
Software Requirements Specification

for

Lostify

Version 1.0

Prepared by

Group #6

Group Name: hAPPy Developers

Aayush Kumar	230027	aayushk23@iitk.ac.in
Aman Raj	230116	amanraj23@iitk.ac.in
Anirudh Cheriyanaseri Bijay	230140	anirudhcb23@iitk.ac.in
Ayush Patel	220269	ayushpatel22@iitk.ac.in
Krishna Kumayu	230576	kkrishna23@iitk.ac.in
Marada Teja Satvik	230636	maradateja23@iitk.ac.in
Satwik Raj Wadhwa	230937	satwikraj23@iitk.ac.in
Shaurya Johari	230959	shauryaj23@iitk.ac.in
Somaiya Narayan Aniruddh	231019	snarayana23@iitk.ac.in
Vinay Chavan	231155	vinay23@iitk.ac.in

Course: CS253

Mentor TA: Jeswaanath Gogula

Date: 23 January 2025

Contents

CONTENTS	II
REVISIONS	III
1 INTRODUCTION	1
1.1 PRODUCT SCOPE	1
1.2 INTENDED AUDIENCE AND DOCUMENT OVERVIEW	1
1.3 DEFINITIONS, ACRONYMS AND ABBREVIATIONS	1
1.4 DOCUMENT CONVENTIONS	2
1.5 REFERENCES AND ACKNOWLEDGMENTS	2
2 OVERALL DESCRIPTION	3
2.1 PRODUCT OVERVIEW	3
2.2 PRODUCT FUNCTIONALITY	4
2.3 DESIGN AND IMPLEMENTATION CONSTRAINTS	4
2.4 ASSUMPTIONS AND DEPENDENCIES	6
3 SPECIFIC REQUIREMENTS	7
3.1 EXTERNAL INTERFACE REQUIREMENTS	7
3.2 FUNCTIONAL REQUIREMENTS	12
3.3 USE CASE MODEL	15
4 OTHER NON-FUNCTIONAL REQUIREMENTS	25
4.1 PERFORMANCE REQUIREMENTS	25
4.2 SAFETY AND SECURITY REQUIREMENTS	25
4.3 SOFTWARE QUALITY ATTRIBUTES	25
APPENDIX A – DATA DICTIONARY	29
APPENDIX B – GROUP LOG	30

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
1.0	Aayush Kumar Anirudh Cheriyachanaseri Bijay Krishna Kumayu Satwik Raj Wadhwa	Preliminary draft	23/01/25

1 Introduction

1.1 Product Scope

The Lostify app for IIT Kanpur provides a platform to help the IIT Kanpur community locate and recover lost items efficiently. The app acts as a centralized hub for reporting lost and found items, enabling users to easily submit and track lost belongings. The app also features real-time notifications, search functionalities, and a secure verification system to ensure safe and accurate item retrieval. This solution aims to streamline the lost-and-found process, providing a hassle-free experience for the IIT Kanpur community.

1.2 Intended Audience and Document Overview

This document is primarily intended for developers, documentation authors, stakeholders, project managers, testers, users, and approvers. The intent and purpose of this document for each party may vary significantly.

This is an evolving document and as such is subject to change.

In its initial form, it is incomplete by definition and shall undergo continuous refinement. Requirements presently listed may be modified and additional requirements may be introduced as development progresses and the product description gains further clarity. This information will serve as a framework for the current definition and future evolution of the project.

1.3 Definitions, Acronyms and Abbreviations

Admin	A registered user of the app with special privileges
API	Application programming interface
FAQs	Frequently asked questions
HTTPS	Hypertext Transfer Protocol Secure
IEEE	Institute of Electrical and Electronics Engineers
IIT Kanpur IITK Institute	Indian Institute of Technology, Kanpur
IP address	Internet Protocol address
JSON	JavaScript Object Notation
OTP	One-time password
Registered user	A user of the app who has signed up using his Institute credentials

SDK	Software development kit
SQL	Structured Query Language
UI	User interface
UUID	Universally Unique Identifier
UX	User experience
XML	Extensible Markup Language

1.4 Document Conventions

These software requirements specifications were prepared following the software and systems engineering standards of the Institute of Electrical and Electronics Engineers (IEEE) cited below.

1. Text in this document is laid out in Arial with body text having font size 11 pt.
2. The document maintains a 1" margin and is single-spaced throughout.
3. Section titles use font size 18 pt. and are shaded. Top-level headings use font size 14 pt., while second-level headings use 12 pt.
4. Unordered list items are prefixed with circular bullets.
5. Comments are italicised and keywords are in bold.

1.5 References and Acknowledgments

Institute of Electrical and Electronics Engineers (IEEE), Inc. (1998, October 20). IEEE Recommended Practice for Software Requirements Specifications. *IEEE Std 830-1998*, 1-40. New York, NY, USA: IEEE, Inc. [doi:10.1109/IEEESTD.1998.88286](https://doi.org/10.1109/IEEESTD.1998.88286)

2 Overall Description

2.1 Product Overview

The Lostify application is a self-contained product designed to streamline the process of reporting and reclaiming lost items within a localised community environment, such as IITK campus. The application aims to replace traditional, inefficient methods such as WhatsApp groups, notice boards, word-of-mouth, or standalone spreadsheets with a unified digital platform that ensures better tracking, reporting, and claiming of lost and found items.

