This project is a recipe search and generation engine with a focus on Kenyan cuisine. It allows users to search for recipes, generate new recipes based on input, view recipe details, and watch related YouTube videos. The project consists of a backend built with Node.js and Express.js, a MongoDB database for storing recipes, a custom Kenyan recipe database, integration with a recipe API, and a frontend built with React.js and styled using Tailwind CSS.

# Step-by-Step Flow

## 1. Setting Up the Environment

### Backend Setup:

- Install Node.js and Express.js.
- Set up MongoDB and connect it to the backend.
- Create a custom Kenyan recipe database.
- Integrate with a recipe API (e.g., Spoonacular or Edamam) for additional recipes.
- Set up the YouTube API for video embedding.

### Frontend Setup:

- Install React.js.
- Integrate Tailwind CSS for styling.
- Set up the basic structure for React components.

## 2. Backend Functionality

1. **Recipe Model**:
   - Define a schema for recipes in MongoDB ( `recipeModel.js` ).
   - Include fields for Kenyan-specific recipe attributes.
2. **Recipe Controller**:
   - Implement functions to handle recipe search, generation, and video retrieval ( `recipeController.js` ).
3. **Kenyan Recipe Service**:
   - Create a service to manage the custom Kenyan recipe database ( `kenyanRecipeService.js` ).

4. **Recipe API Service**:
    - Set up a service to interact with the chosen recipe API (`recipeApiService.js`).
5. **Routes**:
    - Define RESTful API endpoints for searching recipes, generating recipes, and retrieving YouTube videos (`recipeRoutes.js`).
6. **Server Setup**:
    - Create an Express.js server and connect it to MongoDB (`app.js`).

## 3. Frontend Functionality

1. **Tailwind CSS Integration**:
    - Install and configure Tailwind CSS.
    - Import Tailwind CSS into the React project.
2. **Recipe Search and Generation Component**:
    - Create a React component for recipe search and generation (`RecipeSearchAndGenerate.js`).
    - Implement search and generation functionality using Axios to call the backend API.
    - Display search results and generated recipes using Tailwind CSS for styling.

# Detailed Flow

## Backend

1. **Start the Server**:
    - Run the Express.js server (`app.js`).
    - Connect to MongoDB.
2. **Recipe Search and Generation Endpoint**:
    - When a request is made to `/api/recipes/search-or-generate`, the server first checks the Kenyan recipe database.
    - If no matching Kenyan recipes are found, it falls back to the recipe API.
    - If generation is requested, it uses the available data to create a new recipe.
    - The results are sent back to the frontend.
3. **Video Endpoint**:
    - When a request is made to `/api/recipes/video`, the server uses the YouTube API to find related videos.
    - The video data is sent back to the frontend.

## Frontend

1. **User Interaction**:
   - The user enters a dish name or ingredients in the search bar.
2. **Search or Generate Request**:
   - The frontend sends a POST request to `/api/recipes/search-or-generate` with the user's input.
3. **Display Results**:
   - The backend processes the request, searches the Kenyan database and/or recipe API, and returns the results or generated recipe.
   - The frontend displays the recipe details using Tailwind CSS for styling.
4. **Video Display**:
   - After displaying the recipe, the frontend requests a related video.
   - The frontend embeds the YouTube video below the recipe details.

# Example Flow

1. **User Searches for a Recipe**:
   - The user types "Kenyan Pilau" into the search bar and clicks the "Generate Recipe" button.
2. **Frontend Sends Request**:
   - The frontend sends a POST request to `/api/recipes/search-or-generate` with the dish name.
3. **Backend Processes Request**:
   - The backend first checks the Kenyan recipe database for "Kenyan Pilau".
   - If found, it returns the authentic Kenyan recipe.
   - If not found, it checks the recipe API or generates a new recipe based on available data.
4. **Frontend Displays Recipe**:
   - The frontend receives the recipe data and displays the ingredients and instructions.
5. **Frontend Requests Video**:
   - The frontend sends a GET request to `/api/recipes/video?q=Kenyan+Pilau+recipe`.
6. **Backend Processes Video Request**:
   - The backend uses the YouTube API to find related videos and sends the data back to the frontend.
7. **Frontend Displays Video**:
   - The frontend displays the related YouTube video embedded below the recipe details.

# Conclusion

This project allows users to search for recipes (with a focus on Kenyan cuisine), generate new recipes based on input, view detailed instructions, and watch related YouTube videos. The backend handles data storage, recipe search and generation, and video retrieval, while the frontend provides a user-friendly interface styled with Tailwind CSS.

The inclusion of a custom Kenyan recipe database ensures authentic local recipes are prioritized, while the integration with a recipe API provides a wide variety of additional recipes. This hybrid approach allows for a rich user experience that celebrates Kenyan cuisine while offering diverse recipe options.

By following the steps outlined above, you can set up and run the project to achieve the desired functionality, providing users with a unique and culturally rich recipe exploration experience.