**Software Requirements Specification (SRS) Document**

For: University Lost and Found Mobile Application

**Content**

**Chapter one**

**1.Introduction**

1.1 Purpose**......................................................................................................................2**

1.2 Scope**…….....................................................................................................................2**

1.3 Definitions**...................................................................................................................3**

1.4 References**...................................................................................................................3**

**2. Overall Description**

2.1 User Needs**..................................................................................................................4**

2.2 System Features (High-Level) **.....................................................................................4**

2.3 Operating Environment**...............................................................................................5**

2.4 Constraints**....................................................................................................................6**

**3. Specific Requirements**

**3.1 Functional Requirements.............................................................................................6**

3.1.1 User Authentication.**...................................................................................................6**

3.1.2 Post Management**.......................................................................................................7**

3.1.3 Search and Discovery**...................................................................................................7**

3.1.4 Contact Mechanism**......................................................................................................8**

3.1.5 Profile Management**......................................................................................................8**

**3.2 Non-Functional Requirements........................................................................................8**

3.2.1 Performance**...................................................................................................................8**

3.2.2 Security**...........................................................................................................................8**

3.2.3 Usability**..........................................................................................................................9**

3.2.4 Reliability**........................................................................................................................9**

**3.3 External Interfaces**

3.3.1 User Interfaces**.................................................................................................................9**

3.3.2 Hardware Interfaces**.........................................................................................................9**

3.3.3 Communication Interfaces**...............................................................................................10**

**3.4 Data Management...............................................................................................................10**

3.4.1 Data Storage**......................................................................................................................10**

3.4.2 Data Retention**..................................................................................................................10**

**4. Data Management**

4.1 Data Storage**.........................................................................................................................10**

4.2 Data Retention**......................................................................................................................11**

4.4 Data Deletion and Compliance**.............................................................................................11**

1. **Introduction**

**1.1 Purpose**

This Software Requirements Specification (SRS) document outlines the functional and non-functional requirements for the development of a University Lost and Found Mobile Application. The primary purpose of this application is to provide a self-managed platform for university students, faculty, and staff to report lost items and search for found belongings within the campus community. Unlike traditional lost-and-found systems, the app eliminates intermediaries by enabling direct communication between users while maintaining privacy. Key goals include streamlining the process of posting and discovering lost items, ensuring secure user authentication, and fostering a collaborative environment for reuniting owners with their belongings. By excluding administrative oversight and in-app chat features, the app prioritizes simplicity, user autonomy, and efficiency, ensuring that users retain full control over their posts and interactions.

**1.2 Scope**

The application’s scope is strictly defined to address the core need of connecting individuals who have lost or found items on campus. In-scope functionalities include:

**User registration and authentication**: Users must create accounts to post or interact with listings, ensuring accountability.

**Post creation and management**: Users can submit detailed listings for lost or found items, including descriptions, categories (e.g., electronics, ID cards), locations, dates, and contact information.

**Search and filtering**: Users can search posts using keywords, categories, or date/location filters to locate relevant items.

**Direct contact**: A secure mechanism allows users to contact the owner/finder via email or phone without revealing personal details publicly.

**Profile management**: Users can update their profiles, view their post history, and mark items as resolved.

Out-of-scope features include administrative dashboards for user or post moderation, in-app messaging systems, reward mechanisms, or integration with external university databases. The app relies solely on user-generated content and self-management, with no oversight beyond basic data validation during registration.

**1.3 Definitions**

**Lost Item**: A physical object reported as missing by a user, including details such as description, location, and contact information.

**Found Item**: A physical object reported as discovered by a user, with attributes similar to a lost item post.

**User Profile**: A personalized account containing a user’s credentials, contact details, and history of active/resolved posts.

**Post Status**: Indicates whether a post is Active (item still missing/found) or Resolved (item returned/claimed).

**OAuth 2.0**: A protocol for secure user authentication.

**1.4 References**

**OAuth 2.0**: Used for secure user login and authentication.

**Firebase**: A backend-as-a-service (BaaS) platform for user authentication, database management, and cloud storage (e.g., for post images).

**University IT Policies**: Any institutional guidelines for data handling or email domain validation (e.g., restricting registration to @university.edu emails).

**2. Overall Description**

**2.1 User Needs**

The application is designed to address the specific needs of a university community, where students, faculty, and staff frequently misplace or discover items in shared spaces like classrooms, libraries, or cafeterias. Primary users fall into two categories:

**Individuals who lost an item**: They require a quick, intuitive way to report their loss, including details like item description, location, and contact information. Privacy is critical, as they want to share contact details only with legitimate finders.

