Flutter Interview Project: AI-Powered Meal Analyzer & Planner

🎯 Project Overview

You are required to develop a Flutter mobile application that leverages Google Gemini AI to analyze meal photos and generate personalized meal plans. This project will assess your skills in Flutter development, API integration, state management, and UI/UX design.

📱 Required Features

Tab 1: Meal Photo Analysis

- *Camera Integration*: Allow users to take photos of their meals
- *AI Analysis*: Send images to Google Gemini 2.5 API for analysis
- *Meal Identification*: Display the identified meal name and description
- *Nutritional Breakdown*: Show calories, protein, carbs, fat, fiber, and other nutrients
- *Ingredients List*: Display detected ingredients in the meal
- *Save Functionality*: Store analyzed meals for future reference

Tab 2: Meal Planning Assistant

- *User Preferences*: Allow users to input dietary preferences and restrictions
- *Goal Setting*: Set daily calorie targets and macro goals
- *AI Meal Planning*: Use Gemini AI to generate complete daily meal plans
- *Customization*: Allow users to specify meal types (breakfast, lunch, dinner, snacks)
- *Nutritional Summary*: Show total daily nutrition for the generated plan

Tab 3: History & Analytics

- *Meal History*: Display previously analyzed meals
- *Nutrition Tracking*: Show daily/weekly nutrition summaries
- *Progress Visualization*: Charts or graphs showing nutrition trends

🔧 Technical Requirements

Mandatory Technologies

- *Flutter SDK*: Latest stable version
- *Google Gemini AI API*: For meal analysis and meal plan generation
- *State Management*: Choose one (Provider, Bloc, Riverpod, GetX)
- *HTTP Client*: For API communication (http, dio, or similar)
- *Camera Package*: For photo capture functionality
- *Local Storage*: For offline data persistence

API Integration Requirements

- Proper API key management (not hardcoded in source)
- Error handling for network failures
- Loading states during API calls
- Proper JSON parsing and data modeling

UI/UX Requirements

- Modern Material Design 3 implementation
- Responsive design for different screen sizes
- Smooth animations and transitions
- Bottom navigation or tab-based navigation
- Proper error and loading state UI

📋 Deliverables

1. Source Code

- Complete Flutter project with organized folder structure
- Clean, well-commented code following best practices
- Proper git version control with meaningful commits

2. APK File

- Generate and provide a release APK file
- Ensure the APK works on Android devices (API level 21+)
- Test on at least one physical device or emulator

3. Documentation

- README file with setup instructions
- API key configuration guide
- List of dependencies and their purposes
- Brief explanation of architecture decisions

4. Screenshots

Provide screenshots of:

- Camera screen with meal photo capture
- Meal analysis results with nutrition breakdown
- Meal planner interface with user inputs
- Generated daily meal plan
- History/analytics screen
- Navigation between different tabs
- Error handling scenarios

5. Demo Video (Optional but Recommended)

- 2-3 minute video demonstrating key features
- Show the complete user flow from photo capture to meal plan generation

🎯 Assessment Criteria

Technical Implementation (40 points)

- *API Integration*: Successful integration with Gemini AI (10 points)
- *State Management*: Proper implementation and data flow (10 points)
- *Code Quality*: Clean, organized, and maintainable code (10 points)
- *Error Handling*: Graceful handling of failures and edge cases (10 points)

User Interface & Experience (25 points)

- *Design Quality*: Modern, intuitive, and visually appealing UI (10 points)
- *User Flow*: Smooth navigation and logical user experience (8 points)
- *Responsiveness*: Works well on different screen sizes (7 points)

Feature Completeness (25 points)

- *Core Functionality*: All required features working correctly (15 points)
- *Data Persistence*: Proper storage and retrieval of user data (10 points)

Best Practices (10 points)

- *Project Structure*: Well-organized code architecture (5 points)
- *Documentation*: Clear setup instructions and code comments (3 points)
- *Security*: Proper handling of API keys and sensitive data (2 points)

🚀 Getting Started

Setup Steps

1. Create a new Flutter project
1. Set up Google AI Studio account and obtain Gemini API key
1. Add required dependencies to pubspec.yaml
1. Configure API key securely (using environment variables or config files)
1. Set up project folder structure
1. Implement features incrementally

Recommended Development Timeline

- *Day 1*: Project setup, API integration, basic camera functionality
- *Day 2*: Implement meal analysis feature and results display
- *Day 3*: Build meal planning interface and AI integration
- *Day 4*: Add history/analytics, implement local storage
- *Day 5*: Polish UI, testing, bug fixes, APK generation

📚 Resources & References

Flutter Packages (Suggestions)

- camera or image_picker for photo capture
- http or dio for API calls
- provider , flutter_bloc , or riverpod for state management
- sqflite or shared_preferences for local storage
- fl_chart for nutrition charts (optional)

API Documentation

- Google AI Studio: https://makersuite.google.com/
- Gemini API Documentation: https://ai.google.dev/docs

🎁 Bonus Features (Extra Credit)

- Barcode scanning for packaged foods
- Weekly meal planning
- Shopping list generation
- Social sharing of meal plans
- Dark mode implementation
- Offline functionality with sync
- Custom recipe creation and storage

📝 Submission Guidelines

What to Submit

1. *GitHub Repository*: Complete source code with proper README
1. *APK File*: Working release build
1. *Screenshots*: All required screenshots in a dedicated folder
1. *Documentation*: Setup and usage instructions

Submission Format

- Email with repository link and APK download link
- Or provide a shared drive folder with all deliverables
- Include your name and contact information

Deadline

- *Timeline*: 3 working days from assignment date

⚠️ Important Notes

- Ensure your API key is not exposed in the source code
- Test the app thoroughly before submission

- Make sure the APK installs and runs correctly
- Focus on core functionality first, then add polish
- Ask questions early if you encounter blocking issues
- Demonstrate your problem-solving skills through code comments

🏆 Evaluation Focus

We will particularly assess:

- How you structure and organize your Flutter project
- Your approach to integrating external APIs
- Quality of state management implementation
- UI/UX design decisions and implementation
- Error handling and edge case management
- Code readability and maintainability
- Problem-solving approach when facing challenges

Good luck with your implementation! This project will showcase your Flutter development skills and ability to work with modern AI APIs.