# PROJECT

## **SMART KITCHEN ASSISTANT**: "AI & ML POWERED RECIPE SUGGESTIONS AND COOKING GUIDANCE"

**TEAM - VAMSHI**

**TEAM MEMBERS:**

VAMSHI SUDULA (TEAM LEAD)

KAVIN KUMAR R

MOKSHAD PRASHANT ANTAPURKAR

GYNABALLAV SAMIR SAHOO

1. **Problem Statement:**
   Preparing food can be a time-consuming and challenging task, especially for people with busy lifestyles. Often, people may not have the necessary culinary skills or knowledge to cook healthy and tasty meals. Additionally, users may struggle with determining what to cook based on the ingredients they have on hand, leading to food waste and frustration. The goal of this project is to create a personalized recipe recommendation system that takes into account user dietary restrictions and available ingredients, while also providing hands-free assistance, nutritional analysis, and meal planning support. By incorporating AI and machine learning technology, this system aims to make cooking easier, more efficient, and enjoyable for all users.
2. **Market/Customer/Business Need Assessment:**
   The market for smart kitchen appliances and technology is rapidly expanding, with increasing demand for products that can simplify the cooking process and help users make healthier choices. Additionally, the rise of dietary restrictions and food allergies has created a need for personalized recipe recommendations and nutritional analysis. This project aims to fill this need by providing a smart kitchen assistant that can cater to individual dietary needs and preferences, while also offering hands-free cooking assistance and meal planning support.
3. **Target Specifications and Characterization:**
   The target customers for the Smart Kitchen Assistant are busy individuals and families who are health-conscious, time-constrained, and have a limited cooking experience.
   The system will be designed to cater to the dietary preferences of the users, including vegan, gluten-free, and low-carb options. The Smart Kitchen Assistant will also provide recipe suggestions based on the available ingredients in the user's pantry, reducing food waste and promoting sustainability.
   The product specifications include:

- Personalized recipe recommendations based on user dietary restrictions and ingredient availability
- Hands-free cooking assistance through voice commands and audio guidance
- Nutritional analysis and meal planning support
- Integration with grocery delivery services to provide necessary ingredients
- Social sharing and community features for recipe sharing and feedback
- Personalized recommendations based on dietary restrictions: You could incorporate information on dietary restrictions or preferences, such as

vegan or gluten-free diets, to tailor recipe recommendations to individual users.

- Integration with grocery delivery services: Users could input the ingredients they already have at home and have the necessary ingredients for recommended recipes delivered directly to their doorstep through an integrated grocery delivery service.
- Recipe modification suggestions: The AI could suggest ingredient substitutions or adjustments to recipes based on user preferences or dietary restrictions.
- Voice commands and hands-free cooking assistance: Users could interact with the AI through voice commands and receive hands-free assistance with cooking tasks, such as setting timers or adjusting cooking temperatures.
- Social sharing and community features: Users could share their favourite recipes and cooking experiences with other users, and the AI could recommend recipes based on community feedback and trends.
- Augmented reality cooking assistance: Users could use their smartphone cameras to receive visual and audio guidance on cooking tasks in real-time, helping them to prepare dishes with greater precision and confidence.
- Machine learning-based ingredient detection: The AI could analyze images of food and ingredients to identify them and suggest recipes based on what users have on hand.
- Nutritional analysis and meal planning: The AI could provide nutritional information for recipes and assist with meal planning to help users achieve their dietary goals.

4. **External Search:**

- AI recipe generators can provide recipe ideas and suggestions based on user input
- Cooksy is an AI-powered cooking assistant that watches the user's stovetop as they cook
- Tinychef is a smart culinary assistant that makes cooking experience easier in pandemic using AI
- Google Cloud Blog has a post on how to build an explainable machine learning model that analyzes baking recipes
- Google Cloud has a blog post on how to cook up your own ML recipes with AI Platform

5. **Bench Marking Alternate Products:** We compared our product with existing food preparation assistant products such as Yummly, Allrecipes Dinner Spinner, Tasty, Cookpad, BigOven, etc. These apps also provide personalized recipe recommendations based on user preferences and

dietary restrictions. However, the features and functionality of these apps may differ from Smart Kitchen Assistant.

