Software Requirements Specification

for

DealSimplified

Version 1.0

Prepared by

Group 14 **Group Name: Kasukabe Defence Group**

Esra Fatima	220391	esra22@iitk.ac.in
Kanika Chaturvedi	220497	kanikac22@iitk.ac.in
Krishiv Geriani	220545	krishivg22@iitk.ac.in
Manavjeet Singh	220616	manavjeetw22@iitk.ac.in
Rachit Choudhary	220845	rachitc22@iitk.ac.in
Riddhima Vijayvargiya	220883	riddhima22@iitk.ac.in
Ritul	220896	ritul22@iitk.ac.in
Riya Agarwal	220899	ariya22@iitk.ac.in
Vishal Singh	221207	vishals22@iitk.ac.in
Vishap Raj	221208	vishapraj22@iitk.ac.in

Course: CS253
Mentor TA: Vikrant Chouhan
Date: 24-1-25

CONTENTS II

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
Version 1.0	Esra Fatima Kanika Chaturvedi Krishiv Geriani Manavjeet Singh Rachit Choudhary Riddhima Vijayvargiya Ritul Riya Agarwal Vishal Singh Vishap Raj	First Draft	24/01/2025

1 Introduction

1.1 Product Scope

DealSimplified is an innovative platform designed to address two key aspects of campus life at IIT Kanpur: managing lost and found items and facilitating an in-campus marketplace for buying and selling pre-owned goods. The portal provides a centralized system for reporting, searching, and recovering lost items and an intuitive marketplace for campus residents to trade items securely and efficiently.

The primary goal of **DealSimplified** is to simplify everyday challenges by offering users an organized and mobile-friendly solution. For the lost and found feature, users can upload details of lost or found items, including photos and descriptions, and benefit from automated matching algorithms and real-time notifications. For the marketplace, users can list items with descriptions, photos, and prices, and communicate with potential buyers or sellers directly through the platform. This project aims to promote a sense of community, enhance trust, and foster sustainable practices by creating a seamless and efficient platform for campus-based transactions.

1.2 Intended Audience and Document Overview

The Software Requirement Document (SRS) is a crucial document in the software development life cycle, and it is intended for various readers involved in different stages of the project. The primary audience includes:

- **Developers:** The SRS serves as a detailed guide for developers, providing them with insights into the system's functional and non-functional requirements. It outlines the features, constraints, and specifications that need to be implemented, helping the development team understand the scope and expectations.
- **Customers/Users:** For customers or end-users, the SRS is a crucial document that communicates what the software will do, its features, and how it will benefit them. It sets the foundation for aligning customer expectations with the final product, ensuring that the delivered software meets their needs and requirements.
- **Testers:** Testers rely on the SRS to create test cases and validate that the software functions as intended. The document provides a basis for developing test scenarios and ensures that the testing process aligns with the specified requirements. It becomes a reference point for evaluating the software's correctness and performance.

The Software Requirements Specification (SRS) is a crucial document in software development, highlighting its essential role in project success through clear requirements. It starts with an overview of the software's purpose and functionalities, followed by detailed descriptions of user interfaces to offer stakeholders insights into user interactions. The document concludes with use cases, illustrating expected software behavior in real-world scenarios. This structured approach guides stakeholders, fostering a collective understanding of the software's objectives and requirements, establishing a strong foundation for the development process.

1.3 Definitions, Acronyms and Abbreviations

Abbreviations/Acronyms	Definitions
CSS	Cascading Style Sheets
HTML	HyperText Markup Language
JS	Java Script
REST	Representational State Transfer
SQL	Structured Query Language
API	Application Programming Interface

1.4 Document Conventions

The document follows the IEEE formatting requirements

- Aerial font size 11 is used.
- Document is single spaced.
- 1” margins are maintained.
- Bullet point ordering has been used as a listing type setting tool.

1.5 References and Acknowledgments

- *User Interface components were designed using Figma.*
- *Prof. Idranil Saha, for providing the SRS template document and teaching the required concepts.*
- *Our TA, Vikrant Chauhan, for his valuable inputs from time to time in creating this docum*

2 Overall Description

2.1 Product Overview

Context and Origin:

DealSimplified: A Unified Lost & Found and Marketplace Portal is a new, self-contained product designed specifically for the IIT Kanpur campus community. This platform addresses two major campus needs: efficient Lost & Found management and a convenient, localized marketplace for buying, selling, and exchanging goods. It aims to foster engagement within the campus community while streamlining processes that were previously fragmented or inefficient. By integrating advanced algorithms and user-friendly interfaces, DealSimplified eliminates the need for multiple, scattered solutions, creating a unified and comprehensive platform.

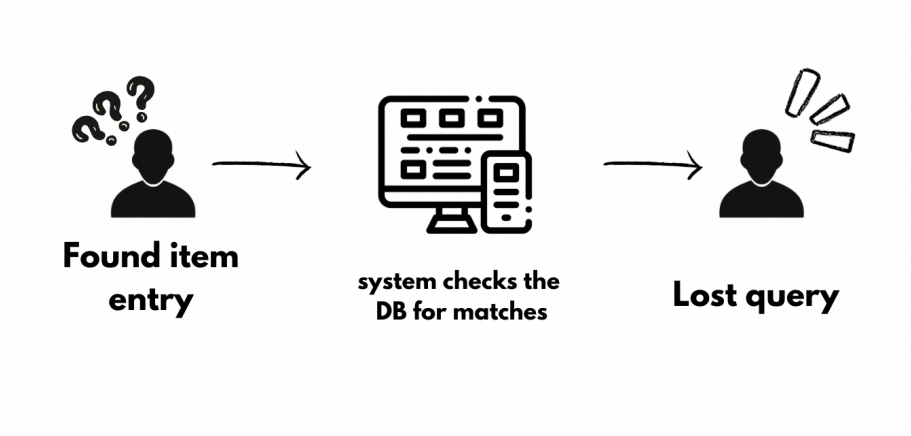
Product Perspective:

DealSimplified is an independent product that interacts with users through web and mobile applications. It leverages machine learning for intelligent matching and fair price suggestions, along with automated notification systems for real-time updates. The platform connects with existing campus systems such as user authentication databases for secure logins, ensuring that only authorized students and staff can use it. While this product is a standalone solution, it integrates seamlessly with campus email and notification systems to enhance user communication.

2.2 Product Functionality

The DealSimplified platform provides the following major functionalities:

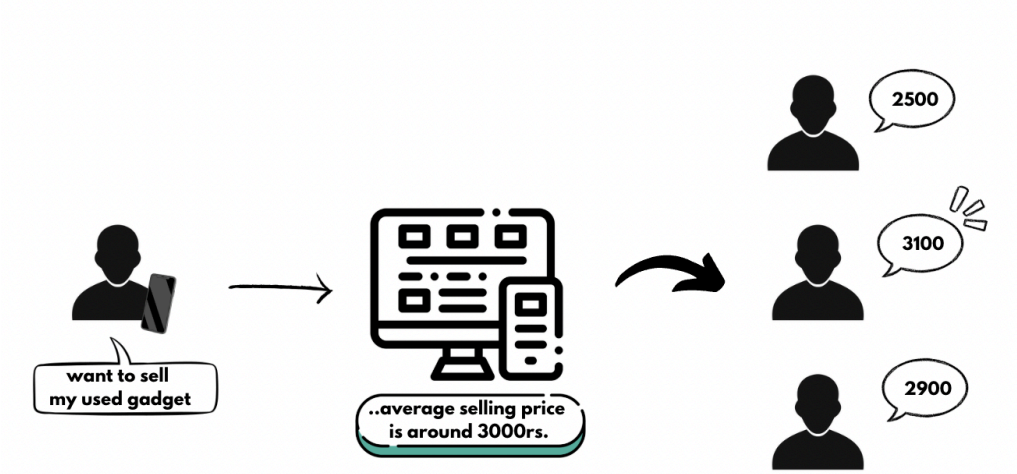
Lost & Found Features:



- Reporting System:
    - Users can report lost or found items with detailed descriptions, photos, dates, locations, and categories.
- Automated Matching & Notifications:

    - Smart algorithms analyze item descriptions and photos to match lost items with found ones.
    - Users receive automated notifications for potential matches.
- Item Categorization and Search Filters:
    - Items are categorized into groups such as cycles, wearables, electronics, and ID cards.
    - Filters for date, location, and category refine user searches.
- Admin Approval for Claims:
    - Admins verify ownership claims to ensure secure item recovery, using questioning or proof of ownership.

Marketplace Features:

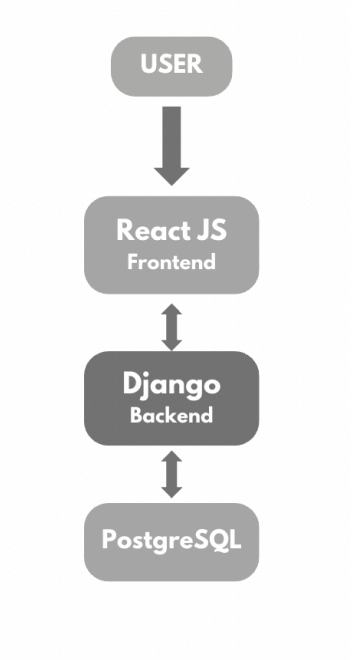


- Listings for Buying & Selling:
    - Users can post items with descriptions, photos, and pricing.
    - An ML model suggests a fair price based on the item's age and condition.
- Messaging System:
    - Facilitates communication between buyers and sellers.
- Search Filters and Notifications:
    - Filters for categories like electronics, furniture, and books.
    - Automated notifications alert users about items matching their preferences.
- Wishlist Feature:
    - Buyers can create wishlists for desired items and receive alerts when matching items are listed.
- Bidding System:
    - Sellers can enable a bidding option for competitive pricing.
- Trade Option:
    - Facilitates item exchanges instead of direct purchases (e.g., books for electronics).

2.3 Design and Implementation Constraints

Hardware Requirements:

- The application requires a working computer or mobile network with internet access for it to function properly.
- The mobile device must have at least 2 GB of free RAM and minimum 0.1 Mbps of internet speed for the application to open.



Software Requirements:

- Our software is designed to work on Windows, Linux and MAC OS.
- It needs a web browser such as Google Chrome, Mozilla Firefox, Bing, Internet Explorer or Opera to work.
- The software and its dependencies may require an operating system which runs on 64-bit architecture and may also require the latest version of Android/IOS/Google Chrome for proper functioning.

