Here's a consolidated specification combining the \*\*Meal Maven UI design\*\* with the \*\*client's functional requirements\*\*:

---

### \*\*Core Features & Integration Plan\*\*

\*\*1. Splash Screen & Onboarding\*\*

- \*\*Lottie Animation Splash Screen\*\* (client requirement)

- \*\*Onboarding Flow\*\* (from design):

- Introduce app features (meal planning, grocery lists).

- Collect dietary preferences (vegan, gluten-free) via toggleable cards.

- Skip option for guest users.

\*\*2. Authentication\*\*

- \*\*Firebase Authentication\*\* for:

- Email/password sign-up/login.

- Social logins (Google, Facebook, Twitter).

- \*\*Guest Mode\*\*:

- Access meal browsing, search, and Meal of the Day.

- Restricted from favorites/planning (design-aligned popup to encourage sign-up).

- \*\*Session Persistence\*\*: Use `SharedPreferences` to retain login state.

\*\*3. Home Dashboard (Design-Inspired)\*\*

- \*\*Meal of the Day\*\*: Prominent card with random meal from [TheMealDB](https://themealdb.com/api.php).

- \*\*Weekly Meal Planner\*\* (client + design):

- Interactive calendar (design’s swipeable UI).

- Drag-and-drop meals into daily slots (breakfast/lunch/dinner).

- \*\*Quick Actions\*\* (design):

- "Generate Plan" (AI-based suggestions via TheMealDB filters).

- "Add Custom Meal" (manual entry).

\*\*4. Recipe Browsing & Search\*\*

- \*\*Search Filters\*\* (client requirement):

- Country, ingredient, category (use TheMealDB API endpoints).

- \*\*Recipe Catalog UI\*\* (from design):

- Grid/List view toggle.

- Cards with meal image, name, cooking time.

- \*\*Offline Access\*\*: Saved favorites/plans visible via Room DB.

\*\*5. Meal Details Screen\*\*

- \*\*Mandatory Elements\*\* (client):

- Name, image, country origin, ingredients (with icons), steps, embedded video (use `ExoPlayer`).

- "Add to Favorites" button (❤️ icon).

- \*\*Design Enhancements\*\*:

- Step-by-step carousel with images (design’s visual style).

- Nutritional info (if available via API).

\*\*6. Favorites & Meal Planning\*\*

- \*\*Local Storage\*\*: Room DB for favorites and weekly plans (client requirement).

- \*\*Sync/Backup\*\*: Firebase Firestore to sync data across devices (post-login).

- \*\*Offline Mode\*\*: Display cached favorites/plans if no network.

\*\*7. Grocery List (Design Bonus)\*\*

- Auto-generate from meal plans (design feature).

- Manual editing (client’s "add/remove items").

- Check-off items with swipe gestures.

\*\*8. Profile & Settings\*\*

- \*\*Dietary Preferences\*\*: Update allergies/diets (from onboarding).

- \*\*Sync Status\*\*: Indicator for Firebase backup.

- \*\*Dark Mode\*\*: Toggle (design’s theme customization).

---

### \*\*Technical Stack\*\*

1. \*\*API Integration\*\*:

- TheMealDB for recipes, categories, countries, and Meal of the Day.

- YouTube Data API for parsing/embedding video URLs (client’s video requirement).

2. \*\*Local Database\*\*:

- \*\*Room DB\*\*: Favorites, meal plans, grocery lists.

3. \*\*Authentication & Sync\*\*:

- \*\*Firebase Auth\*\*: Social/email logins.

- \*\*Firestore\*\*: Backup user data (favorites/plans).

4. \*\*UI/UX\*\*:

- \*\*Lottie\*\*: Animated splash screen.

- \*\*Material Design 3\*\*: Components (cards, bottom nav, FAB).

- \*\*Navigation Component\*\*: For fragments (Home, Search, Profile).

5. \*\*Bonus Feature\*\*:

- \*\*Device Calendar Integration\*\*: Use `CalendarContract` to add meals to the local calendar.