This system is built to provide a seamless and user-friendly interface, ensuring accessibility for all users. Key features include user authentication, intuitive item search, detailed item reporting, and a robust notification system. The application serves as a bridge between individuals who have lost items and those who have found them, facilitating quick and efficient communication and resolution.



2.1.1 Product Perspective

The Lostify application operates as a centralized platform connecting individuals within the community who have lost or found items. The system's main objective is to minimize the time and effort required to reunite lost items with their rightful owners, leveraging features such as:

- **User authentication:** Secure login system using credentials or single sign-on to verify users and maintain the integrity of the platform.
- **Lost item reporting:** Users can submit detailed reports of their lost items, including descriptions, photos, and the location/time where the item was last seen.
- **Found item reporting:** Similarly, users can report items they have found, with options to upload photos and provide details about where the item was discovered.
- **Search and filter options:** An advanced search bar allows users to filter items based on criteria such as category, date, location, or keywords, making it easier to locate relevant listings.
- **Notifications:** Real-time updates keep users informed about status changes or new listings.
- **Claim verification:** A streamlined process for users to claim items, involving communication between the finder and the owner, supported by verification steps. This shall involve submitting the found item to the nearest security official and informing the location of said official.

This product is ideal for environments with a high concentration of lost-and-found incidents, such as college campuses, office buildings, or residential complexes.

2.2 Product Functionality

This product is designed to simplify the process of finding and reclaiming lost items while fostering a supportive and interactive community. It includes the following features:

- Profile (sign up and log in)
- Item posting and categorisation
- Search and filter
- Interactive chat
- Customised notifications
- Verification and claim system
- Lost item recovery assistance
- Moderation and reporting
- Dark mode and theme customisation

2.3 Design and Implementation Constraints

The design and implementation of the lost-and-found Android application are subject to the following constraints to ensure functionality, security, and usability.

2.3.1 Database Storage

All user profiles, item details, and activity logs must be securely stored in a robust, cloud-based database accessible by the Android app. The database should ensure data consistency, integrity, and quick retrieval.

2.3.2 User Authentication

Users must log in using their correct usernames and passwords to access their accounts and perform actions like posting or claiming items. Secure authentication protocols (e.g., OAuth2 or token-based systems) must be implemented to protect user credentials.

2.3.3 Permissions Control

Different categories of users need different levels of permission. For instance, a security official may need different permissions than an admin, both of whom need greater permissions than a standard user (such as a regular student or faculty member).

2.3.4 Performance and Response Time

The app's response time for loading screens, posting items, or retrieving results should not exceed 2 minutes under typical network conditions, ensuring a smooth and efficient user experience.

2.3.5 Platform Accessibility

The application will be developed for Android devices and optimized for various screen sizes and resolutions. Users can access the app from any Android device with Internet browsing capabilities and an active Internet connection.

2.3.6 Skill Requirements

The app will feature an intuitive and user-friendly interface, requiring only basic knowledge of Android navigation to operate. Users do not need technical expertise to utilise the platform.

2.3.7 Offline Functionality

Basic features such as viewing recently posted items or drafts of posts should be accessible offline. Synchronisation with the database will occur automatically when the device reconnects to the Internet.

2.3.8 Battery and Resource Optimization

The app must be designed to minimize resource consumption, ensuring it operates efficiently without draining device battery or overloading system memory.

2.3.9 Error Handling and Recovery

The app should provide clear and informative error messages for issues like login failures, server unavailability, or incorrect user input. Automatic recovery mechanisms should handle minor disruptions (e.g., temporary network loss).

2.4 Assumptions and Dependencies

Most functionalities of the application require a good Internet connection. It is assumed that users have access to good connection facilities.

Smooth running of the application requires that the backend downtime be minimal and that the server be capable of handling the expected traffic.

An initial database of users to assign admin privileges to shall exist. They shall also have direct access to the user database and can modify it to set permissions for other users.

3 Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

The user interface (UI) for the application is designed to provide an intuitive and interactive experience for users. Below are the logical characteristics of each interface and a description of how users will interact with the application.

3.1.1.1 Login and Registration Page

Sign in

Email address

Password

[Forgot password?](#)

Login

[Don't have an account?](#)

Sign up

Logical Characteristics:

- Users can log in using their credentials or register for a new account.
- The interface features input fields for username, password, and optional two-factor authentication.

User Interaction:

- Users input their IITK credentials and tap the Login button.