2.4 Assumptions and Dependencies

- Users must be proficient in the use of mobile / web applications and have a basic understanding of their working.
- Users require a working email-ID for them to register and login. The email will be used for verification and authentication, as well as to receive updates from the software.
- Admins and sellers must regularly check the platform to cater to user requirements, update item listings, and respond to buyer-seller communication.
- The web app shall be hosted on a web hosting service (in any cloud platform).
- The database used will be stored and accessed through a cloud platform

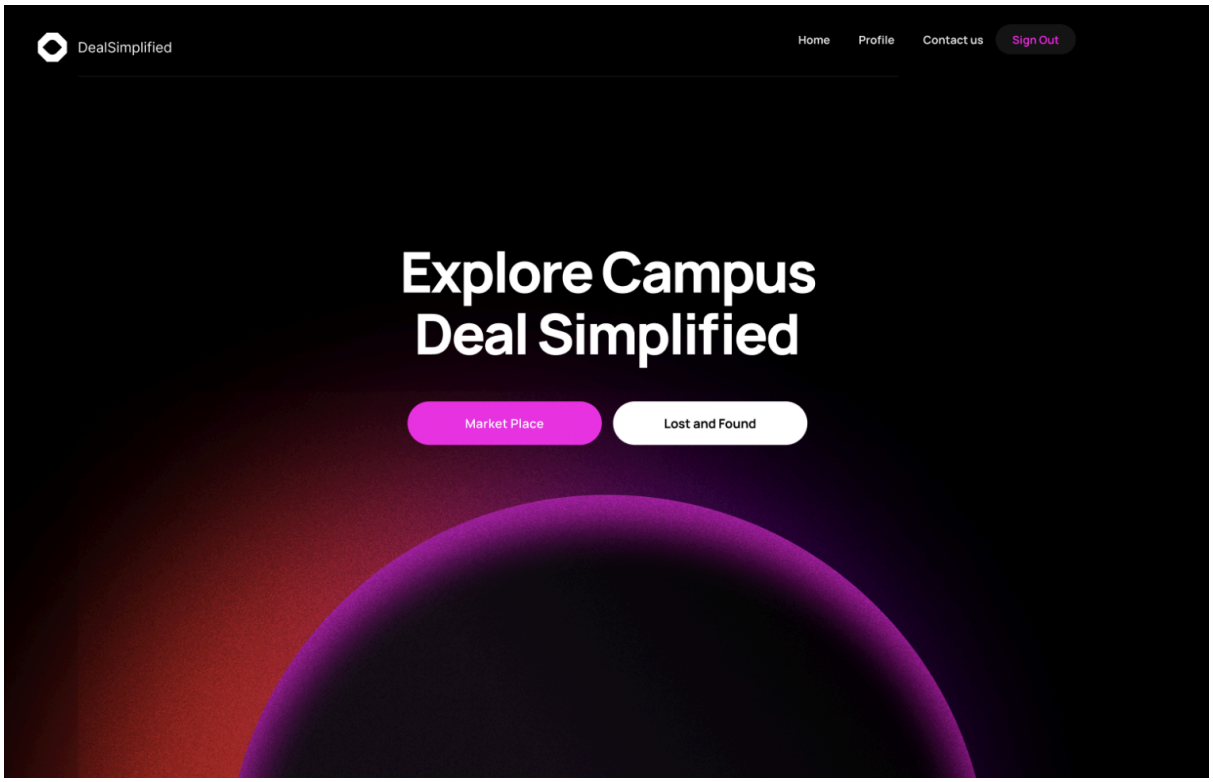
3 Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

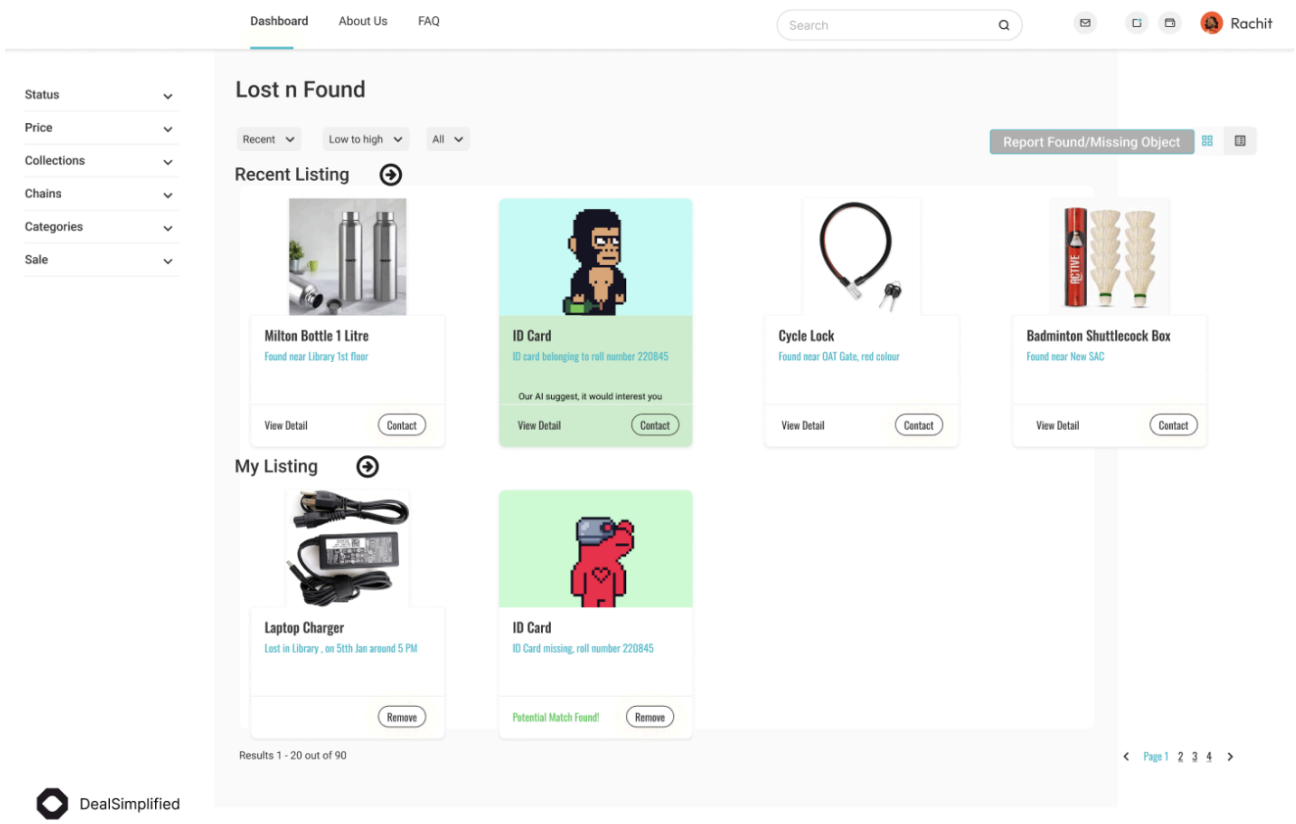
1. Entry Page:

