1. INTRODUCTION

1.1. Project Overview

Flavour Fusion is an AI-driven recipe blogging web application designed to generate customized and well-structured recipe blogs based on user input. The application allows users to enter a recipe topic and specify the desired word count. Using a pre-trained generative AI model, the system produces detailed and engaging recipe content in a short time.

The project aims to simplify the process of recipe content creation for food bloggers, cooking enthusiasts, and busy individuals. Instead of manually writing lengthy recipes, users can rely on the application to generate high-quality content instantly. The application is built using Streamlit for the user interface and integrates the Gemini Flash Lite (models/gemini-flash-lite-latest) model for efficient and fast text generation.

Overall, Flavour Fusion demonstrates the practical use of generative AI in content creation by providing a user-friendly, time-saving, and effective solution for automated recipe blog generation.

1.2. Objectives

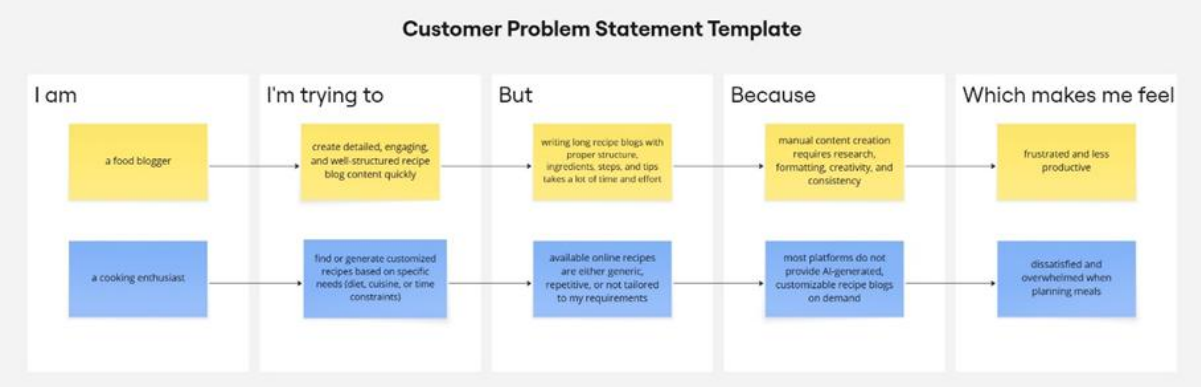
The main objectives of the Flavour Fusion: AI-Driven Recipe Blogging project are:

- To develop a web-based application that generates recipe blogs using artificial intelligence
- To allow users to create customized recipe content by providing a topic and word count
- To reduce the time and effort required for manual recipe writing
- To integrate a pre-trained generative AI model for fast and accurate text generation
- To provide a simple and user-friendly interface using Streamlit
- To enhance user experience through efficient and engaging content generation

2. Project Initialization and Planning Phase

2.1. Defining Problem Statement

Therefore, food bloggers and busy individuals are in dire need of an easier way to pen or write a detailed recipe blog in the shortest length of time. Writing recipes manually requires a lot of efforts, creativities, and proper formatting that are really very time-consuming. Also, most of the recipes available online tend to be general and do not meet specific user needs. The result is usually frustration and reduced productivity. The Flavour Fusion application solves this problem by using an AI model to quickly generate customized and well-structured recipe blogs based on user input.



Problem Statement (PS)	I am	I'm trying to	But	Because	Which makes me feel
PS-1	A food blogger	Create detailed, and high-quality recipe blog content in less time	Writing long and recipe blogs manually is time-consuming	It requires research, creativity, formatting, and consistency	Frustrated and less productive
PS-2	A cooking enthusiast	Generate customized recipes	Existing recipes available online are generic and not customizable	Most platforms do not offer AI-driven recipe blog generation	Dissatisfied and overwhelmed while planning meals