3.1.1.2 Home Dashboard

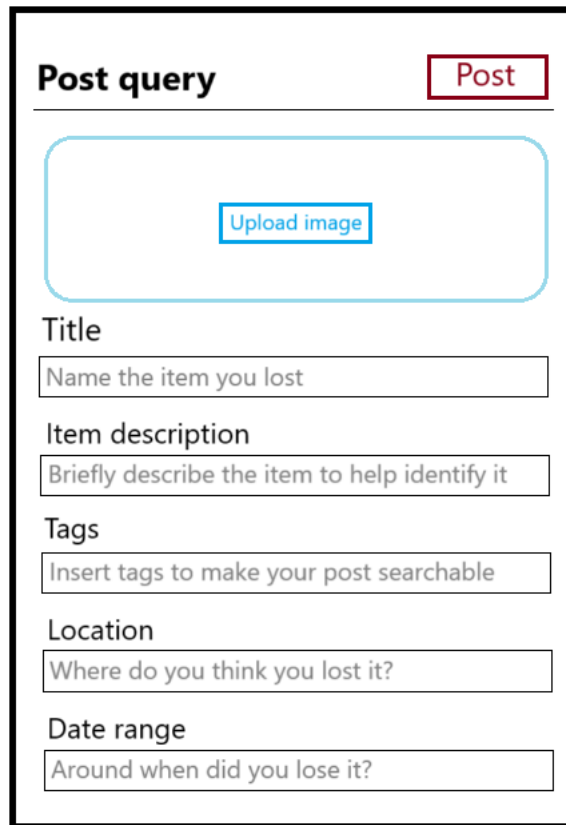
**Logical Characteristics:**

- Displays an overview of lost and found items categorized by type (e.g., electronics, personal belongings).
- Includes shortcuts to key functionalities like posting an item and recent notifications.

User Interaction:

- Swipe to scroll through categories.
- Tap a category or item for more details.
- Access navigation via a bottom menu bar or a hamburger menu.

3.1.1.3 Post Lost/Found Item Page



The form is titled "Post query" and has a "Post" button in the top right corner. It contains several input fields: an "Upload image" button inside a large rounded rectangle, a "Title" field with the placeholder "Name the item you lost", an "Item description" field with the placeholder "Briefly describe the item to help identify it", a "Tags" field with the placeholder "Insert tags to make your post searchable", a "Location" field with the placeholder "Where do you think you lost it?", and a "Date range" field with the placeholder "Around when did you lose it?".

Logical Characteristics:

- Guided form with fields for item description, category, location, and an option to upload photos.
- Drop-down menus and auto-fill suggestions for commonly lost items.

User Interaction:

- Users fill in details, select options, and upload photos directly from their device's camera or gallery.
- Tap Submit to post the item.

3.1.1.4 Search and Filter Interface

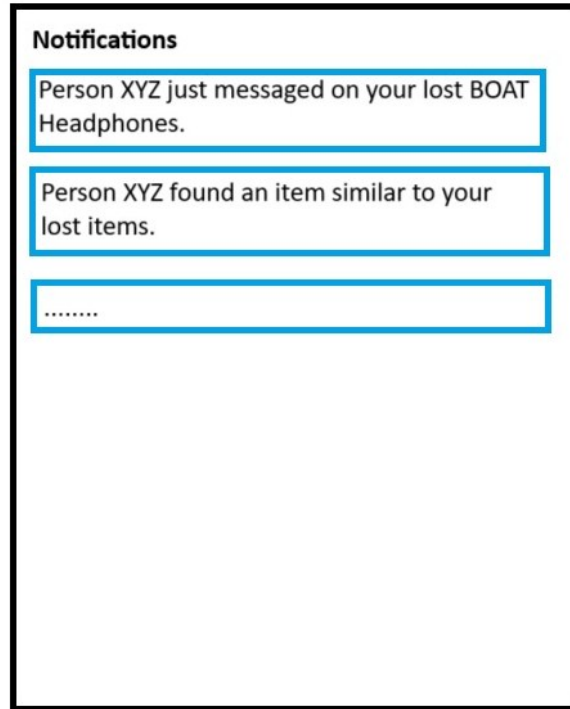
Logical Characteristics:

- A search bar at the top allows users to type keywords.
- Filters (e.g., date, category, location) are used to narrow down results.

User Interaction:

- Users type in keywords and refine their search using filters.
- Results appear dynamically as users adjust criteria.

3.1.1.5 Notifications Page

**Logical Characteristics:**

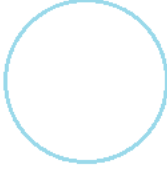
- Displays personalised notifications, such as updates on posted items.
- Organized chronologically, with read/unread status indicators.

User Interaction:

- Tap on notifications to view details or take action (e.g., claim an item).
- Swipe to dismiss notifications.

3.1.1.6 Profile Page

Profile Edit



First M. Last

Primary email

Phone number

Address

Designation

Logical Characteristics:

- Shows user details, activity history, and settings.
- Includes options to edit profile information and manage preferences like notification settings.

User Interaction:

- Users tap to edit information or view past posts.
- Toggle settings (e.g., enable dark mode or manage language preferences).

3.1.1.7 Help & Support Page

Logical Characteristics:

- Provides FAQs, contact options, and guidelines for using the app effectively.

User Interaction:

- Users browse FAQs or initiate a support ticket through a guided form.

The aforementioned pages and interfaces use graphic components to facilitate interaction by the user. The main components are as described below:

- **Navigation design:** The top navigation bar features the app logo, a search icon, and a user profile icon. The bottom navigation bar provides quick access to Home, Notifications, Post Item, Search, and Profile pages.

- **Menu design:** A hamburger menu on the side panel opens a list of all major functionalities for easy navigation.
- **Graphical elements:** Use of clean icons for actions like adding, searching, and editing.
- **Interaction elements:** Buttons shall serve as the primary interaction element, clearly labelled with actions like Submit, Search, and View Details.