When users access the DealSimplified platform, they are first presented with two options: “Lost & Found” and “Marketplace”. After selecting the appropriate option, users are prompted to sign in with their credentials or an ID card scan.
For new users, the system offers an easy sign-up process where they can create an account to get started by providing: **Name**, **Roll No.**, **Email Id** and Address at IITK(Optional) or uploading their ID card.



2. Dashboard (Lost & Found):

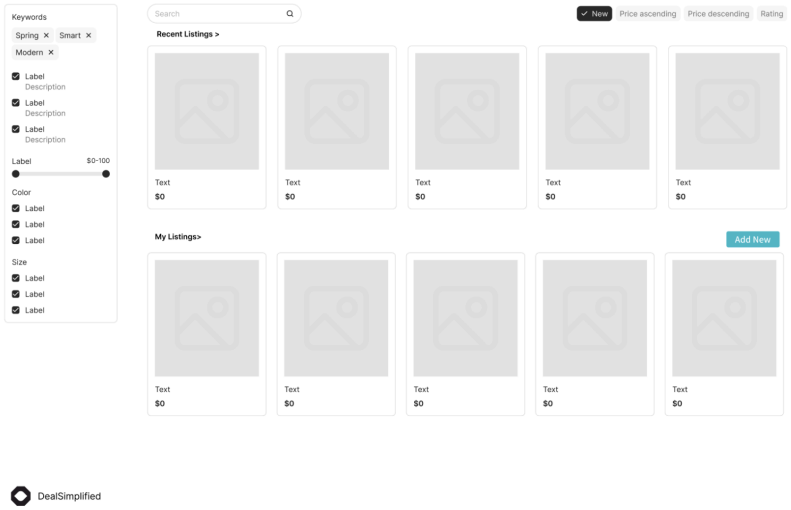
The Lost & Found Dashboard is designed to help users easily manage their lost and found reports. It includes a navigation bar with options like "Report Lost Item," "Report Found Item," and "My Reports" for quick access to different sections.
A search bar lets users search for items using keywords or filters, making it easy to find specific lost or found items. Notifications for potential matches will appear, keeping users informed of any updates. This setup ensures a smooth and straightforward experience for managing reports.

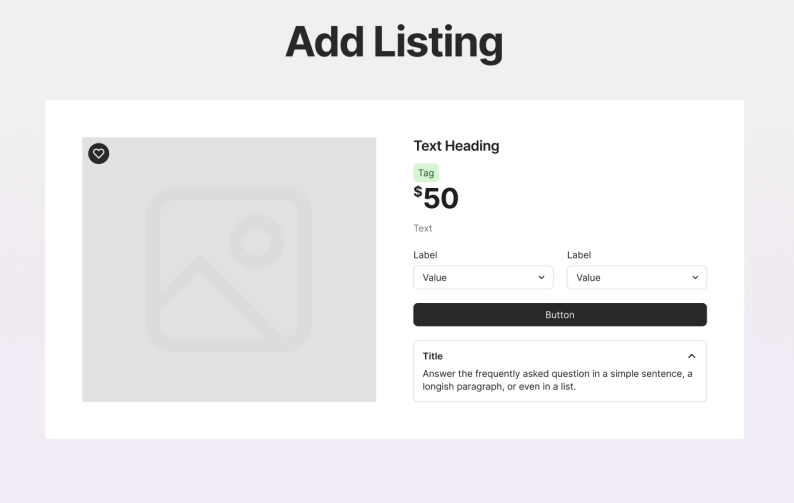


3. Dashboard (Marketplace):

The Marketplace dashboard includes sections like "My Listings," "Recent Listings," and "Browse Items."

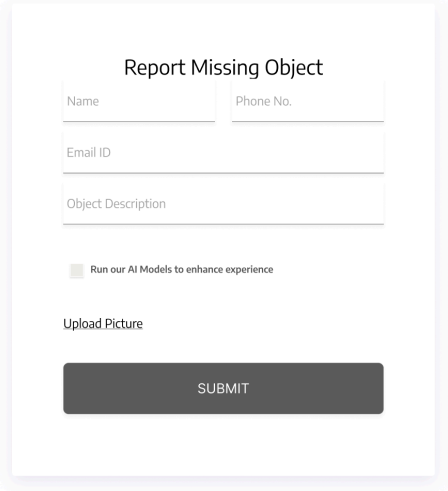
- **Browse Items:** Advanced filters for category, price range, and item condition, ratings, colors or specific keywords/labels can be applied.