---

### \*\*Prioritized Roadmap\*\*

1. \*\*Phase 1 (MVP)\*\*:

- Splash screen + onboarding.

- Meal browsing/search (TheMealDB integration).

- Favorites/planning with Room.

- Firebase auth + guest mode.

2. \*\*Phase 2 (Enhancements)\*\*:

- Grocery list auto-generation.

- Sync/backup with Firestore.

- Embedded video player.

3. \*\*Phase 3 (Bonus)\*\*:

- Calendar integration.

- Dark mode + advanced animations.

---

### \*\*Design-Client Alignment\*\*

| \*\*Client Requirement\*\* | \*\*Design Implementation\*\* |

|-------------------------------|---------------------------------------------------|

| Meal of the Day | Featured card on home dashboard. |

| Weekly planning | Interactive calendar with drag-and-drop meals. |

| Firebase sync | Profile screen sync status indicator. |

| Guest mode restrictions | Disabled CTA buttons (e.g., "Add to Plan"). |

| Recipe video embedding | Full-screen video player with thumbnail preview. |

This hybrid approach merges the client’s functional needs with the modern UI/UX from the Meal Maven design, ensuring both technical feasibility and aesthetic appeal.

Time Line:

**3-Week Development Plan**

**Week 1: Core Setup & Authentication**  
**Goal**: Lay the foundation and implement user auth.

1. **Day 1-2**:
   * Set up Android Studio project with the above file structure.
   * Configure Gradle dependencies (Retrofit, Room, Firebase, Hilt, Lottie).
   * Create SplashActivity with Lottie animation.
2. **Day 3-4**:
   * Implement Firebase Authentication:
     + Email/password login/signup.
     + Google sign-in (using FirebaseAuth).
   * Create LoginFragment and SignupFragment UI.
3. **Day 5-7**:
   * Set up Room Database:
     + Define FavoriteEntity, MealPlanEntity, and DAOs.
     + Create AppDatabase and test CRUD operations.
   * Add SharedPreferencesHelper for session persistence.

**Week 2: Core Features (Meal Browsing, Planning, Favorites)**  
**Goal**: Enable meal discovery and planning.

1. **Day 1-2**:
   * Integrate TheMealDB API:
     + Create MealApiService for endpoints (search.php, categories.php, etc.).
     + Fetch "Meal of the Day" and display on HomeFragment.
2. **Day 3-4**:
   * Build SearchFragment:
     + Filters for country, category, ingredient.
     + RecyclerView to display results.
   * Add click listeners to open MealDetailsFragment.
3. **Day 5-7**:
   * Implement MealDetailsFragment:
     + Show ingredients, steps, and embedded video (ExoPlayer).
     + Add "Favorite" button (Room DB integration).
   * Create weekly planner UI (MealPlanFragment) with drag-and-drop.

**Week 3: Sync, Grocery List, and Polish**  
**Goal**: Finalize features and add polish.

1. **Day 1-2**:
   * Firebase Firestore Sync:
     + Backup favorites/meal plans upon login.
     + Restore data when user logs in again.
2. **Day 3-4**:
   * Build GroceryFragment:
     + Auto-generate list from meal plans.
     + Manual editing (add/remove items).
   * Add swipe-to-delete and check-off functionality.
3. **Day 5-7**:
   * **Bonus**:
     + Calendar integration using CalendarContract.
     + Dark mode toggle in ProfileFragment.
   * **Testing**:
     + Write unit tests for ViewModels.
     + Test offline scenarios (favorites/plans without network).
   * **Polish**:
     + Add animations (fragment transitions, item clicks).
     + Handle edge cases (empty states, API errors).

**Final Checklist**

1. **Version Control**: Initialize Git and commit daily.
2. **API Keys**: Secure TheMealDB/Firebase keys in local.properties or use BuildConfig.
3. **Proguard Rules**: Add rules for Retrofit, Room, and Firebase.
4. **Play Store Prep**: Generate signed APK and prepare store listing.

This plan balances speed and quality. Adjust based on progress, and prioritize testing early!