**

- `self.frame = tk.Frame(...)`: Tạo khung với màu nền #fff3e0.
- `self.detail_name = tk.Label(...)`: Nhãn cho tên món, font lớn, màu cam #f39c12.
- `self.detail_image = tk.Label(...)`: Nhãn cho ảnh, với viền mỏng.
- `self.detail_info_frame = tk.Frame(...)`: Khung chứa thông tin thời gian và khẩu phần.
- `tk.Label(self.detail_info_frame, ...)`: Nhãn tĩnh "Time:" và " | Serves:".

- `self.detail_time` và `self.detail_serves`: Nhấn động để hiển thị thời gian và khẩu phần.
- `self.detail_ingredients` và `self.ingredients_text`: Nhấn cho tiêu đề và nội dung nguyên liệu.
- `self.detail_steps` và `self.steps_text`: Nhấn cho tiêu đề và nội dung hướng dẫn.
- `self.feedback_text = tk.Text(...)`: Ô nhập góp ý, kích thước 4 dòng, 60 ký tự.
- `tk.Button(...)`: Nút "Submit Feedback" với màu xanh lá `#2ecc71`, chữ trắng.
- `self.feedback_list = scrolledtext.ScrolledText(...)`: Vùng cuộn để hiển thị danh sách góp ý, chiều cao 8 dòng.
- `button_frame = tk.Frame(...)`: Khung chứa hai nút mới.
- `tk.Button(button_frame, text="Back to Category", ...)`: Nút với màu nền xám nhạt `#ecf0f1`, chữ đen `#2c3e50`, gọi `show_recipes()` để quay lại danh sách món.
- `tk.Button(button_frame, text="Back to Home", ...)`: Nút tương tự, gọi `show_welcome()` để quay về màn hình chào mừng.
- `def update_details(self, recipe)`: Cập nhật tất cả thông tin món và danh sách góp ý.
- `def submit_feedback(self)`: Lấy góp ý, kiểm tra hợp lệ, lưu vào MongoDB, và làm mới giao diện.

3.6. Controller (`controllers/app_controller.py`)

- **Mục đích:** Điều phối logic giữa Model và View.
- **Nội dung:**

```
from models.data import categories, recipes, save_feedback,
get_feedbacks, get_random_image
from views.welcome_screen import WelcomeScreen
from views.category_screen import CategoryScreen
from views.recipe_list_screen import RecipeListScreen
from views.recipe_detail_screen import RecipeDetailScreen

class AppController:
    def __init__(self, root):
        self.root = root # Cửa sổ chính
        self.root.title("Recipes") # Tiêu đề cửa sổ
        self.root.geometry("1000x800") # Kích thước cửa sổ
        self.root.configure(bg="#fff3e0") # Màu nền cửa sổ

        self.current_category_id = None # Lưu ID thể loại hiện tại
        self.current_recipe_id = None # Lưu ID món hiện tại

        self.welcome_screen = WelcomeScreen(root, self) # Khởi tạo
        màn hình chào mừng
        self.category_screen = CategoryScreen(root, self) # Khởi tạo
        màn hình danh mục
        self.recipe_list_screen = RecipeListScreen(root, self) # Khởi
        tạo màn hình danh sách món
        self.recipe_detail_screen = RecipeDetailScreen(root, self) #
        Khởi tạo màn hình chi tiết

        self.show_welcome() # Hiển thị màn hình chào mừng đầu tiên
```

```

    def show_frame(self, frame):
        for f in (self.welcome_screen.frame,
self.category_screen.frame,
                self.recipe_list_screen.frame,
self.recipe_detail_screen.frame):
            f.pack_forget() # Ẩn tất cả các khung
            frame.pack(fill="both", expand=True) # Hiển thị khung được
chọn

    def get_categories(self):
        return categories # Trả về danh sách thể loại

    def get_recipes_by_category(self, category_id):
        return [r for r in recipes if r["categoryId"] == category_id]
# Lọc món theo thể loại

    def get_recipe_by_id(self, recipe_id):
        return next(r for r in recipes if r["id"] == recipe_id) # Lấy
món theo ID

    def get_random_image(self):
        return get_random_image() # Gọi hàm lấy ảnh ngẫu nhiên từ
Model

    def save_feedback(self, recipe_name, feedback_text):
        save_feedback(recipe_name, feedback_text) # Lưu góp ý vào
MongoDB

    def get_feedbacks(self, recipe_name):
        return get_feedbacks(recipe_name) # Lấy danh sách góp ý

    def show_welcome(self):
        self.show_frame(self.welcome_screen.frame) # Hiển thị màn
hình chào mừng

    def show_categories(self):
        self.show_frame(self.category_screen.frame) # Hiển thị màn
hình danh mục

    def show_recipes(self, category_id):
        self.current_category_id = category_id # Cập nhật ID thể loại
        recipes_data = self.get_recipes_by_category(category_id) #
Lấy danh sách món
        self.recipe_list_screen.update_recipes(recipes_data) # Cập
nhật giao diện
        self.show_frame(self.recipe_list_screen.frame) # Hiển thị màn
hình danh sách

    def show_recipe_detail(self, recipe_id):
        self.current_recipe_id = recipe_id # Cập nhật ID món
        recipe = self.get_recipe_by_id(recipe_id) # Lấy thông tin món
        self.recipe_detail_screen.update_details(recipe) # Cập nhật
giao diện