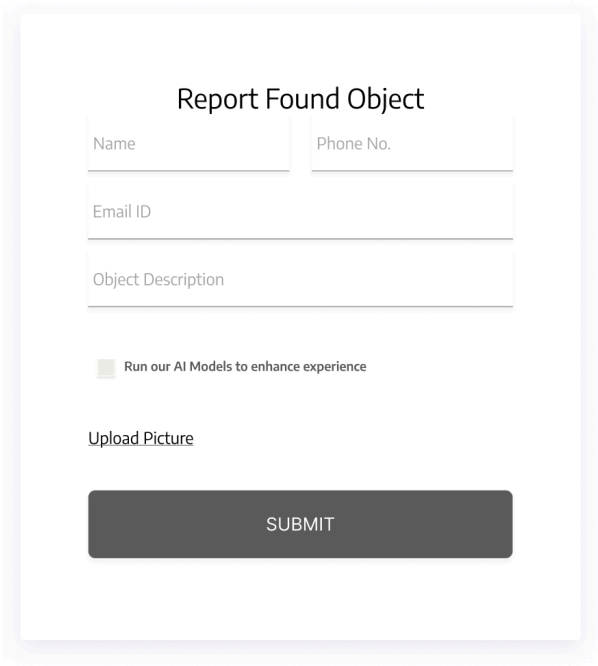
4. Report Missing Object Page:

The "Report Lost Item" page allows users to input important details about their lost item. Users can enter the name, category, description, date, and location where the item was lost. Additionally, they can upload a photo of the item to help identify it. This information is then stored and categorized within the system to make it easier to match with found items. A smart matching algorithm will assist in pairing lost items with any potential matches.



5. Report Found Item Page:

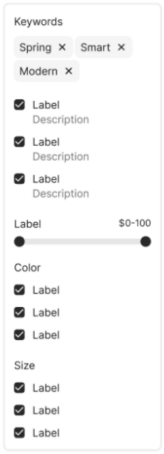
The "Report Found Item" page is similar to the "Report Lost Item" page, with fields for entering the item's name, category, description, date, and location. However, it also includes an extra field for users to specify the condition of the found item. Users can upload a photo of the found item, which will be used to help others identify it. This data is processed and stored in the system to facilitate matching with any lost items.

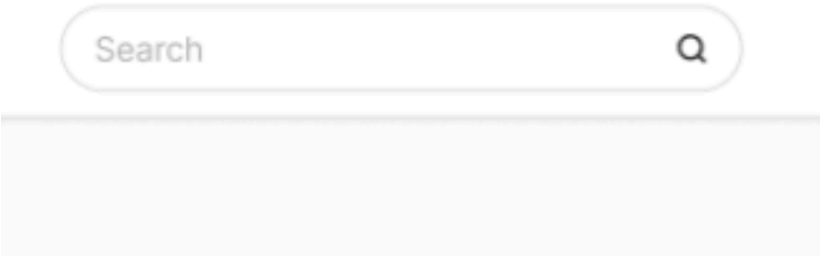


6. Search and Filter Page:

The “Search and Filter Page” is designed to help users easily locate lost or found items and marketplace listings. It features a search bar where users can enter keywords to narrow down their search.

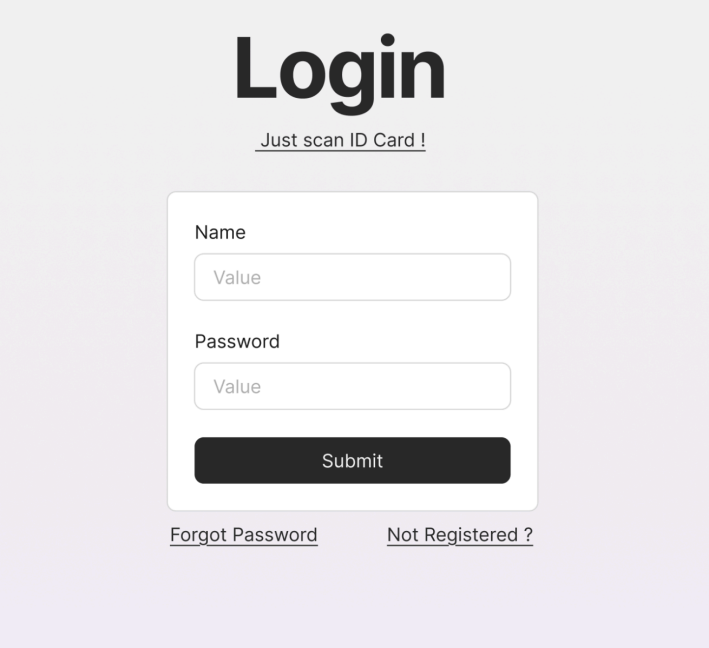
Additionally, various filters are available, allowing users to refine results based on item categories, price range, condition, location, and date ranges. These tools ensure a more targeted and efficient browsing experience.