3.1.2 Hardware Interfaces

The Lost and Found application interacts with the following hardware components during its operation:

- **Smartphones and Tablets:** The primary devices used by users to access the android application.
- **Desktops/Laptops:** Used by administrators or users who prefer accessing the application via an emulator for reporting or managing items.
- **Cameras:** Integrated with smartphones or tablets for capturing photos of lost or found items during the reporting process.

These interfaces ensure that the application functions seamlessly across various devices, providing a consistent and user-friendly experience.

3.1.3 Software Interfaces

During the development of the Lost and Found mobile application using Flutter and Flask, we require a range of software interfaces and tools to ensure efficient development and functionality. Below are the software requirements:

- **Flutter:** Used to develop the cross-platform mobile application, providing a responsive and visually appealing interface for users.
- **Flask:** A lightweight Python web framework for backend development, managing API communication and server-side logic.
- **API integration tools:** To integrate third-party services like push notifications or external ID verification systems.
- **Web browser (with internet access):** Required for debugging, testing, and accessing development tools.
- **Linux/Windows/Mac:** Operating systems for the development environment.

These tools and interfaces ensure that the application is user-friendly, secure, and efficient, while enabling developers to create and test functionalities effectively.

3.2 Functional Requirements

3.2.1 F1: Account Setup and Authentication

The login screen will allow users to create and manage their personal accounts for the app, ensuring secure access to the platform. In particular, the following functionalities are provided:

- **Sign up:** Users must be able to register using their IIT Kanpur credentials (e.g., student email, faculty email). The email address serves as proof that the user is associated with IIT Kanpur.

- **Log in:** The user can set a password upon first use of the program, after which the user remains signed in until the user manually signs out or the app is uninstalled. Upon reinstallation, the user can authenticate using the same credentials.
- **Profile management:** Users can update their profile details such as name, contact information, and photo, which will be displayed to other users during interactions or in posts.
- **Account recovery:** If users forget their login credentials, they can reset their passwords through a secure process.
- **Log out:** Users can log out of the app manually.

3.2.2 F2: Item Posting and Categorisation

The app shall allow users to report lost or found items in an organised manner, making it easier for others to locate them.

- **Posting items:** Users can post information about lost or found items by providing essential details such as item description, location, date, and an image (optional for those reporting a lost item).
- **Item categorisation:** Each item is categorised into predefined types (e.g., electronics, clothing, books) to improve searchability and organisation.
- **Item tags:** Users can add tags to their posts for more specific identification (e.g., 'wallet', 'keys').
- **Editable posts:** Users can edit or delete their own posts if the item is found or recovered.

3.2.3 F3: Search and Filter

The app shall include a search facility whereby users can easily search for and filter items based on specific criteria.

- **Search functionality:** Users can search for lost or found items using keywords. Item tags, categories, location, or date of posting can be used to filter results.
- **Sorting:** Users can sort the search results by date.

3.2.4 F4: Interactive Chat and Forums

The platform shall facilitate communication between users to help them claim or return lost items.

- **In-app chat:** Users can initiate private chats with the person who posted the item (either lost or found), allowing them to discuss details like the location and condition of the item.
- **Message notifications:** Users are notified of new messages in the chat or forum threads to ensure timely communication.

3.2.5 F5: Customised Notifications

Users will be kept informed about the status of their items and relevant updates through push notifications.

- **Lost item updates:** Users will receive notifications when an item they have reported as lost is found.
- **Found item alerts:** Users will get alerts when someone posts a found item that may belong to them.

- **General announcements:** Notifications for system updates, new features, or campus-wide lost-and-found initiatives can be sent.

3.2.6 F6: Verification and Claim System

Ensure that lost items are correctly claimed by the rightful owners in a secure manner.

- **Item verification:** When users claim an item, they must provide sufficient proof (e.g., description, photo) to verify ownership.
- **Claim process:** Users can request to claim a found item. The item's poster will review the claim and approve or deny it based on the verification process.
- **Ownership confirmation:** Once ownership is confirmed, the app updates the status of the item to 'recovered'.

3.2.7 F7: Assistance

The app shall include a support section to guide users in using the app and to provide assistance to users who are struggling to find their lost items.

- **Help section:** Users can access resources or guides on how to report lost items effectively.
- **Admin support:** Admins can help resolve disputes or assist in item recovery based on their experience or access to additional resources.

3.2.8 F8: Moderation and Reporting

Manual intervention may be necessary to ensure that the platform is used in a responsible manner and to prevent misuse. The system shall provide facilities common to all users as well as special permissions to admins for the same.

- **Content moderation:** Admins can review and approve posts to ensure they meet the app's guidelines.
- **Flagging system:** Users can flag inappropriate posts (e.g., false claims, spam) for review by moderators.
- **User reporting:** If a user is not adhering to the platform's rules, other users can report them for misconduct.
- **Action enforcement:** Admins can take action on flagged items, such as removing posts or banning users, to maintain the integrity of the platform.

3.2.9 F9: Dark Mode and Theme Customisation

To enhance user experience, the app shall provide visual preferences and customisation options.

