Nutri-Scan – AI-Powered Food Analyzer & Health Rating System

1. Introduction

Consumers often face challenges in understanding the nutritional content of packaged food products. Traditional review systems rely on user-generated reviews, which can be biased or fake. Many products contain hidden unhealthy ingredients such as excessive sugar, preservatives, and additives. NutriScan is an AI-driven web application that automatically analyzes ingredient lists and provides AI-generated health insights and ratings to help consumers make informed decisions.

2. Objective

- Develop an AI-powered food analysis system to provide automated, unbiased health ratings.
- Eliminate fake reviews by replacing them with AI-driven ingredient analysis.
- Detect harmful additives, preservatives, and allergens in packaged food.
- Offer personalized recommendations for healthier food alternatives.
- Provide a user-friendly web interface for seamless product search and analysis.
- Implement an Admin Module for efficient product and review management.

3. Project Category

Web Application (AI-Powered) – This project falls under AI-integrated food analysis and health rating systems.

4. Tools and Languages

✅ **Frontend:** React.js, Tailwind CSS
✅ **Backend:** Node.js, Express.js
✅ **Database:** MongoDB / Firebase
✅ **AI & APIs:** OpenAI API (GPT-based analysis), Open Food Facts API
✅ **Development Tools:** VS Code, Postman, GitHub

5. Hardware and Software Requirements

Hardware:

- Processor: Intel i5 or above
- RAM: Minimum 8GB
- Storage: Minimum 100GB HDD/SSD

Software:

- Windows/Linux/macOS

- Node.js, MongoDB/Firebase
- React.js, Express.js
- Postman for API testing

6. Modules of the Software

1. **User Interface Module** – Handles UI for product search and display.
2. **Product Fetching Module** – Retrieves food product data from Open Food Facts API.
3. **AI-Based Ingredient Analysis Module** – Generates automatic reviews and health scores.
4. **Allergen & Harmful Ingredient Detection Module** – Identifies potential allergens.
5. **Alternative Product Recommendation Module** – Suggests healthier options.
6. **User Contribution Module** – Allows users to add missing products for AI analysis.
7. **admin Module** – Provides a dashboard for managing products, monitoring AI reviews, and handling user reports.

7. Module Description

1. User Interface Module:

- Implements a clean, responsive UI using React.js and Tailwind CSS.
- Provides search functionality for users to find food products.
- Displays AI-generated ratings & reviews in an intuitive format.

2. Product Fetching Module:

- Fetches product details like ingredients, nutrition, and barcodes from Open Food Facts API.

3. AI-Based Ingredient Analysis Module:

- Uses AI (GPT-based model) to analyze food ingredient lists.
- Assigns health ratings (1-5) based on the presence of additives, preservatives, and sugars.

4. Allergen & Harmful Ingredient Detection Module:

- Flags allergens (e.g., gluten, nuts, dairy) based on user dietary preferences.
- Highlights harmful ingredients (e.g., MSG, artificial colors).

5. Alternative Product Recommendation Module:

- Suggests healthier alternatives based on nutritional value.

6. User Contribution Module:

- If a product isn’t found in the database, users can add it manually.

- AI then analyzes the new product and generates a review.

7. Easy Admin Module:

- A dedicated Admin Dashboard for managing the application.
- Admin can add, edit, and delete food products manually.
- Allows moderation of AI-generated reviews if flagged as inaccurate.
- User Report Handling: Admins can review and address reported issues (e.g., incorrect product data).
- System Monitoring: Admins can track database updates, API requests, and application logs.

8. Module Design

Diagrams:

- **Use Case Diagram:** Depicts user interactions with the system.
- **Data Flow Diagram (DFD):** Shows how data moves between modules.
- **Class Diagram:** Illustrates the object-oriented structure of the system.
- **Control Flow Diagram (CFD):** Represents the logical flow of operations in the software.

9. Database Design

- **Database:** MongoDB / Firebase
    - **Tables:**
        - **User Information**: Stores user details, including login information and dietary preferences.
        - **Product Data**: Contains product-specific information such as name, brand, ingredients, and nutritional facts.
        - **Review History**: Tracks health ratings and reviews generated by the AI for each product.

10. Limitations & Future Scope

Limitations:

- AI analysis depends on the accuracy of ingredient data from external sources.
- Some niche food products may not be available in the database.

Future Enhancements:

✅ **Mobile App Version** with barcode scanning for faster food analysis.
✅ **Expanded AI Analysis** to provide even more precise health insights.
✅ **Multi-Language Support** for diverse user groups.

11. Conclusion

NutriScan is an AI-powered food review system that eliminates fake user reviews and provides ingredient-based health analysis. By leveraging AI, databases, and real-time food insights, NutriScan helps consumers make healthier food choices effortlessly. The project is designed to be scalable, practical, and ready for deployment within 2 months.

```text
my-mern-app/
├── package.json           # Project dependencies and scripts
├── .env                   # Environment variables (MongoDB URI, API keys)
├── server.js              # Main Express.js server file
├── vite.config.js         # Vite configuration for the frontend

├── src/
│   ├── frontend/          # React frontend code (using Vite)
│   │   ├── index.html       # HTML entry point
│   │   ├── src/           # React source files
│   │   │   ├── App.jsx      # Main application component
│   │   │   ├── components/  # Reusable React components
│   │   │   │   ├── LandingPage.jsx   # Landing page component
│   │   │   │   ├── ProductCard.jsx   # Component to display a product
│   │   │   │   ├── SearchBar.jsx     # Search input component
│   │   │   │   └── ...              # Other components
│   │   │   ├── pages/         # React Page components
│   │   │   │   └── ProductDetailsPage.jsx # React page to display
product info
│   │   │   ├── services/      # API service files
│   │   │   │   └── api.js       # React API service
│   │   │   └── main.jsx         # React's entry point
│   ├── backend/           # Node.js/Express.js backend code
│   │   ├── models/          # MongoDB models
│   │   │   ├── Product.js     # Product model
│   │   │   └── User.js        # User model (for admin, etc.)
│   │   ├── controllers/     # Route handlers (business logic)
│   │   │   ├── productController.js  # Handles product-related routes
│   │   │   └── authController.js     # Handles user authentication
│   │   ├── routes/          # API endpoint definitions
│   │   │   ├── productRoutes.js    # Defines product routes (/api/products)
│   │   │   ├── authRoutes.js       # Defines authentication
routes (/api/auth)
│   │   │   └── adminRoutes.js      # Defines admin routes (/api/admin)
│   │   ├── middleware/      # Middleware functions
│   │   │   └── authMiddleware.js # Middleware for authentication
│   │   └── app.js             # Express app setup
│   ├── database/          # MongoDB connection
│   │   └── db.js              # Connects to MongoDB
│   ├── utils/             # Utility functions
│   │   └── helper.js          # General helper functions
│   ├── config/            # Configuration files
│   │   └── .env             # Backend environment variables
│   ├── ai/                # AI Module
│   │   └── openai.js        # Code related to OpenAI APIs (GPT)
│   └── openfoodfacts/     #OpenFoodFacts API
│       └── openfoodfacts.js # code for pulling data from open food facts API
├── .gitignore             # Git ignore file
└── README.md              # Project documentation
```