2.2. Project Proposal (Proposed Solution)

Project Overview	
Objective	The main objective of this project is to develop an AI-powered web application that generates customized and high-quality recipe blogs based on user inputs.
Scope	The scope of the project includes building a Streamlit-based web interface where users can enter a recipe topic and desired word count. The system generates a complete recipe blog using an AI model and displays it to the user.
Problem Statement	
Description	Many food bloggers and users find it difficult to create detailed and well-structured recipe blogs within a short time. Manual content creation requires creativity, research, and proper formatting, making it time-consuming.
Impact	Solving this problem helps users save time and effort while creating customized recipe blogs
Proposed Solution	
Approach	The proposed solution uses a Streamlit web application integrated with the Gemini Flash Lite model. Users provide a recipe topic and word count through the UI. The AI model processes the input and generates a complete recipe blog.
Key Features	• User-friendly Streamlit interface • Custom recipe blog generation based on topic and word count • AI-powered content generation using Gemini Flash Lite • Programmer joke display for better user experience

Resource Requirements

Resource Type	Description	Specification/Allocation
Hardware		
Computing Resources	CPU for application execution	Standard system CPU
Memory	RAM required to run application	8 GB
Storage	Disk space for application files	50 GB SSD
Software		
Frameworks	Python web framework	Streamlit
Libraries	AI and utility libraries	google-generativeai
Development Environment	IDE and version control	VS Code, Git
Data		
Data	User input text	Recipe topic and word count entered by user

2.3. Initial Project Planning

Product Backlog, Sprint Schedule, and Estimation

The following table represents the product backlog and sprint-wise planning for the **Flavour Fusion** project.

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members	Sprint Start Date	Sprint End Date
Sprint 1	User Interface Setup	USN-1	As a user, I can access a Streamlit-based interface to enter a recipe topic and word count.	2	High	All Team Members	28 January 2026	31 January 2026
Sprint 1	Input Validation	USN-2	As a user, I want the	1	High	All Team Members	28 January	31 January

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members	Sprint Start Date	Sprint End Date
			application to validate my inputs before generating the recipe.				2026	2026
Sprint 2	AI Model Integration	USN-3	As a user, I want the system to generate a recipe blog using the Gemini Flash Lite model.	3	High	All Team Members	02 Februar y 2026	09 Februar y 2026
Sprint 2	Joke Generation	USN-4	As a user, I want to see a programming joke while the recipe is being generated.	1	Medium	All Team Members	02 Februar y 2026	09 Februar y 2026
Sprint 3	Output Display	USN-5	As a user, I want to view the generated recipe blog clearly on the screen.	2	High	All Team Members	12 Februar y 2026	18 Februar y 2026
Sprint 3	Deployment	USN-6	As a user, I want the application to be deployed and accessible through the internet.	2	Medium	All Team Members	12 Februar y 2026	18 Februar y 2026

3. Data Collection and Preprocessing Phase

3.1. Data Collection plan and Raw Data Sources Identified

3.1.1. Data Collection Plan

Section	Description
Project Overview	Flavour Fusion is an AI-driven recipe blogging application that generates customized recipe blogs based on user input. The project dynamically processes user-provided text input to generate content.
Data Collection Plan	The data for this project is collected **directly from users at runtime** through a Streamlit-based web interface. Users provide: • Recipe topic • Desired word count
Raw Data Sources Identified	Since this project uses real-time user input and a pre-trained AI model, no external datasets (CSV, images, or files) are collected or stored.

3.1.2. Raw Data Sources Template

Source Name	Description	Location/ URL	Format	Size	Access Permissions
User Input Data	Text entered by users (recipe topic & word count)	Streamlit Web Interface	Text	Small	Public (user-provided)
AI Model Output	Generated recipe blog content	Gemini Flash Lite API	Text	Variable	Restricted (API-based)

3.2. Data Quality Report

Data Quality Report

The Data Quality Report summarizes potential data quality issues related to user-provided input in the Flavour Fusion application. Since the project does not rely on an external dataset, data quality focuses on ensuring valid, clean, and meaningful text input before processing by the AI model.