**Individuals who found an item**: They need a centralized platform to search for matching lost posts or create new found-item listings. They prioritize efficient search tools (e.g., filters for item type, date, or location) to identify the owner swiftly.

Secondary needs include the ability to edit or delete posts (e.g., if an item is recovered), view post history, and maintain a secure profile. Users expect minimal friction in navigation and a design that aligns with common mobile app conventions.

**2.2 System Features (High-Level)**

The application’s core features are structured to align with user workflows:

**User Authentication**:

Users create accounts using university-affiliated emails (e.g., @university.edu) to ensure the community remains restricted to verified members. Passwords are securely encrypted, and a recovery option is provided via email.

**Lost/Found Posting**:

Users can create posts with mandatory fields (title, category, description) and optional fields (images, exact location, date). Posts are tagged as “Lost” or “Found” and include contact details (email/phone) hidden behind a “Contact” button to prevent public exposure.

**Search and Filters**:

A dynamic search bar allows keyword queries (e.g., “black backpack”), while filters refine results by category (e.g., electronics, clothing), date range, or campus location (e.g., “Library” or “Building A”).

**Direct Contact**:

Clicking the “Contact Owner” button on a post opens the user’s default email or phone app with pre-filled recipient details, streamlining communication without storing messages within the app.

**Profile Management**:

Users update personal information (e.g., phone number) and view their active/resolved posts. Resolved posts are archived but accessible for reference.

**2.3 Operating Environment**

The app is designed for iOS and Android mobile platforms to ensure broad accessibility. Key technical components include:

**Frontend**: Built with cross-platform frameworks ( Flutter) for consistency across devices.

**Backend**: A cloud-based service (Firebase) handles user authentication, database storage (posts, profiles), and image hosting.

**Dependencies**:

Integration with device features (camera for uploading item photos, GPS for optional location tagging).

Validation of university email domains during registration to restrict access to campus members.

**2.4 Constraints**

The system operates under the following limitations:

**No Admin Features**:

Posts and user accounts are entirely self-managed. There is no administrative dashboard to moderate content, resolve disputes, or delete inappropriate posts, relying instead on user reporting and community accountability.

**No In-App Chat**:

Communication occurs externally via email or phone, which simplifies development but limits traceability of interactions.

**University Affiliation Requirement**:

Registration is restricted to users with verified university email addresses, excluding visitors or external parties.

**Post Expiry:**

To prevent clutter, unresolved posts are automatically archived after 30 days, requiring users to repost if the item remains missing.

**3. Specific Requirements**

**3.1 Functional Requirements**

**3.1.1 User Authentication**

**FR1: User Registration**

Users can create an account using a valid university email address (e.g., @university.edu). The registration process includes:

**Mandatory fields**: Full name, university email, password (minimum 8 characters with at least one number and special character).

Email verification via a one-time link sent to the user’s inbox.

Domain validation to restrict registration to university-affiliated users.

**FR2: Secure Login**

Users log in with their registered email and password. Credentials are encrypted using hashing. Sessions expire after 30 minutes of inactivity.

**3.1.2 Post Management**

**FR4: Post Creation**

**Users create posts with**:

**Mandatory fields**: Post type (Lost/Found), title (max 50 characters), category (e.g., Electronics, Books), description (max 500 characters), date of loss/finding.

**Optional fields**: Upload up to 3 images (JPEG/PNG, max 2MB each), location (selectable campus landmarks or free-text entry).

**Contact details**: User’s email/phone number, hidden behind a “Contact” button visible only to logged-in users.

**FR5: Post Editing/Deletion**

Users can edit or delete their own posts at any time. Edits are logged with timestamps.

**FR6: Post Expiry**

Posts are automatically marked as “Expired” after 30 days unless manually resolved. Users receive a notification 3 days before expiry.

**3.1.3 Search and Discovery**

**FR7: Keyword Search**

Users search posts using keywords (e.g., “black calculator”). Results prioritize exact matches, followed by partial matches.

**FR8: Advanced Filters**

Filters include:

Category (e.g., Clothing, IDs).

**Date range** (last 7/30 days or custom).

**Location** (e.g., “Science Building” or “Cafeteria”).

**3.1.4 Contact Mechanism**

**FR9: Secure Contact**

Clicking the “Contact Owner” button reveals the poster’s email/phone number. For email, the user’s default email app opens with a pre-filled subject line (e.g., “Found Your [Item Name]”).