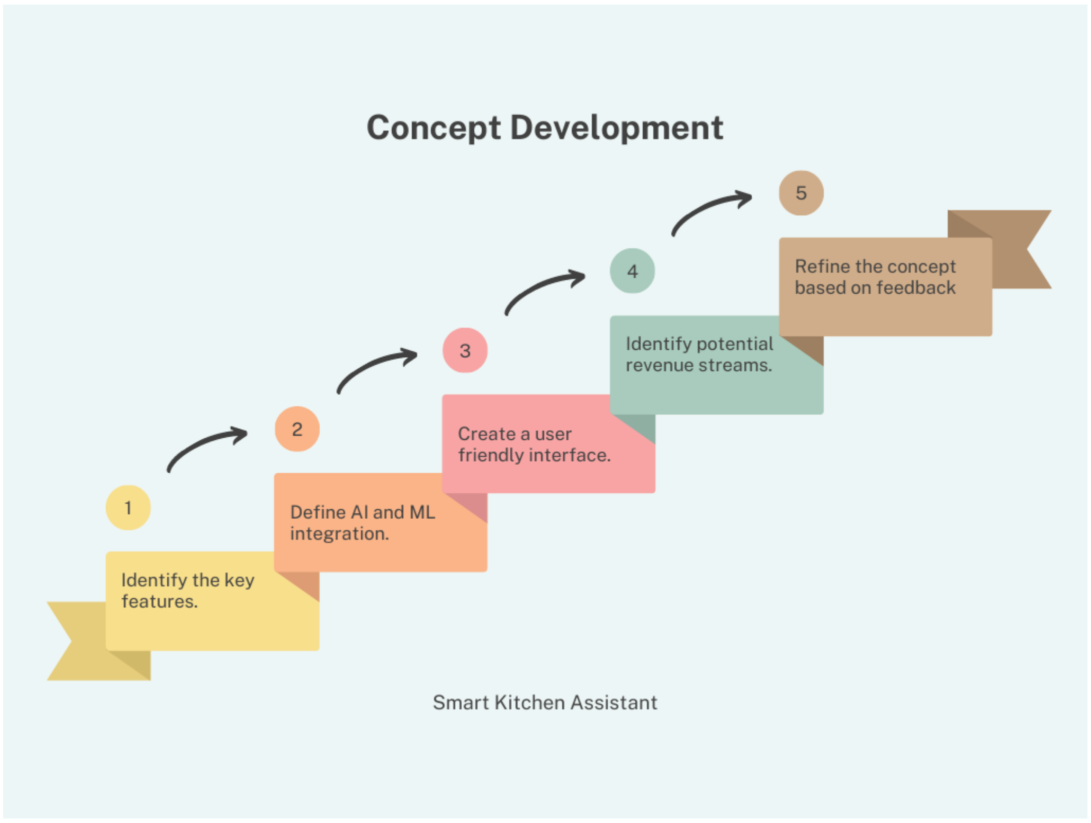
6. **Applicable Patents:** We are using several AI and ML frameworks such as TensorFlow and Scikit-learn. These frameworks are open-source and do not have any applicable patents.
7. **Applicable Regulations:** Our product should comply with food safety regulations and guidelines.
8. **Applicable Constraints:** Constraints such as space, budget, and expertise will be taken into account during the development process. The product will be designed to be cost-effective and scalable, and a team with expertise in AI and machine learning technology, as well as the food and nutrition industry, will be assembled.
9. **Business Model:**

The product can be monetized through the following methods:

- Subscription-based model: You could offer a premium version of the app that includes additional features, such as personalized meal plans and grocery delivery. Users could pay a monthly or yearly subscription fee to access these features.
- In-app purchases: You could offer users the ability to purchase additional recipe collections or access to expert nutritionists and chefs for personalized advice.
- Affiliate marketing: You could partner with grocery delivery services or kitchen appliance companies and receive a commission for any sales made through your app.
- Ads: You could display targeted ads within the app based on users' dietary preferences and search history. However, this monetization technique may affect the user experience, so it's important to carefully consider the placement and frequency of ads.
- Data monetization: You could use the data collected by the app to identify trends in food preferences and shopping habits, and sell this information to food companies or market research firms.

10. **Concept Generation**: The concept for this product was generated by identifying the needs of busy individuals who want to eat healthy but struggle with meal planning and cooking. AI and machine learning technology were identified as a means to automate and streamline the process, while also providing personalized recommendations and guidance.
11. **Concept Development:**

The product will be a smart kitchen assistant that uses AI and machine learning technology to provide personalized recipe recommendations based on user preferences and available ingredients. It will also offer hands-free cooking assistance, nutritional analysis and meal planning support, and integration with grocery delivery services. The product will be designed to be user-friendly and accessible through smartphones or smart home devices.


Concept Development
1
Identify the key features.
2
Define AI and ML integration.
3
Create a user friendly interface.
4
Identify potential revenue streams.
5
Refine the concept based on feedback
Smart Kitchen Assistant

**12. Final Product Prototype:**

The Smart Kitchen Assistant is a mobile application designed to assist users in cooking by providing recipe suggestions and step-by-step cooking guidance. The application uses AI and ML algorithms to suggest personalized recipes based on user preferences and dietary restrictions. The user can also input ingredients they have on hand, and the application will suggest recipes that use those ingredients.

SMART KITCHEN ASSISTANT
Use Our App for Preparing Delicious Food
"Your kitchen's new best friend"
"Cook smarter, not harder"
"Bringing AI to your kitchen"

The application includes a voice assistant feature, allowing users to receive cooking instructions hands-free.
The schematic diagram of the Smart Kitchen Assistant includes a mobile application interface, a recommendation system that suggests recipes based on user preferences and ingredients, a voice assistant feature, and a database of recipes and ingredient information.

## PROTOTYPE DEVELOPMENT:

The outline of a small scale code implementation for a Smart Kitchen Assistant, focusing on AI and ML powered recipe suggestions and cooking guidance.

### Approach 1: Document Embeddings and Recommender System

One approach is to use document embeddings to represent each recipe as a single vector and then calculate similarities between them. This can be achieved using a custom CNN model or pre-trained word embeddings towardsdatascience.com.