- **Dark mode:** Users can toggle between light and dark modes to reduce eye strain, especially in low-light environments.
- **Custom themes:** The app offers customisable themes and colour schemes, allowing users to personalize the visual experience to suit their preferences.
- **Font adjustments:** Users can adjust font sizes for better readability based on their accessibility needs.

3.3 Use Case Model

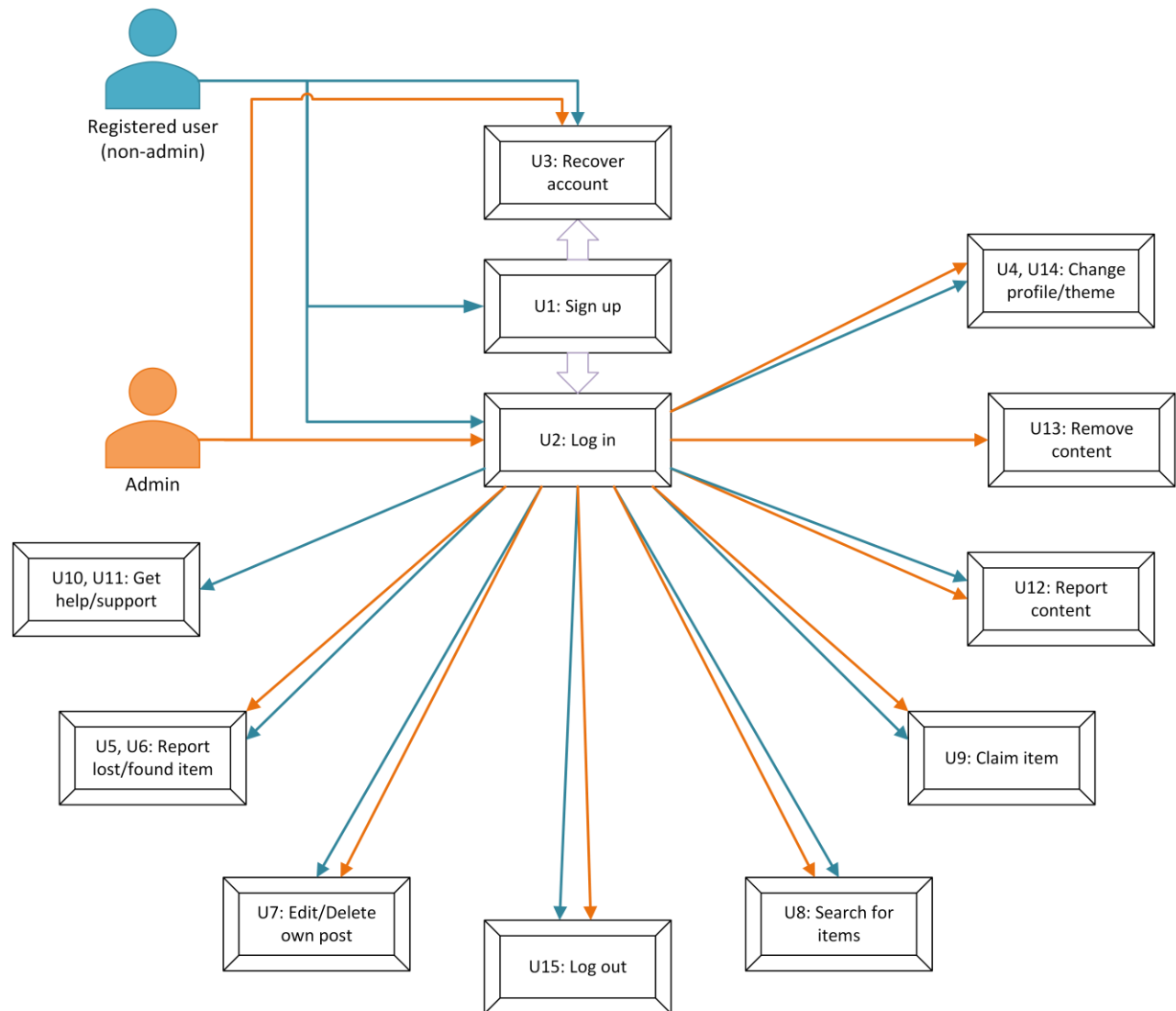


Figure 1: Use case diagram

3.3.1 U1: Sign Up

Author

[Satwik Raj Wadhwa](#)

Purpose

Creation of an account to authenticate with the app using the Institute email address with the specified permission level.

Requirements Traceability

[F1](#)

Priority

High.

Preconditions	<ul style="list-style-type: none"> The user must have an @iitk.ac.in email address and be able to verify ownership of the address. The app must not be signed in to. The app must have access to the Internet.
Postconditions	An account is created for the user with the specified permission level.
Actors	Members of the campus community.
Exceptions	<ul style="list-style-type: none"> The user may not have an @iitk.ac.in email address. An account will not be created. Email verification may fail. An account will not be created. An account may already exist for the given email address. A new account will not be created. The user is not authorised to sign up with the specified permission level. An error message will be displayed. Internet access may not be available. An error message will be displayed.
Includes	None.
Notes	A passphrase must be provided for use during login.