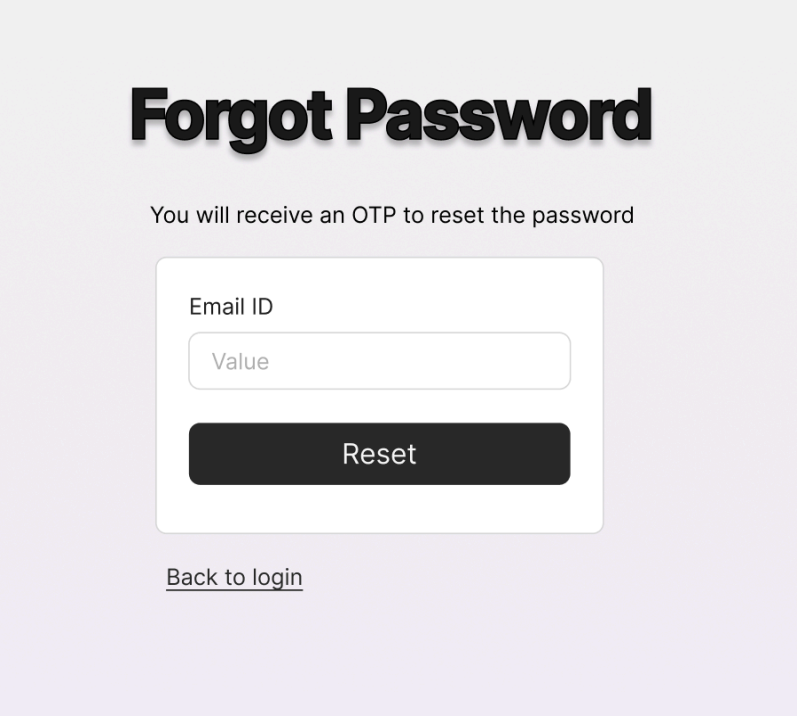




7. Login Page:

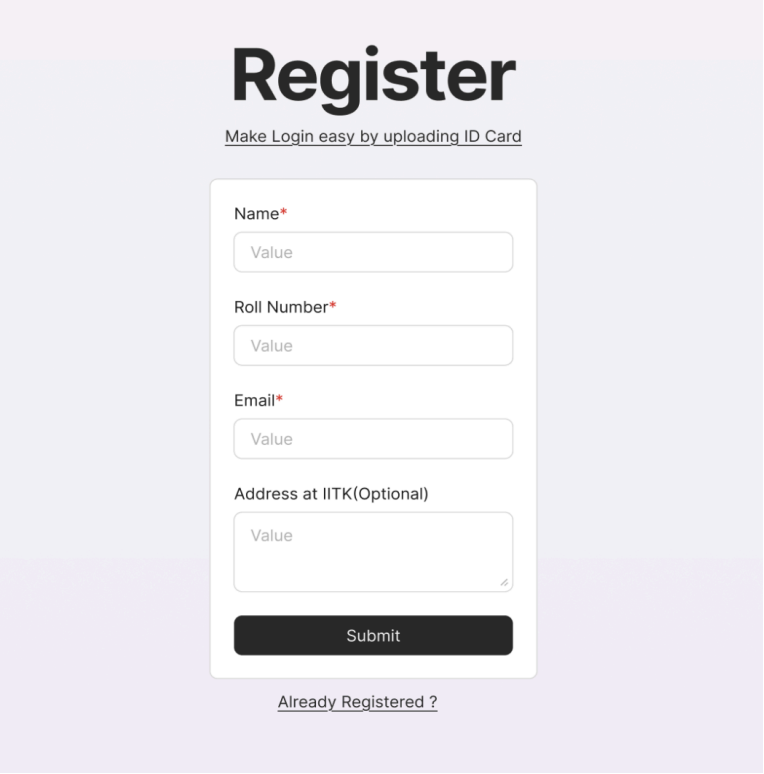
On the login page, users are prompted to enter their registered email address or phone number along with their password. If their credentials are correct, they are redirected to the home page. If the login details are incorrect, users will be prompted to re-enter them.
Additionally the page includes a “Forgot Password” option allowing users to recover their account through a recovery mail or an OTP.





8. Sign-Up Page:

New users need to provide basic information, including their name, email address, phone number and a password, to create an account. To prevent spam and ensure the security of the platform, users will also be required to complete a CAPTCHA verification. Once registered, they will have access to the full suite of DealSimplified features, such as reporting lost or found items, managing marketplace, listings and more.



9.Notifications Page:

The platform will provide real-time updates for users, notifying them of newly reported lost or found items, as well as fresh marketplace listings. For marketplace listings, notifications will be sent when there are new responses or messages from interested buyers. Similarly, for lost-and-found claims, users will be notified if someone provides additional information or responds to their reported item. These timely notifications ensure that users stay informed about the status of their items and can take prompt action when necessary.