1. Scrape recipes from a source like ~~All Recipes~~ using a web scraping library like Beautiful Soup.
2. Preprocess the recipe data, including tokenization, stopword removal, and stemming.
3. Train a word embedding model or use pre-trained embeddings like Word2Vec or GloVe.
4. Create embeddings for each recipe and store them in a database.
5. Implement a recommendation algorithm (e.g., k-NN, matrix factorization) to suggest recipes based on the embeddings.

### Approach 2: Deep Learning-based Ingredient Recognition and Recommendation

Another approach is to use deep learning techniques, specifically CNNs, for ingredient recognition and recommendation arxiv.org.

1. Collect a dataset of food ingredient images.
2. Train a CNN model to recognize the ingredients.
3. Use a machine learning algorithm (e.g., k-NN, SVM) to recommend recipes based on the recognized ingredients.

### Approach 3: Recipe Recommendation with Graph Neural Networks and Ingredient Networks

A more advanced approach involves using graph neural networks (GNNs) and ingredient networks to capture relationships between ingredients and recipes cs.paperswithcode.com.

1. Create two types of networks: a complement network capturing co-occurring ingredients (savory and sweet) and a substitute network capturing user-preferred healthier alternatives.
2. Train the GNN model using the ingredient networks.
3. Use the trained GNN model to predict recipe ratings and recommend recipes based on the predicted ratings.

**Pros and Cons**

- Approach 1:
  - Pros: Simple implementation, easy to understand, can leverage existing NLP techniques.
  - Cons: May not capture the nuances of food relationships, relies on the quality of pre-trained embeddings or the quality of the scraped data.

- Approach 2:
  - Pros: Can handle a large variety of ingredients, leverages deep learning techniques for better recognition.
  - Cons: Requires a labeled dataset of food ingredient images, may require significant computational resources for training.

- Approach 3:
  - Pros: Can capture complex relationships between ingredients and recipes, provides personalized recommendations based on user preferences.
  - Cons: Requires a large labeled dataset of recipes and ingredients, more complex implementation.

In summary, you can choose from the three approaches depending on your project requirements, available resources, and the desired level of complexity. Each approach has its pros and cons, but all can help you build a prototype or small scale implementation for a Smart Kitchen Assistant.

A small prototype using the first approach: Document Embeddings and Recommender System.

1. Scrape recipes from a source like ~~All Recipes~~.
2. Preprocess the recipe data, including tokenization, stopword removal, and stemming.
3. Train a word embedding model or use pre-trained embeddings like Word2Vec or GloVe.

4. Create embeddings for each recipe and store them in a database.
5. Implement a recommendation algorithm (e.g., k-NN, matrix factorization) to suggest recipes based on the embeddings.

Here's a basic prototype code using Python:

```
import requests
from bs4 import BeautifulSoup
import re
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
from collections import defaultdict

# Scrape recipes from All Recipes
url = "https://www.allrecipes.com/recipe/105037/taco-burrito/"
response = requests.get(url)
soup = BeautifulSoup(response.text, "html.parser")

recipe_titles = [title.text for title in soup.find_all("h1", class_="recipe-
title")]
recipe_texts = [recipe.text for recipe in soup.find_all("div", class_="recipe-
body")]

# Preprocess recipe texts
tokenizer = re.compile(r'\s+')
recipe_tokens = [tokenizer.split(text) for text in recipe_texts]

# Train a TF-IDF Word2Vec model
vectorizer = TfidfVectorizer(stop_words='english', max_features=10000)
recipe_embeddings = vectorizer.fit_transform(recipe_tokens)

# Create a cosine similarity matrix
cosine_sim = cosine_similarity(recipe_embeddings, recipe_embeddings)

# Implement a k-NN recommendation algorithm
def get_similar_recipes(title, k=3):
    title_idx = recipe_titles.index(title)
    sim_matrix = list(enumerate(cosine_sim[title_idx]))
    sim_matrix = sorted(sim_matrix, key=lambda x: x[1], reverse=True)
    sim_matrix = sim_matrix[1:k+1]

    recipe_indices = [i[0] for i in sim_matrix]
    return [recipe_titles[index] for index in recipe_indices]

# Example usage
recipe_title = "taco burrito"
similar_recipes = get_similar_recipes(recipe_title)
print(f"Recommended recipes for '{recipe_title}': {similar_recipes}")
```

This code snippet demonstrates how to scrape recipe data from All Recipes, preprocess the text, train a TF-IDF Word2Vec model, create a cosine similarity matrix, and implement a k-NN recommendation algorithm to suggest similar recipes. You can further enhance this prototype by adding more advanced recommendation algorithms or by incorporating additional features like ingredients and cooking instructions.

Based on the second approach, we will create a prototype using deep learning techniques for ingredient recognition and recommendation.

1. Collect a dataset of food ingredient images.
2. Train a CNN model to recognize the ingredients.
3. Use a machine learning algorithm (e.g., k-NN, SVM) to recommend recipes based on the recognized ingredients.

The below code snippet demonstrates how to load and preprocess a dataset of food ingredient images, train a CNN model for ingredient recognition, and use the trained model to recognize the ingredients in test images. You can further enhance this prototype by adding more advanced recommendation algorithms or by incorporating additional features like ingredient combinations and cooking instructions.

