Week 1: Authentication & Setup

Focus: Set up Firebase Authentication and initial API structure.

- **Day 1: Project Setup**
    - Initialize a **Node.js/Express** project.
    - Set up Firebase in your project (install SDK and initialize the app).
    - Create necessary config files for Firebase and Node.js environment.
- **Day 2-3: Firebase Authentication**
    - Implement Firebase **email/password** authentication.
    - Implement **Google login** using Firebase.
    - Create API endpoints for **signup/login** and user authentication status (e.g., /auth/signup, /auth/login, /auth/status).
- **Day 4: Token Validation**
    - Implement token-based authentication (use Firebase **ID tokens** for verifying users).
    - Create middleware to protect routes by checking user authentication status.
- **Day 5: User Management**
    - Create API endpoints to manage user data (e.g., /users).
    - Store basic user information in Firebase (email, name, user type: client/provider).
    - Document the API using **Postman** (provide frontend dev with Postman collection).

Deliverables:

- Firebase Authentication (Email, Google).
- Token-based authentication middleware.
- Basic user API (e.g., user registration, login).
- Postman documentation for authentication routes.

Week 2: Booking System API

Focus: Develop APIs for the booking system, allowing clients to book photography/editing services.

- **Day 1-2: Database Structure & API Design**
    - Design Firebase database structure (Firestore) for:
        - User profiles (clients and service providers).

        - Projects (services offered by providers).
        - Bookings (project bookings by clients).
    - Define and document API endpoints for booking management (e.g., /projects, /bookings).
- **Day 3-4: Project Management API**
    - Implement API endpoints for service providers to create, update, and delete projects (e.g., /projects/create, /projects/update).
    - Create the ability to list available services for clients to browse.
- **Day 5: Booking Management API**
    - Implement API for clients to book services (e.g., /bookings/create).
    - Develop endpoints to view and manage bookings (e.g., /bookings/user/:id for clients and providers).

Deliverables:

- API for creating and managing services (projects).
- API for booking services.
- Firestore structure for storing user data, projects, and bookings.
- Postman documentation for booking and project APIs.

Week 3: Order Tracking & Notifications

Focus: Implement real-time order tracking and notifications for clients to monitor progress.

- **Day 1-2: Order Status Tracking API**
    - Define order states (e.g., Pending, In Progress, Delivered).
    - Implement API to update and track the status of a booking (e.g., /bookings/:id/status).
    - Ensure the client can query the status of their booking in real-time using Firebase Firestore listeners or polling.
- **Day 3-4: Real-Time Firebase Integration**
    - Use **Firebase Realtime Database** or **Firestore triggers** for real-time updates on the client side.
    - Send real-time updates (e.g., order status changes) to the client when the service provider updates a booking.
- **Day 5: Notifications Setup**

- Integrate Firebase Cloud Messaging (FCM) to send notifications to users about order status updates (optional based on time).
- Implement API endpoint for sending notifications from the backend.

Deliverables:

- Real-time order tracking system.
- API for updating order status.
- (Optional) Notifications setup with Firebase Cloud Messaging.
- Postman documentation for tracking APIs.

Week 4: Testing, Optimization & Deployment

Focus: Finalize the backend, test integration with the frontend, and deploy.

- **Day 1-2: Testing**
  - Test all API endpoints with **Postman** (authentication, booking, tracking).
  - Coordinate with the frontend developer to test the integration of key flows (e.g., authentication, booking creation, real-time tracking).
- **Day 3: Bug Fixes & Optimization**
  - Fix any issues found during testing (API bugs, database issues).
  - Optimize database structure and query efficiency (e.g., avoid redundant reads/writes).
- **Day 4: Deployment Preparation**
  - Set up Heroku for backend deployment.
  - Ensure environment variables (Firebase credentials, API keys) are correctly set on Heroku.
  - Deploy the backend to Heroku and test live endpoints.
- **Day 5: Final Sync & Handover**
  - Perform a final sync-up with the frontend developer.
  - Ensure that the frontend has access to all necessary backend documentation (Postman collection, API keys).

Deliverables:

- Fully tested backend with authentication, booking, and tracking features.
- Backend deployed on Heroku.

- API documentation updated for frontend integration.

Key Sync Points with Frontend Developer:

1. **End of Week 1**: Hand over **authentication endpoints** (signup, login, token validation).
2. **End of Week 2**: Share **booking API** for frontend integration.
3. **End of Week 3**: Ensure frontend has **real-time tracking endpoints** for order status updates.