**FR10: Privacy Guard**

Contact details are never displayed publicly. Access requires active login to deter spam.

**3.1.5 Profile Management**

**FR11: Profile Updates**

Users can edit their name, contact details, and password. Changes to email require re-verification.

**FR12: Post History**

Users view a dashboard of their active, resolved, and expired posts. Resolved posts are archived but searchable.

**3.2 Non-Functional Requirements**

**3.2.1 Performance**

**NFR1: Response Time**

Search results load within 2 seconds under normal load (up to 1,000 concurrent users).

**NFR2: Scalability**

The backend supports up to 10,000 posts and 5,000 registered users without degradation.

**3.2.2 Security**

**NFR3: Data Encryption**

All user data (passwords, contact details) are encrypted and in transit .

**NFR4: Input Sanitization**

User inputs (e.g., post descriptions) are sanitized to prevent SQL injection and XSS attacks.

**3.2.3 Usability**

**NFR5: Onboarding**

First-time users complete registration and create a post in under 5 minutes. Tooltips guide critical actions.

**NFR6: Accessibility**

UI complies with WCAG 2.1 Level AA (e.g., alt text for images, screen reader compatibility).

**3.2.4 Reliability**

**NFR7: Uptime**

99% uptime during academic terms, excluding scheduled maintenance (announced via in-app notifications).

**NFR8: Error Handling**

Clear error messages for common issues (e.g., “Invalid email format”) with recovery suggestions.

**3.3 External Interfaces**

**3.3.1 User Interfaces**

**UI1**: Login/Registration Screen

Clean layout with fields for email, password, and a “Forgot Password?” link.

**UI2**: Post Creation Screen

Form with collapsible sections for mandatory/optional fields and image upload previews.

**UI3**: Search Screen

Search bar at the top, followed by filter chips (category, date, location) and scrollable results.

**3.3.2 Hardware Interfaces**

Camera/Gallery access for uploading item photos.

GPS (optional) for auto-tagging location in posts.

**3.3.3 Communication Interfaces**

Integration with device-native email/SMS apps (e.g., Gmail, Messages).

**3.4 Data Management**

**3.4.1 Data Storage**

**User Data**: Stored in a NoSQL database (Firestore) with fields: UserID, Name, Email, HashedPassword, Phone, Posts.

**Post Data**: Structured as documents with fields: PostID, UserID, Type, Title, Description, Images, Status, Expiry Date.

**3.4.2 Data Retention**

Active posts retained for 30 days; expired posts moved to cold storage (archival) for 6 months.

User accounts persist unless deleted. Deletion triggers anonymization of posts (contact details removed).

**Key Considerations**

**Privacy**: Contact details are never exposed publicly; access requires authentication.

**Scalability**: Cloud infrastructure (Firebase) allows horizontal scaling during peak usage (e.g., start/end of semesters).

**Compliance**: GDPR principles enforced (e.g., right to delete data, explicit consent for email usage).

**4. Data Management**

**4.1 Data Storage**

The application will store two primary types of data: user profiles and lost/found posts.

**User Profiles**:

Stored in a NoSQL database (e.g., Firebase Firestore) with fields such as UserID, Name, HashedPassword, Email, Phone, and RegistrationDate.

Sensitive data (e.g., passwords, contact details) are encrypted using encryption at rest.

Users’ email addresses are validated against a university domain (e.g., @university.edu) during registration.

**Lost/Found Posts**:

Each post includes fields like PostID, UserID, Type (Lost/Found), Title, Description, Category, Location, Date, Status (Active/Resolved/Expired), and ExpiryDate.

Images uploaded with posts are stored in cloud storage (e.g., Firebase Storage) with optimized compression to reduce load times.

Posts are linked to user accounts via UserID to enforce ownership and editing permissions.

**4.2 Data Retention**

**Active Posts**:

Remain visible for 30 days unless marked as “Resolved” by the user.

Users receive a push notification 3 days before expiry, prompting them to renew or resolve the post.

**Expired/Resolved Posts**:

Archived in a separate database collection for 6 months for audit purposes but excluded from search results.

After 6 months, posts are permanently deleted, including associated images.

**User Accounts**:

Persist indefinitely unless deleted by the user.

Account deletion triggers anonymization of all associated posts (user contact details removed, posts retained as “Anonymous”).

**4.4 Data Deletion and Compliance**

**User-Initiated Deletion**:

Users can delete their accounts via the profile settings, which triggers:

Removal of personal data (email, phone number) from the database.

Anonymization of posts (retains item descriptions but removes user identifiers).