3.1.2 Hardware Interfaces

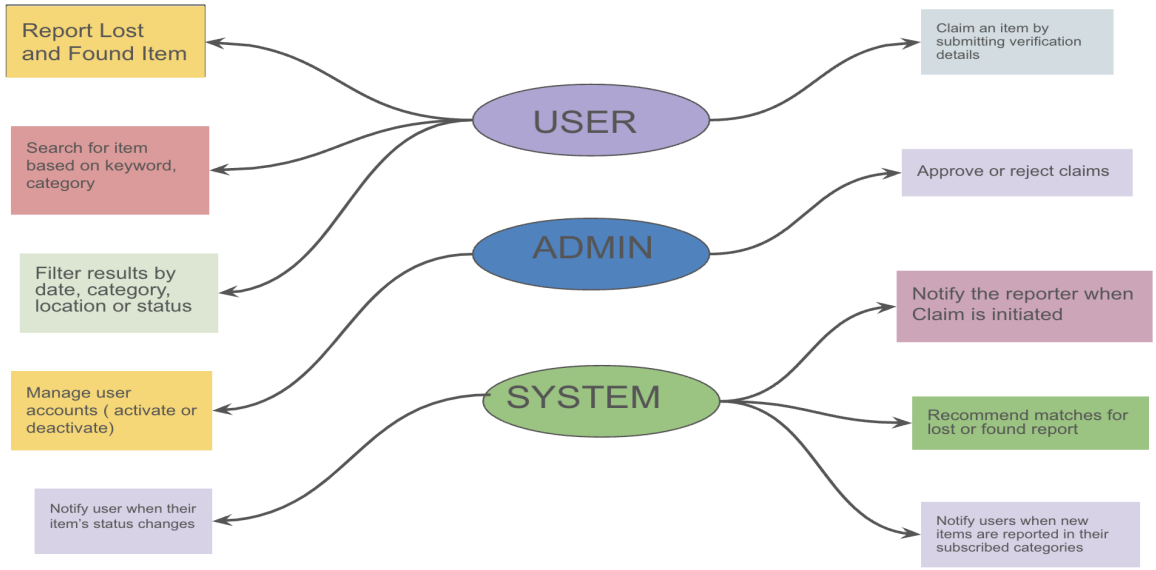
We shall be hosting the web app and working on the databases using a cloud platform. The users will be able to access the web app through their laptops, PC, mobiles etc.

3.1.3 Software Interfaces

• *The server side components including the database will be hosted in a Linux-based environment*
• *The client side components must be functional on modern web browsers such as Google Chrome, Safari, Mozilla Firefox etc.*
• *The database management system will be PostgreSQL.*

3.2 Functional Requirements

LOST AND FOUND:



LOST AND FOUND

User Registration and Login

3.2.1 **F1:** Students register using their official IITK email ID.

```
user_id = register_user(email, password,id_card_image)
```

3.2.2 **F2:** Login functionality for students.

```
session_token = login_user(email, password)
```

3.2.3 **F3:** Password recovery for users.

```
status = recover_password(email)
```

3.2.4 **F4:** Role-based access for admin and users.

```
role_assigned = assign_role(user_id, role)
```

Item Reporting

3.2.5 **F5:** Students can report a lost item.

```
item_id = report_lost_item(user_id, description, category, image
          location, date)
```

3.2.6 **F6:** Students can report a found item.

```
item_id = report_found_item(user_id, description, category, image,
          location, date)
```

Search and Filter

3.2.7 **F7:** Search for items using keywords or item_id

```
search_results = search_items(keyword,item_id)
```

3.2.8 **F8:** Filter items by category, location, date, or status.

```
filtered_results = filter_items(category, location, date, status)
```

Claim Management

3.2.9 **F9:** Users claim an item by submitting proof of ownership.

```
claim_id = submit_claim(user_id, item_id, verification_details)
```

3.2.10 **F10:** Notify reporters when a claim is initiated.

```
notification_sent = notify_reporter(reporter_id, claim_id)
```

3.2.11 **F11:** Admin validates or rejects claims.

```
claim_status = validate_claim(admin_id, claim_id, status)
```

Category Management

3.2.12 **F12:** Use predefined categories for reporting and searching items.

```
categories = get_categories()
```

3.2.13 **F13:** Admin can add or update categories.

```
update_status = manage_categories(admin_id, category, action)
```

Admin Panel

3.2.14 **F14:** Admin manages user accounts (activate, deactivate).
```
account_status = manage_user_account(admin_id, user_id, action)
```

3.2.15 **F15:** Admin resolves claim disputes.
```
dispute_resolution = resolve_dispute(admin_id, claim_id)
```

Item Tracking and Status Updates

3.2.16 **F16:** Track the status of reported lost items.
```
lost_status = track_lost_status(item_id)
```

3.2.17 **F17:** Track the status of reported found items.
```
found_status = track_found_status(item_id)
```

3.2.18 **F18:** Update item status upon successful return.
```
status_updated = update_item_status(item_id, status)
```

Matching System

3.2.19 **F19:** Recommend matches for lost and found items using NLP and CV.
```
matches = recommend_matches(item_id)
```

Notifications

3.2.20 **F20:** Notify users of potential matches for their lost items.
```
notification_sent = notify_match(user_id, match_id)
```

3.2.21 **F21:** Notify users when someone claims a found item they reported.
```
notification_sent = notify_claim(user_id, item_id)
```

3.2.22 **F22:** Notify users about status changes in their items.
```
notification_sent = notify_status_change(user_id, item_id,
new_status)
```

3.2.23 **F23:** Allow users to subscribe to specific categories or keywords for notifications.
```
subscription_id = subscribe_notifications(user_id, category,
keyword)
```

Community Features

3.2.24 **F24:** Enable commenting for easier communication between reporters and claimants.

```
comment_id = add_comment(item_id, user_id, comment)
```

3.2.25 **F25:** Enable chat for reporters and claimants.

```
chat_session_id = start_chat(reporter_id, claimant_id)
```

MARKETPLACE

Listing Management

3.2.26 **F26:** Create a listing for selling an item.

```
listing_id = create_listing(user_id, description, category,
image, price, condition)
```

3.2.27 **F27:** Edit or update a listing

```
status = update_listing(user_id, listing_id, description,
category, image, price, condition)
```

3.2.28 **F28:** Delete a listing

```
status = delete_listing(user_id, listing_id)
```

Searching and Filtering

3.2.29 **F29:** Search for items in the marketplace.

```
search_results = search_marketplace(keyword, listing_id)
```

3.2.30 **F30:** Filter marketplace items.

```
filtered_results = filter_marketplace(category, price_range,
condition)
```

Messaging and Notifications

3.2.31 **F31:** Enable chat for sellers and buyers.

```
chat_session_id = start_chat(seller_id, buyer_id)
```

3.2.32 **F32:** Notify buyers of matching items.

```
notification_sent = notify_marketplace_match(user_id, listing_id)
```

Wishlist Management

3.2.33 **F33:** Add an item to the buyer's wishlist.

```
wishlist_id = add_to_wishlist(user_id, listing_id)
```

3.2.34 **F34:** Notify buyers when a wished item is listed.

```
notification_sent = notify_wishlist_match(user_id, listing_id)
```

Bidding and Trade

3.2.35 **F35:** Place a bid on an item.