3.3.2 U2: Log In

Author	Satwik Raj Wadhwa
Purpose	To log in to the app using the credentials supplied during sign-up.
Requirements Traceability	F1
Priority	High.
Preconditions	<ul style="list-style-type: none"> The user must already have signed up. The app must not be logged in to. The app must have access to the Internet.
Postconditions	If the passphrase supplied is valid, the app is logged in to and the home tab is displayed. Else, the user is asked to retry.
Actors	Registered users.
Exceptions	Internet access may not be available. An error message will be displayed.
Includes	U1
Notes	

3.3.3 U3: Recover Account

Author	Krishna Kumayu
Purpose	To reset the account password if the user forgets the password set.
Requirements Traceability	F1
Priority	Medium.
Preconditions	<ul style="list-style-type: none">• The user must already have signed up.• The app must not be logged in to.• The app must have access to the Internet.• The user must have access to the email address associated with the account.
Postconditions	The user is prompted to set a new password. Alternatively, a system-generated password is set, and the user is redirected to the login page.
Actors	Registered users.
Exceptions	Internet access may not be available. An error message will be displayed.
Includes	U1
Notes	To authenticate before password reset, the user may be asked to enter an OTP sent on the registered email address. Alternatively, a new password is generated at random and set for the account; thereafter, it is emailed to the registered email address.

3.3.4 U4: Update Profile

Author	Aayush Kumar
Purpose	To update the profile of the user, including their name, profile picture, and contact details.
Requirements Traceability	F1
Priority	Low.
Preconditions	<ul style="list-style-type: none">• The app must be logged in to.• The app must have access to the Internet.
Postconditions	The user profile is updated with the new details. The change is propagated to the server.

Actors	Registered users.
Exceptions	Internet access may not be available. An error message will be displayed.
Includes	U2
Notes	

3.3.5 U5: Report Lost Item

Author	Krishna Kumayu
Purpose	To post details of a lost item.
Requirements Traceability	F2
Priority	High.
Preconditions	<ul style="list-style-type: none"> • The app must be logged in to. • The app must have access to the Internet.
Postconditions	The post details are relayed to the server. After successful database update, the app is refreshed to reflect the new post, which is publicly visible.
Actors	Registered users.
Exceptions	<ul style="list-style-type: none"> • Internet access may not be available. An error message will be displayed. • The server may not be online. An error message will be displayed, or the post will be queued.
Includes	U2
Notes	<ul style="list-style-type: none"> • The post cannot be submitted till all <i>required</i> fields are filled. • Posts may be subject to admin review before database update.

3.3.6 U6: Report Found Item

Author	Krishna Kumayu
Purpose	To post details of a found item that was apparently misplaced.
Requirements Traceability	F2
Priority	High.

Preconditions	<ul style="list-style-type: none"> • The app must be logged in to. • The app must have access to the Internet.
Postconditions	The post details are relayed to the server. After successful database update, the app is refreshed to reflect the new post, which is publicly visible.
Actors	Registered users.
Exceptions	<ul style="list-style-type: none"> • Internet access may not be available. An error message will be displayed. • The server may not be online. An error message will be displayed, or the post will be queued.
Includes	U2
Notes	<ul style="list-style-type: none"> • The post cannot be submitted till all <i>required</i> fields are filled (this includes an image of the item found). • Posts may be subject to admin review before database update.

3.3.7 U7: Edit/Delete Own Post

Author	Anirudh Cheriyanaseri Bijay
Purpose	To edit or delete one's own post.
Requirements Traceability	F2
Priority	Medium.
Preconditions	<ul style="list-style-type: none"> • The app must be logged in to. • The app must have access to the Internet. • The post must not be closed.
Postconditions	<ul style="list-style-type: none"> • If a post is edited, the edit details are relayed to the server. After successful database update, the app is refreshed to reflect the edited post, which is publicly visible. • If a post is deleted, the database is updated and the app is refreshed to reflect the deletion.
Actors	Registered users.
Exceptions	<ul style="list-style-type: none"> • Internet access may not be available. An error message will be displayed. • The server may not be online. An error message will be displayed, or the request will be queued. • The post has been closed (<i>i.e.</i>, a found item has been marked returned). An error message will be displayed.

Includes	U6 , U7
Notes	Edits may be subject to admin review before database update.

3.3.8 U8: Search for Items

Author	Satwik Raj Wadhwa
Purpose	To search for a post regarding a lost/found item using keywords and filters based on date, location, category etc.
Requirements Traceability	F3
Priority	High.
Preconditions	The app must be logged in to.
Postconditions	Search results are displayed as per the query.
Actors	Registered users.
Exceptions	None.
Includes	U2
Notes	If the app is offline, the results shown may be from cached posts.

3.3.9 U9: Claim Item

Author	Krishna Kumayu
Purpose	To notify a poster that the posted item may belong to the user (if the post is for a found item) or the poster (if the post is for a lost item) and to open a chat with him/her.
Requirements Traceability	F4 , F6
Priority	Medium.
Preconditions	<ul style="list-style-type: none"> • The app must be logged in to. • The app must have access to the Internet.
Postconditions	The request is relayed to the server. After successful database update, the app is refreshed to reflect the action and the poster is notified.
Actors	Registered users.