Data Source	Data Quality Issue	Severity	Resolution Plan
User Input (Text)	Empty or missing recipe topic	High	Input validation is applied to ensure the topic field is not empty before submission.
User Input (Text)	Invalid or very low word count	Moderate	Word count is validated to ensure it falls within an acceptable range.
User Input (Text)	Special characters or extra spaces	Low	Input text is cleaned by trimming unnecessary spaces before processing.
User Input (Text)	Ambiguous or unclear prompts	Low	Clear prompt formatting is applied before sending input to the AI model.

3.3. Data Preprocessing

In the Flavour Fusion project, data preprocessing focuses on **user-provided textual input** rather than images. Since the application uses a **pre-trained generative AI model**, no traditional dataset collection or image preprocessing is required.

Section	Description
Data Overview	The data used in this project consists of user-entered text inputs such as recipe topic and desired word count. No external dataset is used.
Text Cleaning	User input is cleaned by removing unnecessary spaces and validating empty or invalid entries.
Input	Ensures the recipe topic is not empty and the word count is within an

Validation	acceptable range.
Token Handling	The input text is passed to the Gemini Flash Lite model, which internally handles tokenization and text processing.
Prompt Formatting	The user input is formatted into a structured prompt before being sent to the AI model for recipe generation.
Error Handling	Handles invalid inputs or API-related issues gracefully by displaying appropriate error messages.
Data Preprocessing Areas	
Loading Data	User inputs are collected directly through Streamlit text input fields.
Input Validation	Code ensures valid recipe topic and word count before processing.
Prompt Creation	Code formats the validated input into a prompt for the Gemini Flash Lite model.
Model Invocation	The formatted prompt is sent to the AI model for recipe blog generation.
Output Handling	The generated recipe text is received and displayed on the Streamlit interface.

4. Model Development Phase

4.1. Model Selection Report

In this project, the focus is on selecting a pre-trained generative language model suitable for real-time recipe blog generation. Unlike traditional deep learning projects that require training CNNs or RNNs, this application leverages an existing large language model (LLM) to generate high-quality text content. The model is selected based on performance, response time, ease of integration, and suitability for text-based content generation.

Model	Description
Gemini Flash Lite (models/gemini-flash-latest)	A lightweight pre-trained generative language model designed for fast and efficient text generation. It supports real-time content creation with good coherence, relevance, and low latency, making it ideal for recipe blog generation.

4.2. Initial Model Training Code, Model Validation and Evaluation Report

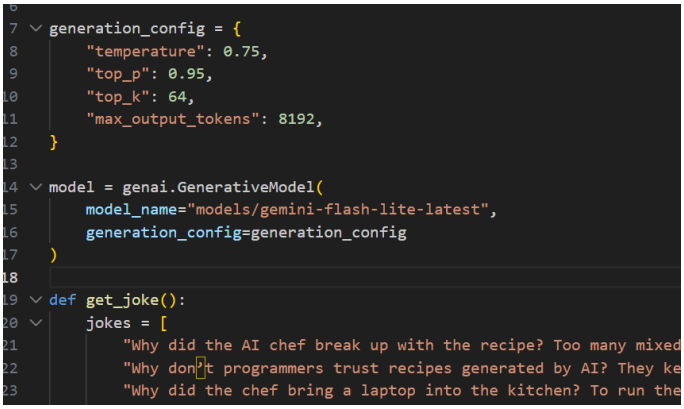
In this project, no custom model training is performed. Instead, a pre-trained generative AI model is integrated and used for recipe blog generation. The focus of this phase is on model selection, configuration, prompt design, and output evaluation, rather than training from scratch.

Initial Model Training Code:

Model Selection and Initialization

The **Gemini Flash Lite (models/gemini-flash-latest)** model is selected due to its lightweight architecture, faster response time, and suitability for real-time text generation tasks.