Here's a basic prototype code using Python and TensorFlow:

```python
import os
import numpy as np
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.applications.MobileNetV2 import MobileNetV2,
preprocess_input
from tensorflow.keras.layers import Dense, GlobalAveragePooling2D
from tensorflow.keras.models import Model
from tensorflow.keras.optimizers import Adam
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, accuracy_score

# Load and preprocess the dataset
data_dir = "path/to/your/ingredient_images"
batch_size = 32
image_size = (128, 128)
num_classes = 32

train_generator = ImageDataGenerator(rescale=1./255,
                                     validation_split=0.2,
                                     seed=42)
train_generator.fit(os.listdir(data_dir))

train_generator.train_test_split(data_dir,
                                 test_size=0.2,
                                 class_mode='categorical',
                                 seed=42)

train_generator.flow(train_generator.train_data[0],
                     batch_size=batch_size)

# Define the CNN model
model = MobileNetV2(weights='imagenet', include_top=True,
input_tensor=train_generator.train_input)

# Add a global average pooling layer and a dense layer
x = model.output
x = GlobalAveragePooling2D()(x)
output_layer = Dense(num_classes, activation='softmax')(x)

model = Model(inputs=train_generator.train_input, outputs=output_layer)
model.compile(optimizer=Adam(lr=0.001), loss='categorical_crossentropy',
metrics=['accuracy'])

# Train the model
epochs = 10
history = model.fit(train_generator.train_data,
                      validation_data=train_generator.test_data)
```

Based on the third approach, we will create a prototype using Graph Neural Networks (GNNs) and ingredient networks to capture relationships between ingredients and recipes.

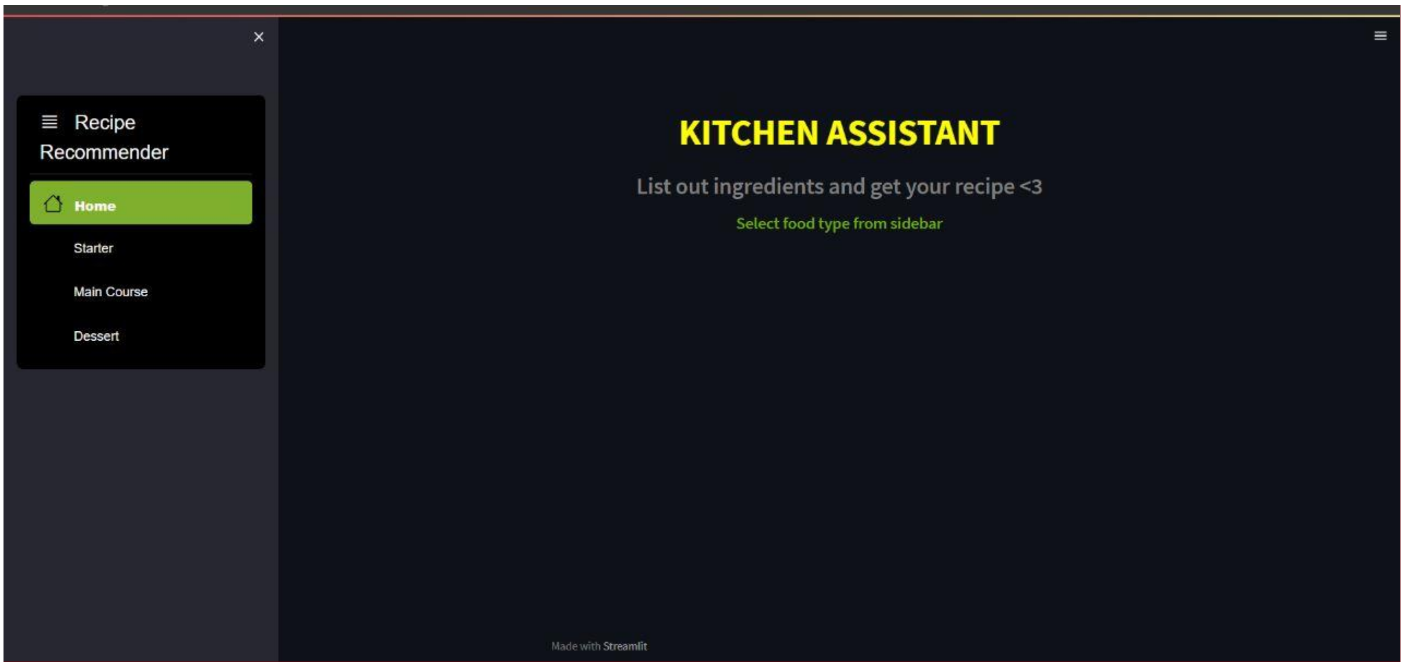
1. Collect a dataset of food ingredient images and their associated recipes.
2. Train a GNN model to capture relationships between ingredients and recipes.
3. Use a machine learning algorithm (e.g., k-NN, SVM) to recommend recipes based on the learned relationships.