Exceptions	<ul style="list-style-type: none"> • Internet access may not be available. An error message will be displayed. • The server may not be online. An error message will be displayed, or the request will be queued.
Includes	U2
Notes	

3.3.10 U10: Access Help

Author	Aayush Kumar
Purpose	To access the guide on using the app.
Requirements Traceability	F7
Priority	Low.
Preconditions	None.
Postconditions	The user gets either a link to the manual for the app, or an instructions page.
Actors	Registered users.
Exceptions	None.
Includes	None.
Notes	

3.3.11 U11: Request Support

Author	Aayush Kumar
Purpose	To request help from admins in case of technical issues or disputes.
Requirements Traceability	F7
Priority	Low.
Preconditions	<ul style="list-style-type: none"> • The app must be logged in to. • The app must have access to the Internet.
Postconditions	Admins are notified of the request.

Actors	Registered users except admins.
Exceptions	<ul style="list-style-type: none"> • Internet access may not be available. An error message will be displayed. • The server may not be online. An error message will be displayed.
Includes	U2
Notes	

3.3.12 U12: Report Inappropriate Content

Author	Anirudh Cheriyanaseri Bijay
Purpose	To report irrelevant, abusive, or otherwise inappropriate posts and chats for removal.
Requirements Traceability	F8
Priority	Low.
Preconditions	<ul style="list-style-type: none"> • The app must be logged in to. • The app must have access to the Internet.
Postconditions	The action is logged at the server. If a post/account is found to have received more reports than a threshold, admins are notified.
Actors	Registered users.
Exceptions	<ul style="list-style-type: none"> • Internet access may not be available. An error message will be displayed. • The server may not be online. An error message will be displayed.
Includes	U2
Notes	The user may be asked to specify the reason for reporting.

3.3.13 U13: Remove Inappropriate Content

Author	Anirudh Cheriyanaseri Bijay
Purpose	Removal of content that violates the platform's policies.
Requirements Traceability	F8
Priority	Low.

Preconditions	<ul style="list-style-type: none"> • The app must be logged in to. • The app must have access to the Internet.
Postconditions	The action is logged at the server. The database is updated and the deletion is propagated to all users.
Actors	Admins.
Exceptions	<ul style="list-style-type: none"> • Internet access may not be available. An error message will be displayed. • The server may not be online. An error message will be displayed.
Includes	U2 , U5
Notes	

3.3.14 U14: Customise Theme

Author	Anirudh Cheriyanaseri Bijay
Purpose	To customise the display theme of the app.
Requirements Traceability	F9
Priority	Low.
Preconditions	The app must be logged in to.
Postconditions	The changes are implemented after the user confirms his choices.
Actors	Registered users.
Exceptions	None.
Includes	U2
Notes	

3.3.15 U15: Log Out

Author	Satwik Raj Wadhwa
Purpose	Logging out of the app.
Requirements Traceability	F1
Priority	High.

Preconditions	The app must be logged in to.
Postconditions	The user is redirected to the login page.
Actors	Registered users.
Exceptions	None.
Includes	U2
Notes	

4 Other Non-functional Requirements

4.1 Performance Requirements

The application is required to provide provides a seamless and efficient user experience under various conditions. These requirements are based on discussions with stakeholders and the anticipated usage patterns.

4.1.1 General Performance Requirements

- **Response time:** The application shall respond to user actions, such as navigating pages or submitting a post, within 30 seconds under normal usage conditions. Loading the user profile or retrieving search results shall not exceed 50 seconds.
- **Concurrency:** The application shall support at least 1,000 concurrent users without degradation in performance.
- **Data processing time:** The system shall process and store a user-submitted post (with images and text) within 30 seconds after submission.
- **Search functionality:** The search functionality shall display results for a query within 30 seconds, even when filtering and sorting are applied.

4.1.2 Mobile-specific Performance Requirements

- **App launch time:** The mobile application shall launch within 30 seconds on devices meeting the minimum hardware requirements.
- **Offline mode:** Core features, such as viewing saved posts, shall function offline and sync with the server within 30 seconds of reconnecting to the internet.

4.1.3 Real-time Requirements

- **Real-time notifications:** The system shall push real-time notifications to users within 10 second of an event trigger, such as a response to a post.

4.2 Safety and Security Requirements

- Login credentials shall be stored encrypted on the database to ensure the user's privacy.
- Users' IP addresses will be logged.
- The system shall be protected against vulnerabilities such as SQL injection attacks.
- Registration shall be based on Institute-provided email to keep the application local to the campus community.

4.3 Software Quality Attributes

The app must meet specific software quality attributes to ensure a robust, user-friendly, and maintainable system. Below, we outline the critical quality attributes, their definitions, and how they will be achieved.

4.3.1 Reliability

Definition:

Reliability refers to the ability of the system to operate correctly under specified conditions for a defined period without failure.

Requirements:

- The system shall maintain 99.9% uptime, allowing users to access services with minimal interruption.
- Data integrity must be ensured during all operations, with regular database backups occurring every 12 hours.
- The app shall recover from unexpected crashes within 2 minutes and restore user progress (e.g., partially filled forms).

Implementation Approach:

- Implement automated testing for critical features to detect and resolve issues before deployment.
- Use fault-tolerant server architectures and cloud services to prevent downtime due to hardware failures.
- Log all critical errors for post-incident analysis and continuous improvement.