The model is initialized using the Google Generative AI API. Configuration parameters such as temperature, top-p, top-k, and maximum output tokens are defined to control creativity, response quality, and output length.



```
generation_config = {
    "temperature": 0.75,
    "top_p": 0.95,
    "top_k": 64,
    "max_output_tokens": 8192,
}

model = genai.GenerativeModel(
    model_name="models/gemini-flash-lite-latest",
    generation_config=generation_config
)

def get_joke():
    jokes = [
        "Why did the AI chef break up with the recipe? Too many mixed
        "Why don't programmers trust recipes generated by AI? They ke
        "Why did the chef bring a laptop into the kitchen? To run the
```

Model Validation and Evaluation Report:

Model	Summary	Training and Validation Performance Metrics
Gemini Flash Lite	Pre-trained generative language model optimized for fast text generation	Relevance of generated content, adherence to word count, coherence, clarity, and response time

5. Model Optimization and Tuning Phase

5.1. Tuning Documentation

The Model Optimization and Tuning Phase focuses on improving the performance and quality of the generated recipe blogs by configuring and fine-tuning model generation parameters rather than training a neural network from scratch. Since the project uses a pre-trained generative AI model, optimization is achieved through prompt design and parameter tuning.

Hyperparameter Tuning Documentation:

Model	**Tuned Hyper parameters**
Gemini Flash Lite	**Temperature:** Controls creativity of output (set to a moderate value for balanced creativity). **Top-p:** Limits token selection to the most probable tokens for coherent responses. **Top-k:** Restricts token sampling to reduce irrelevant content. **Max Output Tokens:** Ensures generated recipe blogs match the desired word count. **Response Format:** Set to plain text for easy display in the UI.

5.2. Final Model Selection Justification

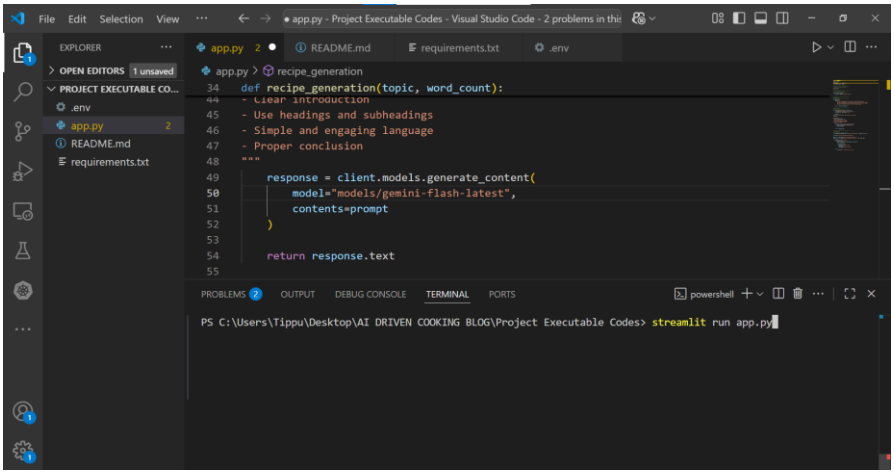
Final Model	**Reasoning**
Gemini Flash Lite (models/gemini-flash-latest)	Selected due to its fast response time, efficient resource usage, high-quality text generation, and seamless integration with Streamlit for real-time recipe blog generation.