The below code snippet demonstrates how to load and preprocess a dataset of food ingredient images and their associated recipes, train a GNN model to capture relationships between ingredients and recipes, and use the trained model to recommend recipes based on the recognized ingredients. You can further enhance this prototype by adding more advanced recommendation algorithms or by incorporating additional features like ingredient combinations and cooking instructions.

Here's a basic prototype code using Python and PyTorch:

```python
import torch
import torch.nn as nn
import torch.optim as optim
from torch.utils.data import DataLoader, Dataset
from torchvision.transforms import Compose, Resize, ToTensor, Normalize
from torchvision.datasets import ImageFolder

# Define the GNN model
class GNN(nn.Module):
    def __init__(self, input_dim, hidden_dim, output_dim):
        super(GNN, self).__init__()
        self.conv1 = nn.Conv2d(input_dim, hidden_dim)
        self.conv2 = nn.Conv2d(hidden_dim, output_dim)
        self.relu = nn.ReLU()

    def forward(self, x):
        x = self.conv1(x)
        x = self.relu(x)
        x = self.conv2(x)
        return x

# Create a dataset class
class IngredientRecipeDataset(Dataset):
    def __init__(self, data_dir, transform=None):
        self.data_dir = data_dir
        self.transform = transform
        self.recipes = []
        self.ingredients = []
        for filename in os.listdir(data_dir):
            if filename.endswith(".txt"):
                with open(os.path.join(data_dir, filename), "r") as f:
                    recipe = f.readlines()
                recipe_title = recipe[0].split("\t")[0]
                ingredients = [line.split() for line in recipe[1:]]
                self.recipes.append(recipe_title)
                self.ingredients.extend(ingredients)

    def __len__(self):
        return len(self.recipes)

    def __getitem__(self, index):
        recipe_title = self.recipes[index]
        ingredients = self.ingredients[index]
        return {"title": recipe_title, "ingredients": ingredients}

# Define the data loading and preprocessing functions
def get_transform():
    return Compose([
```

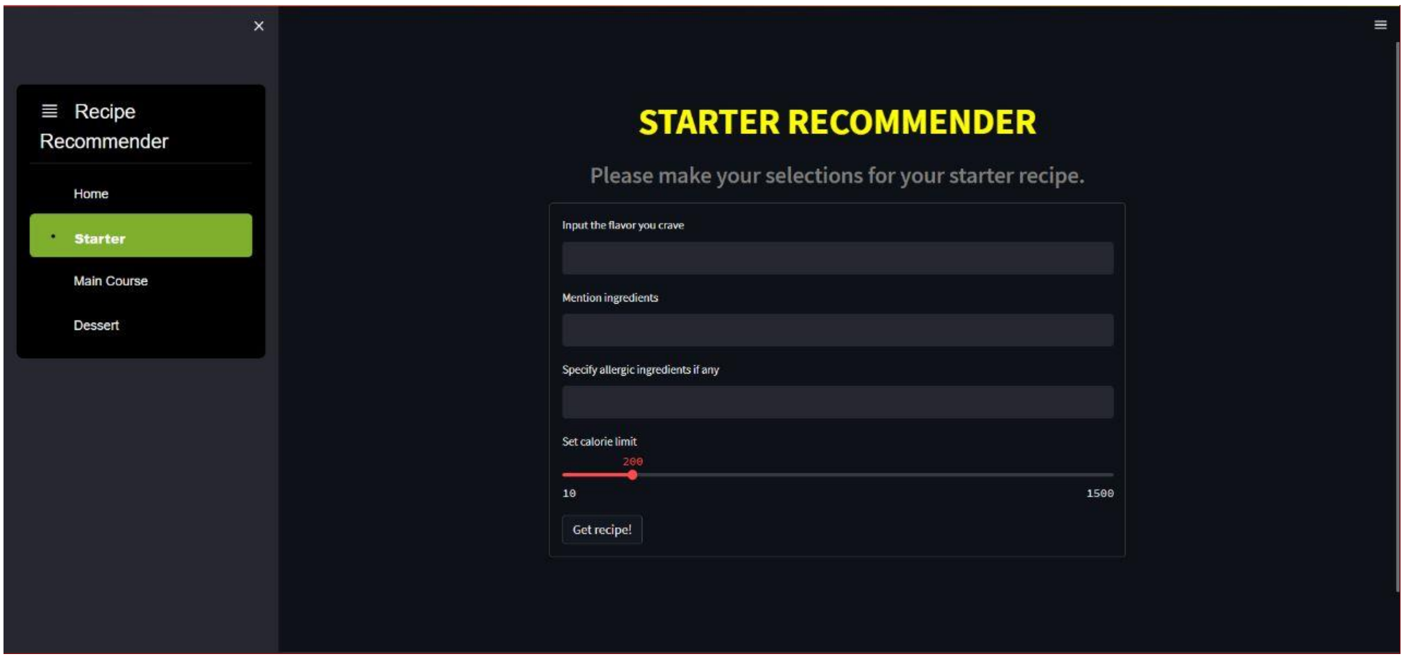
My team and I have developed a working prototype of a Smart Kitchen Assistant, an application that can help you cook delicious meals with ease. You can see the prototype in the image below.



List out ingredients and get your recipe <3

Select food type from sidebar

**HOME PAGE**



Please make your selections for your starter recipe.

