# **Lost & Found Item Tracker**

# **Feature Roadmap & Use Cases**

### **🔑 1. User Authentication System**

🎯 *Allow users to register, log in, and secure their sessions.*

* 👥 Actors: Guest, Authenticated User
* 📜 Use Cases:
  + Register with email, username, password
  + Login to receive access token
  + Logout or revoke token
  + View and update profile (name, phone, location)
* 🔗 Dependencies: None (foundation)
* ⏱️ Time Estimate: 2–3 days
* ⚙️ JWT via SimpleJWT or token-based auth with djoser

### **📤 2. Posting a Lost or Found Item**

🎯 *Users can submit a report for a lost or found item.*

* 👥 Actors: Authenticated user (optional anonymous for "found")
* 📜 Use Cases:
  + Submit item with: name, category, description, image, location, date
  + Mark as “lost” or “found”
  + Upload photo
  + Edit or delete own item
* 🔗 Dependencies: Auth
* ⏱️ Time Estimate: 3–4 days
* ⚙️ Use ImageField with proper validation & upload handling

### **🔍 3. Public Item Browsing & Filtering**

🎯 *Anyone can browse/search lost/found items.*

* 👥 Actors: Guest, Authenticated User
* 📜 Use Cases:
  + Search by item name, category, location, or type (lost/found)
  + Filter by date range
  + Sort by newest first or location proximity (basic)
* 🔗 Dependencies: Item CRUD
* ⏱️ Time Estimate: 2 days
* ⚙️ Use DRF filters + django-filter

### **🤝 4. Matching System (Backend Logic)**

🎯 *System suggests matches between lost and found items.*

* 👥 Actors: Authenticated user
* 📜 Use Cases:
  + If a user posts a lost item, system checks for found items with:
    - Same category
    - Keyword match (in name/description)
    - Date within ±3 days
    - Optional: location similarity
  + Return top 3–5 suggestions
* 🔗 Dependencies: Item Model
* ⏱️ Time Estimate: 2 days
* ⚙️ Can use TrigramSimilarity, or simple fuzzy match on text

### **📨 5. Contact / Claim Process**

🎯 *Enable secure communication between poster and interested user.*

* 👥 Actors: Authenticated user
* 📜 Use Cases:
  + If match is found, user can request to contact poster
  + Poster accepts or declines
  + If accepted, contact info is shared (email/phone)
  + System logs the claim attempt
* 🔗 Dependencies: Match System, Auth
* ⏱️ Time Estimate: 2–3 days
* ⚙️ Optional future upgrade: chat system

### **🧾 6. Mark as Resolved**

🎯 *Users can mark items as resolved (returned or recovered).*

* 👥 Actors: Authenticated user
* 📜 Use Cases:
  + Mark item as "Resolved"
  + Remove it from search results
* 🔗 Dependencies: Item CRUD
* ⏱️ Time Estimate: 1 day

### **🛡️ 7. Admin Panel / Moderation**

🎯 *Admin can flag, remove, or monitor reports.*

* 👥 Actors: Admin
* 📜 Use Cases:
  + View all items
  + Flag/report abuse
  + Delete spam/inappropriate content
  + View resolution rate, popular categories
* 🔗 Dependencies: All prior systems
* ⏱️ Time Estimate: 2 days
* ⚙️ Use Django Admin for Phase 1

### **📚 8. API Documentation**

🎯 *Make the API easy to consume by frontend/mobile.*

* 📜 Use Cases:
  + Generate Swagger/OpenAPI documentation
  + Ensure all endpoints are described
* 🔗 Dependencies: All functional APIs
* ⏱️ Time Estimate: 1 day
* ⚙️ Use drf-spectacular or drf-yasg

### **📦 Optional Phase 2 (Post-MVP)**

These can be Gantt-planned separately later:

* 📍 Geo-based radius search (GeoDjango or custom bounding box)
* 🔔 Email notifications on matched items
* 📎 File uploads with S3
* 📱 React/Flutter frontend