6. RESULTS

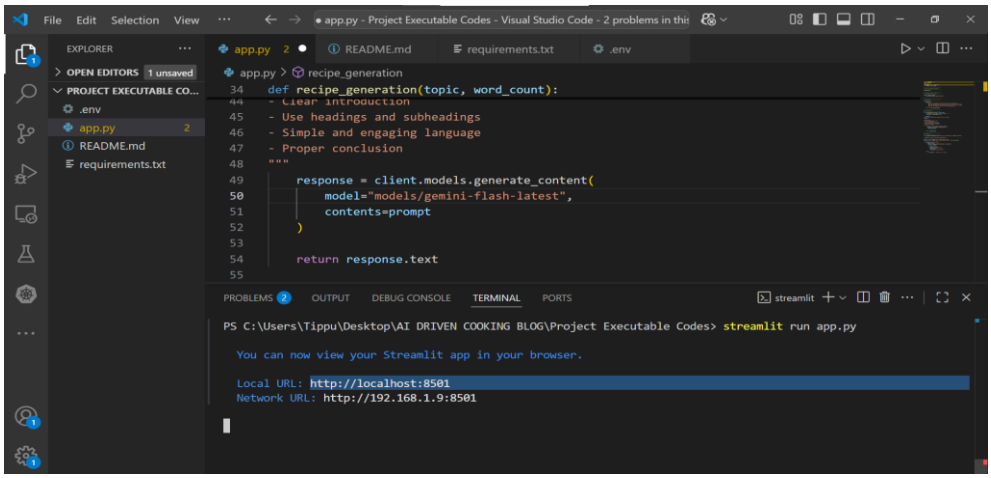
6.1. Output Screenshots

The complete execution of Flavour Fusion: AI-Driven Recipe Blogging application is represented step by step in the following screenshots.

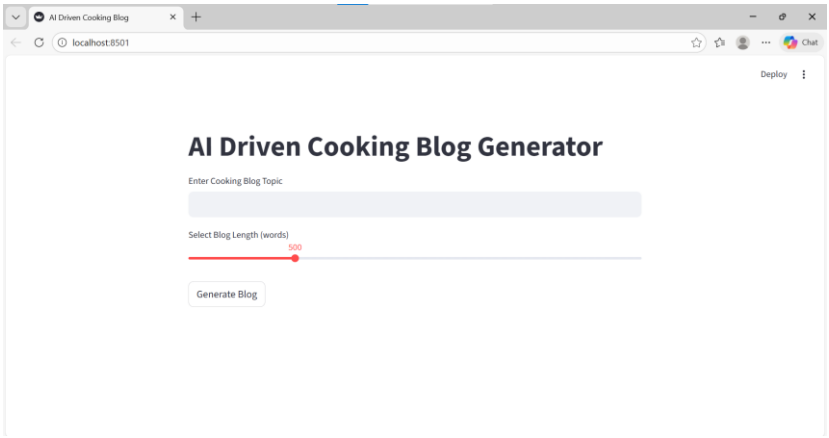
Step 1: To run the Streamlit Application we have to use the command streamlit run app.py in the terminal in path where the app.py file is located.



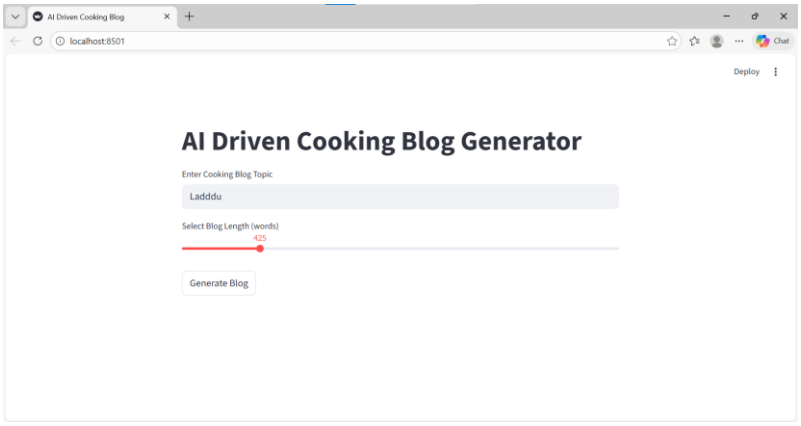
Step 2: After running the command in terminal, the code will get executed and the webpage will open directly. Another way to open webpage is that a localhost link will get generated in the terminal, we can access the webpage using that link.