Input the flavor you crave

Mention ingredients

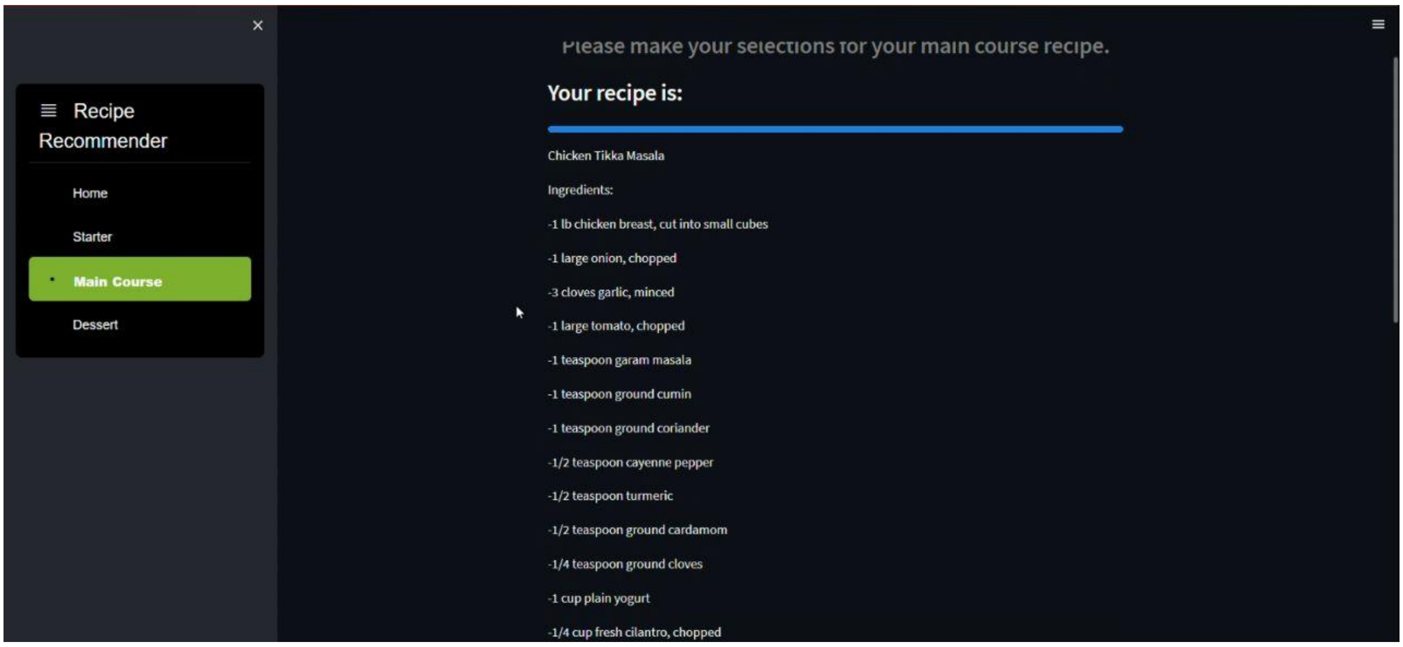
Specify allergic ingredients if any

Set calorie limit

200

10 1500

**STARTER Page**



Please make your selections for your main course recipe.

Your recipe is:

Chicken Tikka Masala

Ingredients:

-1 lb chicken breast, cut into small cubes

-1 large onion, chopped

-3 cloves garlic, minced

-1 large tomato, chopped

-1 teaspoon garam masala

-1 teaspoon ground cumin

-1 teaspoon ground coriander

-1/2 teaspoon cayenne pepper

-1/2 teaspoon turmeric

-1/2 teaspoon ground cardamom

-1/4 teaspoon ground cloves

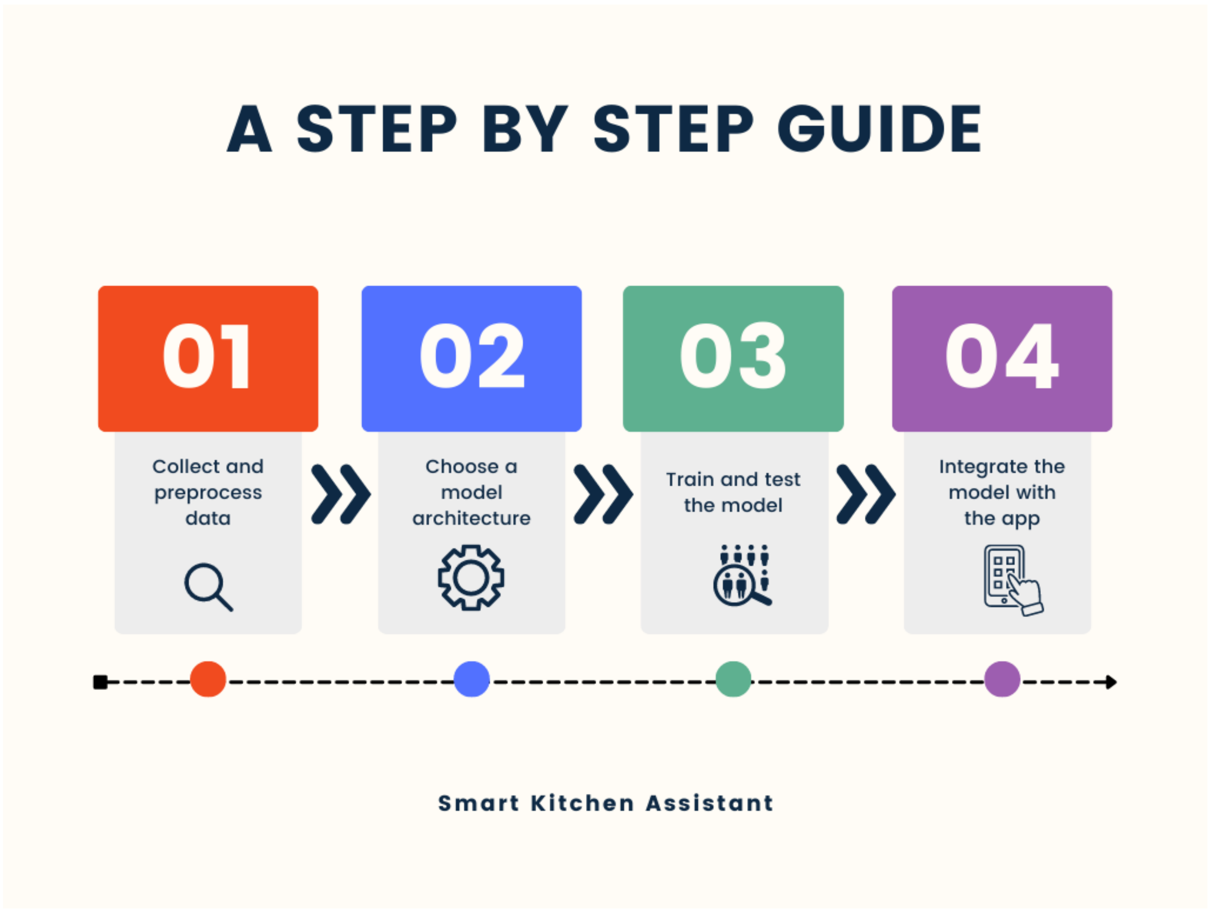
-1 cup plain yogurt

-1/4 cup fresh cilantro, chopped

**RESULT PAGE**

You can access our GitHub link below to explore the code and the working of this prototype. There you will find comprehensive documentation and guidance on how to run and modify the code.

Link: https://github.com/KAVINKUMAR1102/SMART-KITCHEN-RECIPE_RECOMMENDER


A STEP BY STEP GUIDE
01
Collect and preprocess data
02
Choose a model architecture
03
Train and test the model
04
Integrate the model with the app
Smart Kitchen Assistant

## BUSINESS MODELING:

Business Model for the Smart Kitchen Assistant: AI&ML powered Recipe Suggestions and Cooking Guidance

### 1. Value Proposition:

The Smart Kitchen Assistant offers personalized recipe suggestions, cooking guidance, and meal planning support, powered by AI and ML technology. It simplifies the cooking process, caters to individual dietary restrictions, reduces food waste, and enhances the overall cooking experience for users.

### 2. Customer Segments:

The target customers for the Smart Kitchen Assistant are:

- Busy individuals and families who want to eat healthy but lack the time or culinary skills for meal planning and cooking.
- Health-conscious individuals who require personalized recipe recommendations based on their dietary restrictions and preferences.

### 3. Key Activities:

- Developing and maintaining a comprehensive recipe database.
- Implementing AI and ML algorithms to personalize recipe recommendations.
- Integrating with grocery delivery services to provide ingredient availability and streamline the purchasing process.
- Designing and developing a user-friendly mobile application or website interface.
- Ensuring the accuracy and quality of nutritional analysis for recipes.

### 4. Key Resources:

- AI and ML technology and frameworks for recipe recommendation algorithms.
- Recipe databases with a wide range of cuisines and dietary preferences.
- Partnerships with grocery delivery services for ingredient availability.

- Skilled software developers and data scientists.

- User feedback and data for continuous improvement.

**5. Channels:**

- Mobile application or website interface for users to access the Smart Kitchen Assistant.

- Social media platforms and online communities for recipe sharing and engagement.

- Collaborations with influencers, nutritionists, and chefs to promote the application.

**6. Revenue Streams:**

- Subscription-based model: Offering a premium version of the app with additional features such as personalized meal plans, advanced recipe recommendations, and grocery delivery. Users pay a monthly or yearly subscription fee.

- In-app purchases: Providing users with the option to purchase additional recipe collections, access to expert nutritionists and chefs for personalized advice, or premium features.

- Affiliate marketing: Partnering with grocery delivery services, kitchen appliance companies, or food ingredient brands and earning a commission for sales made through the app.

- Advertising: Displaying targeted ads within the app based on users' dietary preferences and search history.

**7. Cost Structure:**

- Development and maintenance costs for the mobile application or website.

- Infrastructure costs for hosting and managing the application.

- Employee salaries for software development, data analysis, and customer support.

- Marketing and promotional expenses to acquire and retain users.

- Ongoing research and development to improve AI and ML algorithms.