4.3.2 Usability

Definition:

Usability focuses on how easy it is for users to learn, navigate, and efficiently interact with the app.

Requirements:

- The app shall have an intuitive user interface, with all primary features (e.g., posting lost items, searching) accessible within 3 clicks.
- New users shall be able to complete their first post within 5 minutes, aided by a guided walkthrough or tutorial.

Implementation Approach:

- Conduct user experience (UX) testing with a diverse audience to identify and resolve pain points.
- Use a clean, responsive design optimized for Android devices with various screen sizes and resolutions.
- Include tooltips, placeholders, and error messages to guide users through complex actions.

4.3.3 Maintainability

Definition:

Maintainability ensures that the software can be easily updated or modified to meet changing requirements.

Requirements:

- The codebase shall adhere to standard conventions (e.g., clean coding practices) to facilitate easy debugging and enhancements.
- Updates or new feature implementations shall not require more than 3 developer days for existing modules.

- Detailed documentation (e.g., API references, architecture diagrams) shall be maintained and updated with each software release.

Implementation Approach:

- Use modular development techniques, allowing independent updates to specific components.
- Employ version control systems (e.g., Git) to manage changes efficiently and prevent conflicts.
- Conduct regular code reviews to ensure adherence to maintainability standards.

4.3.4 Security**Definition:**

Security ensures that the app protects user data from unauthorized access and breaches.

Requirements:

- All user data (e.g., passwords, personal information) shall be encrypted.
- The system shall be tested for vulnerabilities using industry-standard penetration testing methods before deployment.

Implementation Approach:

- Utilise HTTPS for all communications between the app and the server to prevent data interception.
- Monitor suspicious activity and implement automated lockout mechanisms for failed login attempts.
- Store login status tokens securely.
- Apply security patches regularly to address known vulnerabilities.

4.3.5 Portability**Definition:**

Portability allows the same source code to be compiled for different architectures, operating systems and classes of devices, either for versatility or for extensibility.

Requirements:

- The source code must not rely heavily on platform-dependent features or native APIs.
- The development framework must be targeted at cross-platform development.

Implementation Approach:

- The SDK used for the frontend is Flutter. This enables the application to later be ported to desktops and the web, as the need may arise.
- The use of features native to Android shall be kept to a minimum.

4.3.6 Adaptability**Definition:**

Adaptability ensures that the app can be modified to support new features or integrate with external systems without significant redesign.

Requirements:

- The system shall support the addition of new features without requiring a complete overhaul.

Implementation Approach:

- Design the app using a microservices architecture, allowing independent updates to specific components.
- Use APIs and standardised data formats (e.g., JSON, XML) for external integrations.

Appendix A – Data Dictionary

Field Name	Description	Possible States/Values	Operations/ Functionality	Constraints/ Requirements
userEmail	Stores the email address of the user during login or signup.	Valid email format (for instance, name@iitk.ac.in).	Validate format, authenticate user, associate with profile.	Must match IITK email domain (@iitk.ac.in).
loginStatus	Stores the login status (logged in/not logged in) of the user.	Boolean.	Identifies whether the app is logged in to or not.	Token must be stored securely locally.
userRole	Tracks the role of the user in the system.	Standard user, Security, Admin	Determine access permissions for features.	Role-based restrictions apply.
itemStatus	Tracks the status of a reported item.	Lost, Found, Claimed	Update based on user interactions, e.g., marking as claimed.	Only valid transitions allowed (e.g., Lost → Claimed).
searchQuery	User input to search for items using keywords.	Any string.	Filter and match items from the database.	None.
itemCategory	Categorises reported items for easier filtering.	Electronics, Clothing etc.	Used during item posting and searching.	Must match predefined categories.
claimantID	Unique identifier for the user attempting to claim an item.	UUID format.	Associate claimant with item.	Must be valid and match user records.

Appendix B – Group Log

Date & Time	Venue	Minutes
09/01/25 3:00 pm	4 th floor, Rajeev Motwani Building	Brainstormed project ideas and settled down on a lost-and-found application. Decided the target platform to be Android devices. Discussed the basic features and initial scope of the application.
16/01/25 3:00 pm	4 th floor, Rajeev Motwani Building	Assigned the development of the Software Requirements Specification (SRS) document to Aayush Kumar, Anirudh Cheriyanaseri Bijay, Krishna Kumayu, and Satwik Raj Wadhwa. Chose Flutter for frontend development and Flask for backend. Came up with a more refined description of the feature set for the app.
18/01/25 9:30 pm	1 st floor, Rajeev Motwani Building	<i>Meeting limited to members tasked with SRS development.</i> Evolved functional and non-functional requirements aligned with the working and the feature set of the app. Benchmarks decided for measurable requirements. Use cases and dependencies spelt out.
21/01/25 5:00 pm	302, Rajeev Motwani Building	<i>Meeting with TA.</i> Discussed the overall idea and use cases with the TA and confirmed the feasibility of the idea.
23/01/25 3:00 pm	4 th floor, Rajeev Motwani Building	Final review of the SRS document with all team members. Commenced interface design with rough outlines developed. Assigned the design of wireframes in Figma to Ayush Patel, Marada Teja Satvik, Somaiya Narayan Aniruddh, and Shaurya Johari.