# PersonaLens(stack)-flutter

## Folder Structure

```
/PersonaLens
│
├── /assets                # Static files
│   ├── /images            # Profile pictures, icons, illustra
tions
│   ├── /fonts             # Custom fonts (if any)
│   ├── /styles            # Global styles (CSS/Tailwind class
es for web or app themes)
│
├── /lib                   # Main application logic (for Flutt
er or modular architecture)
│   ├── main.dart          # Entry point of the app
│   ├── /models            # Data models (Person, JournalEntr
y, Insights)
│   ├── /screens           # UI Screens
│   │   ├── HomeScreen.dart
│   │   ├── ProfileScreen.dart
│   │   ├── JournalScreen.dart
│   │   ├── InsightsScreen.dart
│   │   └── SettingsScreen.dart
│   ├── /widgets           # Reusable components
│   │   ├── ProfileCard.dart
│   │   ├── JournalEntryTile.dart
│   │   ├── EmotionTag.dart
│   │   └── GraphWidget.dart
│   ├── /services          # Services (API integrations, stora
ge handlers)
│   │   ├── AIService.dart        # AI/ML functionality (e.
g., OpenAI API)
│   │   ├── DatabaseService.dart  # Backend/Firestore handler
s
│   │   ├── EncryptionService.dart # Secure data management
│   │   └── NotificationService.dart # For reminders
│   ├── /utils             # Utility functions
│   │   ├── DateFormatter.dart
│   │   ├── ThemeConfig.dart
│   │   └── Constants.dart
│
├── /backend               # Backend services (for Node.js or
Python server)
```

```
│   ├── /api              # API routes for storing/retrieving
data
│   ├── server.js         # Main server file
│   ├── /models           # Database schemas (Person, Journal
Entry, User)
│   ├── /services         # AI/ML service modules
│   └── /database         # Connection to MongoDB/Firebase
│
├── /test                 # Unit and integration tests
│   ├── ProfileTests.dart
│   ├── JournalTests.dart
│   └── InsightsTests.dart
│
├── /docs                 # Documentation and designs
│   ├── README.md         # Overview of the app
│   ├── API_Documentation.md # Backend API details
│   └── Wireframes.pdf    # Wireframe designs
│
└── /config               # Configurations for deployment and
environment
    ├── firebase.json     # Firebase configuration (if using
Firebase)
    ├── .env              # Environment variables (API keys,
etc.)
    └── package.json      # Dependencies for backend or build
tools
```

---

## Key Modules and Responsibilities

### 1. Models

- **Person Model**: Represents each person the user logs interactions with.

  ```
  class Person {
    String id;
    String name;
    String relationship;
    String profilePictureUrl;
    List<String> tags;
    List<JournalEntry> journalEntries;
  }
  ```

- **Journal Entry Model**: Represents each logged interaction.

  ```
  class JournalEntry {
    String id;
  ```

```
  String personId;
  String content;
  DateTime date;
  List<String> emotions;
}
```

- **Insights Model**: Stores analyzed personality trends.

```
class Insights {
  String personId;
  Map<String, double> traits; // Example: {"Extroversion":
0.8, "Empathy": 0.6}
  String summary;
}
```

---

## 2. UI Screens

- **Home Screen**: Displays profiles and quick stats.
  - Quick navigation to journaling or analysis.
- **Profile Screen**:
  - Detailed view of a person, including logged interactions, tags, and insights.
- **Journal Screen**:
  - Interface for adding, editing, and viewing journal entries.
  - Emotion tagging and interaction categorization.
- **Insights Screen**:
  - Dashboard for personality analysis and relationship trends.
  - Graphical representation of traits, interaction frequency, and mood impacts.
- **Settings Screen**:
  - Configure reminders, privacy settings, and themes.

---

## 3. Core Features

### AI Service

- Sentiment Analysis: Use OpenAI API or Hugging Face for extracting sentiment.
- Personality Analysis: Generate personality insights using NLP.

## Database Service

- Use Firebase/Firestore for real-time sync, or MongoDB for NoSQL storage.
- Store user-generated data securely, ensuring modularity for easy scaling.

## Encryption Service

- Encrypt sensitive information (journal entries, insights) using AES or similar methods.

## Notification Service

- Push notifications for journaling reminders or insights updates.

---

## Tech Stack

- **Frontend**:
  - Flutter for cross-platform mobile apps.
  - React or Vue.js for web version.
- **Backend**:
  - Node.js with Express or Python Django for APIs.
- **Database**:
  - Firebase Firestore or MongoDB for dynamic and scalable storage.
- **AI/ML**:
  - OpenAI GPT for text analysis.
  - TensorFlow Lite (if offline AI is needed).

---