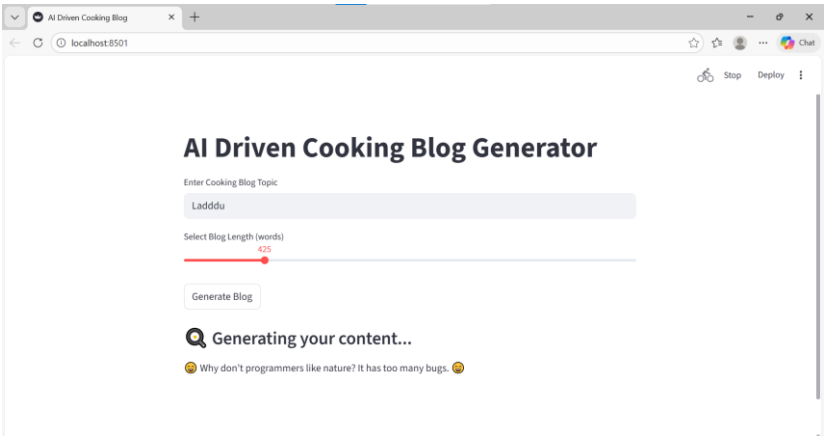
Step 3: The Streamlit webpage opens as shown in the figure given below. This is an automated webpage. No secondary HTML codes required to build this webpage. Python code itself consists the webpage building code.

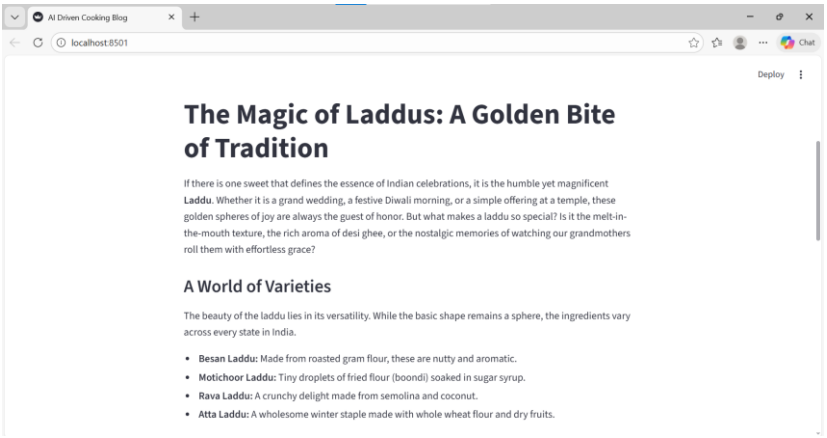


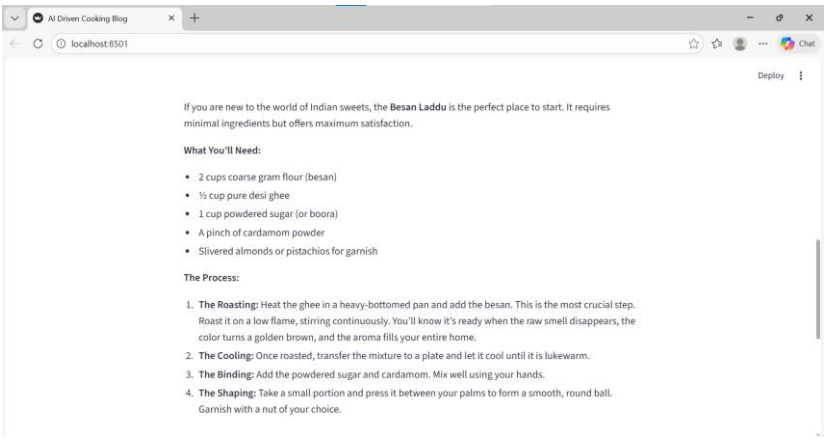
Step 3: The user has to give inputs in the website such as a Recipe Name and Number of Words. The Number of words means with how many words the recipe should be generated. After entering the required details, the user should click on Generate Recipe button to generate the recipe. Here I chose the Chocolate Cake Recipe with 400 words count.