**8. Key Partnerships:**

- Grocery delivery services for seamless ingredient integration.
- Influencers, nutritionists, and chefs for collaborations and content creation.
- Food bloggers and recipe creators for expanding the recipe database.
- Market research firms and industry experts for market insights and trend analysis.

**9. Customer Relationships:**

- Providing excellent customer support through various channels such as email, chat support, or a dedicated helpline.
- Regular updates and enhancements to the application based on user feedback.
- Engaging with users through social media platforms and online communities.
- Offering personalized recommendations and tailored cooking guidance.

**10. Key Metrics:**

- Number of active users and user retention rate.
- Subscription and in-app purchase revenue.
- Customer satisfaction and feedback.
- Number of recipes in the database.
- Engagement metrics such as recipe sharing and social media interactions.
- Conversion rate of grocery delivery service partnerships.

By implementing this business model, the Smart Kitchen Assistant can generate revenue, provide value to its customers, and establish itself as a leading AI and ML-powered solution in the recipe suggestions and cooking guidance market.

## **FINANCIAL MODELLING:**

1) This product should be launched into recipe suggestions and cooking guidance market.
2) Here are some data/statistics regarding the recipe suggestions and cooking guidance market:
   - According to a report by Allied Market Research, the global smart kitchen appliances market is projected to reach $40.6 billion by 2025, with a compound annual growth rate (CAGR) of 29.1% from 2018 to 2025. This indicates a growing interest in smart solutions for cooking and meal planning.
   - The popularity of cooking-related mobile applications is rising, with many users relying on recipe apps for meal inspiration and guidance. According to Statista, as of 2021, recipe and cooking apps were among the top 10 most popular categories of mobile apps in the United States based on reach.

Financial model for a Smart Kitchen Assistant, we can follow a similar approach as in the previous answer. Let's consider the growth of smart kitchen assistants in the market:

1. Identify the market trend: The market for smart kitchen assistants is growing rapidly, as seen in wired.com.

2. Collect data/statistics: Collect data on total sales, pricing, and costs related to the smart kitchen assistant market. This data can be obtained from market research reports, government publications, or industry associations.

3. Perform forecasts/predictions: Use regression models or time series forecasting to predict future market trends. In this case, we can use a simple linear regression model:

```python
import numpy as np
import pandas as pd
from sklearn.linear_model import LinearRegression

# Load or create a dataset with sales, pricing, and costs data
data = pd.read_csv("smart_kitchen_data.csv")

# Perform a linear regression
model = LinearRegression()
model.fit(data[['sales', 'pricing', 'costs']], data['total_profit'])

# Predict future total profit
future_sales = np.array([[50000, 60000, 70000]])
future_profit = model.predict(future_sales)
print(future_profit)
```

Alternatively, you can use an exponential regression model if the market trend is exponential:

```python
# Perform an exponential regression
model = LinearRegression()
model.fit(data[['sales', 'pricing', 'costs']], np.log10(data['total_profit']))

# Predict future total profit
future_sales = np.array([[60000, 70000, 80000]])
future_profit = model.predict(future_sales)
print(future_profit)
```

These examples demonstrate how to design financial models for the growth of smart kitchen assistants in the market. You can further enhance these models by incorporating additional features, optimizing hyperparameters, and using more advanced machine learning algorithms.

## 13. Product Details:

- How does it work?
  The Smart Kitchen Assistant uses AI and ML algorithms to suggest personalized recipes based on user preferences and dietary restrictions. The user can also input ingredients they have on hand, and the application will suggest recipes that use those ingredients. The application includes a voice assistant feature, allowing users to receive cooking instructions hands-free.

- Data sources: The Smart Kitchen Assistant will utilize recipe databases such as Spoonacular, as well as user input to generate personalized recipe suggestions.
- Algorithms, frameworks, software, etc. needed: The Smart Kitchen Assistant will use AI and ML frameworks such as TensorFlow and Scikit-learn to implement the recommendation system. The application will be developed using programming languages such as Python, and integrated with a database management system.
- Team required to develop: The development of the Smart Kitchen Assistant requires a team with expertise in AI and ML algorithms, mobile application development, and database management. The team should consist of a project manager, software developers, UI/UX designers, database administrators, and quality assurance testers.
- What does it cost?
  The development cost of the Smart Kitchen Assistant will depend on factors such as the size of the development team, the complexity of the algorithms and features, and the time frame for development.
- The cost of ongoing maintenance and updates will also need to be considered. The revenue model will depend on the chosen monetization strategy, such as in-app purchases, subscription fees, or advertising revenue.

**Conclusion:**
In conclusion, the Smart Kitchen Assistant powered by AI and ML technology provides an innovative solution for cooking enthusiasts and busy individuals who want to enjoy healthy and delicious meals at home. The project aims to simplify the cooking process and provide recipe suggestions based on user preferences and available ingredients. The final product prototype consists of a user-friendly mobile application that utilizes various data sources, algorithms, and frameworks to deliver personalized recipe suggestions and cooking guidance. The business model relies on monetization through ads, partnerships with grocery stores, and premium subscriptions. With the increasing demand for smart home technology and healthy eating, the Smart Kitchen Assistant has the potential to become a game-changer in the kitchen industry.