```

```
self.show_frame(self.recipe_detail_screen.frame) # Hiển thị
màn hình chi tiết
```

- **Giải thích:**

- `self.root = root`: Lưu tham chiếu đến cửa sổ chính.
- `self.root.title(...)`: Đặt tiêu đề "Recipes".
- `self.root.geometry(...)`: Đặt kích thước 1000x800 pixel.
- `self.root.configure(...)`: Đặt màu nền `#fff3e0`.
- `self.current_category_id` và `self.current_recipe_id`: Biến lưu trạng thái hiện tại.
- `self.welcome_screen = WelcomeScreen(...)`: Khởi tạo các màn hình.
- `def show_frame(self, frame)`: Ẩn tất cả khung và hiển thị khung được chọn.
- `def get_categories()`: Trả về danh sách thể loại từ Model.
- `def get_recipes_by_category(...)`: Lọc danh sách món theo ID thể loại.
- `def get_recipe_by_id(...)`: Lấy thông tin món theo ID.
- `def get_random_image()`: Gọi hàm từ Model để lấy ảnh.
- `def save_feedback(...)` và `def get_feedbacks(...)`: Xử lý góp ý với MongoDB.
- `def show_welcome(), def show_categories(), def show_recipes(...), def show_recipe_detail(...)`: Điều hướng giữa các màn hình.

3.7. Main (`main.py`)

- **Mục đích:** Khởi chạy ứng dụng.
- **Nội dung:**

```
import tkinter as tk
from controllers.app_controller import AppController

if __name__ == "__main__":
    root = tk.Tk() # Tạo cửa sổ chính
    app = AppController(root) # Khởi tạo Controller
    root.mainloop() # Chạy vòng lặp giao diện
```

- **Giải thích:**

- `import tkinter as tk`: Nhập thư viện Tkinter.
- `root = tk.Tk()`: Tạo cửa sổ chính.
- `app = AppController(root)`: Khởi tạo ứng dụng với Controller.
- `root.mainloop()`: Bắt đầu vòng lặp chính để hiển thị và xử lý sự kiện.

Bước 4: Chạy và Kiểm Tra

1. Chạy MongoDB server:

```
mongod
```

2. Chạy ứng dụng:


```
python main.py
```

3. Kiểm tra:

- Màn hình chào mừng hiển thị tiêu đề "Welcome to Recipes." và ảnh ngẫu nhiên.
- Chuyển sang màn hình danh mục, chọn một thể loại để xem danh sách món.
- Xem chi tiết món, kiểm tra nút "Back to Category" (quay về danh sách) và "Back to Home" (quay về chào mừng), nhập góp ý, và kiểm tra danh sách góp ý (dùng MongoDB Compass để xem dữ liệu).

Bước 5: Tối Ưu và Mở Rộng

• Tối ưu:

- Thêm xử lý ngoại lệ cho kết nối MongoDB (nếu server không chạy).
- Sử dụng thread để tải ảnh, tránh làm chậm giao diện.

• Mở rộng:

- Thêm nút "Like" hoặc "Share" như trong hình ảnh tham khảo.
- Tích hợp video hướng dẫn nấu ăn (dùng thư viện như `vlc-python`).
- Sử dụng `ttkthemes` để áp dụng theme hiện đại hơn.

6. Lưu Ý Cho Sinh Viên

- **Hiểu cấu trúc MVC:** Thử thay đổi giao diện (View) mà không ảnh hưởng đến dữ liệu (Model) để hiểu cách tách biệt code.
- **Debugging:** Sử dụng `print()` hoặc debugger trong IDE (như PyCharm, VSCode) để tìm lỗi. Nếu ảnh không hiển thị, kiểm tra đường dẫn file trong `images/`.
- **Tài liệu:** Ghi chú code bằng comment để dễ hiểu khi quay lại chỉnh sửa.
- **Hợp tác nhóm:** Nếu làm theo nhóm, chia nhỏ công việc (Model, View, Controller) và dùng Git để quản lý mã nguồn.
- **Tham khảo giao diện:** Hình ảnh tham khảo là nguồn cảm hứng. Chú ý đến cách bố trí, màu sắc (như thay đổi từ xám mờ sang xám nhạt `#ecf0f1` với chữ đen `#2c3e50`), và phong cách để cải thiện giao diện.

7. Kết Quả Mong Muốn

Sau khi hoàn thành, bạn sẽ có:

- Một ứng dụng **Recipe Manager** với 4 màn hình: chào mừng, danh mục, danh sách món, và chi tiết món.
- Giao diện hiện đại, màu sắc tươi sáng, bố cục card-based, và khả năng đọc tốt với nút "Back to Category" và "Back to Home".
- Chức năng lưu và hiển thị góp ý trong MongoDB.
- Kinh nghiệm thực hành lập trình GUI, xử lý hình ảnh, và làm việc với cơ sở dữ liệu.