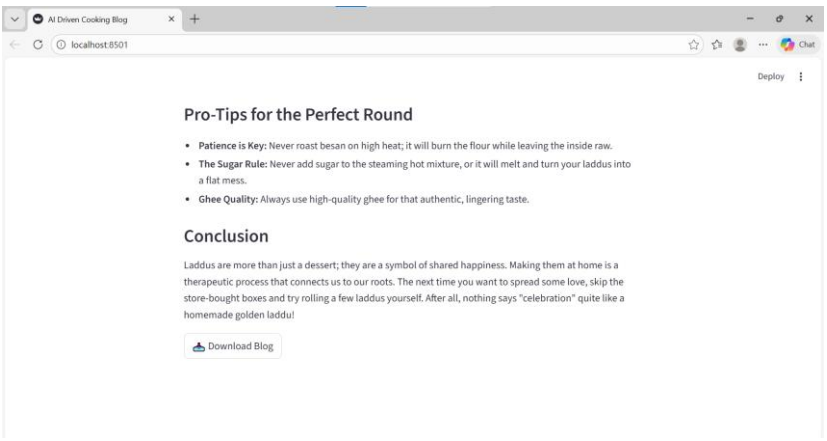


Step 4: After clicking the Generate Recipe button, in fraction of seconds a simple joke will be generated as shown in the below figure to engage the users if recipe generation get delayed and also the user can download the blog too. The Food Recipe will be generated based on the user inputs as shown in the following four images.


AI Driven Cooking Blog
localhost:8501
Chat
Stop
Deploy
AI Driven Cooking Blog Generator
Enter Cooking Blog Topic
Ladddu
Select Blog Length (words)
425
Generate Blog
Generating your content...
Why don't programmers like nature? It has too many bugs.


AI Driven Cooking Blog
localhost:8501
Chat
Deploy
The Magic of Laddus: A Golden Bite of Tradition
If there is one sweet that defines the essence of Indian celebrations, it is the humble yet magnificent Laddu. Whether it is a grand wedding, a festive Diwali morning, or a simple offering at a temple, these golden spheres of joy are always the guest of honor. But what makes a laddu so special? Is it the melt-in-the-mouth texture, the rich aroma of desi ghee, or the nostalgic memories of watching our grandmothers roll them with effortless grace?
A World of Varieties
The beauty of the laddu lies in its versatility. While the basic shape remains a sphere, the ingredients vary across every state in India.
Besan Laddu: Made from roasted gram flour, these are nutty and aromatic.
Motichoor Laddu: Tiny droplets of fried flour (boondi) soaked in sugar syrup.
Rava Laddu: A crunchy delight made from semolina and coconut.
Atta Laddu: A wholesome winter staple made with whole wheat flour and dry fruits.


AI Driven Cooking Blog
localhost:8501
Chat
Deploy
If you are new to the world of Indian sweets, the Besan Laddu is the perfect place to start. It requires minimal ingredients but offers maximum satisfaction.
What You'll Need:
2 cups coarse gram flour (besan)
½ cup pure desi ghee
1 cup powdered sugar (or boora)
A pinch of cardamom powder
Slivered almonds or pistachios for garnish
The Process:
1. The Roasting: Heat the ghee in a heavy-bottomed pan and add the besan. This is the most crucial step. Roast it on a low flame, stirring continuously. You'll know it's ready when the raw smell disappears, the color turns a golden brown, and the aroma fills your entire home.
2. The Cooling: Once roasted, transfer the mixture to a plate and let it cool until it is lukewarm.
3. The Binding: Add the powdered sugar and cardamom. Mix well using your hands.
4. The Shaping: Take a small portion and press it between your palms to form a smooth, round ball. Garnish with a nut of your choice.


AI Driven Cooking Blog
localhost:8501
Chat
Deploy
Pro-Tips for the Perfect Round
Patience is Key: Never roast besan on high heat; it will burn the flour while leaving the inside raw.
The Sugar Rule: Never add sugar to the steaming hot mixture, or it will melt and turn your laddus into a flat mess.
Ghee Quality: Always use high-quality ghee for that authentic, lingering taste.
Conclusion
Laddus are more than just a dessert; they are a symbol of shared happiness. Making them at home is a therapeutic process that connects us to our roots. The next time you want to spread some love, skip the store-bought boxes and try rolling a few laddus yourself. After all, nothing says "celebration" quite like a homemade golden laddu!
Download Blog

7. ADVANTAGES AND DISADVANTAGES

Advantages

- Saves time by automatically generating detailed recipe blogs
- Reduces manual effort for food bloggers and users
- Generates customized content based on user input
- User-friendly interface built using Streamlit
- Fast and efficient content generation using a pre-trained AI model
- No requirement for dataset collection or model training

Disadvantages

- Requires an active internet connection to access the AI model
- Depends on third-party AI APIs for content generation
- Limited to text-based recipe content only
- Output quality depends on the clarity of user input

8. CONCLUSION

The Flavour Fusion: AI-Driven Recipe Blogging project successfully demonstrates how generative AI can be used to automate recipe blog creation. The application allows users to generate customized and well-structured recipe content by providing a topic and word count, reducing the time and effort required for manual writing. By integrating a pre-trained AI model with a user-friendly Streamlit interface, the project delivers fast, efficient, and high-quality results, making it a useful tool for food bloggers and cooking enthusiasts.

9. FUTURE SCOPE

The Flavour Fusion project can be enhanced further by adding support for multiple languages to reach a wider audience. Future improvements may include generating recipe images along with text, adding user accounts to save favorite recipes, and providing personalized recipe recommendations. The application can also be extended to support voice-based input and mobile platform deployment, making it more accessible and user-friendly.

10. APPENDIX

10.1. Source Code

The source code for the Flavour Fusion: AI-Driven Recipe Blogging project includes the implementation of the Streamlit user interface, integration of the Gemini Flash Lite model using the Google Generative AI API, recipe blog generation logic, and the programmer joke feature. The code is written in Python and follows a modular and readable structure.

CODE

```
import streamlit as st

import random

import os

from dotenv import load_dotenv

from google import genai


# --------------------------------

# Load environment variables

# --------------------------------

load_dotenv()
```

```
# --------------------------------
# Configure Gemini Client
# --------------------------------
client = genai.Client(
    api_key=os.getenv("GEMINI_API_KEY")
)


# --------------------------------
# Random Joke
# --------------------------------
def get_joke():
    jokes = [
        "Why don't programmers like nature? It has too many bugs.",
        "Why do programmers prefer dark mode? Because light attracts bugs.",
        "Why did the developer go broke? Because he used up all his cache.",
        "Why was the computer cold? It left its Windows open."
    ]
    return random.choice(jokes)


# --------------------------------
# Blog Generator
# --------------------------------
def recipe_generation(topic, word_count):
    st.write("### 🔍 Generating your content...")
```

```
    st.write(f"😄 {get_joke()} 😄")


    prompt = f"""
Write a well-structured cooking blog on the topic:
"{topic}"

Requirements:
- Around {word_count} words
- Clear introduction
- Use headings and subheadings
- Simple and engaging language
- Proper conclusion
"""
    response = client.models.generate_content(
        model="models/gemini-flash-latest",
        contents=prompt
    )


    return response.text


# --------------------------------
# Streamlit UI
# --------------------------------
st.set_page_config(page_title="AI Driven Cooking Blog", layout="centered")
```

```
st.title("AI Driven Cooking Blog Generator")


topic = st.text_input("Enter Cooking Blog Topic")

word_count = st.slider("Select Blog Length (words)", 200, 1500, 500)


if st.button("Generate Blog"):
    if topic.strip():
        output = recipe_generation(topic, word_count)
        st.write(output)


        st.download_button(
            label=" 📥 Download Blog",
            data=output,
            file_name="cooking_blog.txt"
        )
    else:
        st.warning(" ⚠️ Please enter a topic")
```

10.2. Github & Project Demo Link

Github Link : https://github.com/chinna-ch

DemoVideo:https://drive.google.com/file/d/1G8qmglvjposxqo5sTo6ar0ZR2qyFgdLL/view?usp=sharing