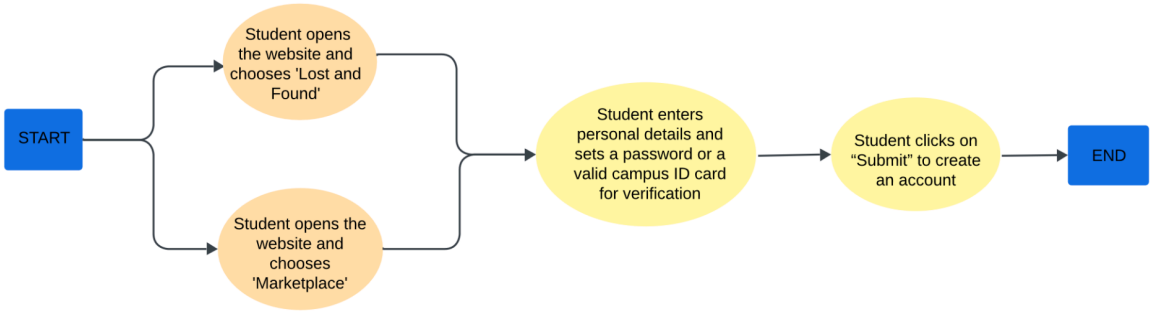
```
bid_id = place_bid(buyer_id, listing_id, bid_amount)
```

3.2.36 **F36:** Initiate a trade for an item.

```
trade_id = initiate_trade(user_id, listing_id, proposed_item_id)
```

3.3 Use Case Model

3.3.1 Use Case #1 (Student Registration(U1))



Author – Ritul

Purpose - To register students on platform to be able to use Lost & Found and Marketplace facilities

Requirements Traceability – IITK email Id, IITK id card, user defined password

Priority - High

Preconditions - User must be an IITK student

Post conditions - Student is registered on the platform

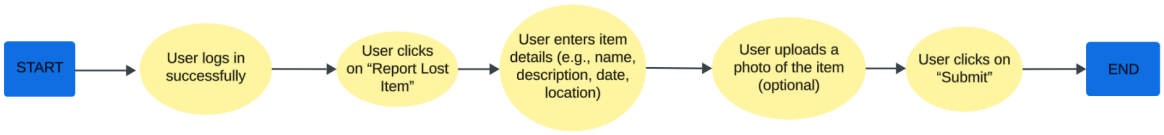
Actors – Campus Students

Exceptions - None

Includes - None

Notes/Issues - None

3.3.2 Use Case #2 (Report Lost Item (U2))



Author – Ritul

Purpose -To allow users to report lost items.
Requirements Traceability – user_id, description of lost item, potential location where item was lost, estimated time when the item was lost, item category, picture of item(optional)

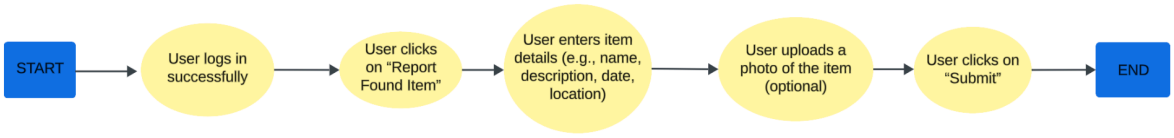
Priority - High
Preconditions - Item must be lost within the campus boundaries
Post conditions - Lost item is successfully reported

Actors – user
Exceptions - None

Includes - U1

Notes/Issues - None

3.3.3 Use Case #3 (Report Found Item (U3))



Author – Kanika

Purpose - To enable the user to report a found item

Requirements Traceability – user_id, description of found item, location where item was found, time when the item was found, item category, picture of item

Priority - High

Preconditions - Item must have been found within the campus boundaries
Post conditions - Found item is successfully reported

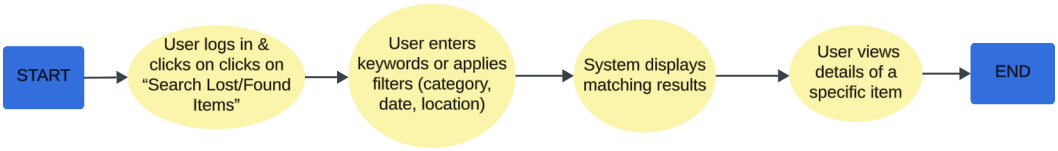
Actors – User

Exceptions - None

Includes - U1

Notes/Issues - None

3.3.4 Use Case #4 (Search Lost & Found Items (U4))



Author – Kanika

Purpose - To help users quickly find lost or found items using keywords or filters

Requirements Traceability – a relevant keyword or item_id

Priority - Medium

Preconditions - The item_id must be valid and accessible

Post conditions - The system successfully displays all the matching results

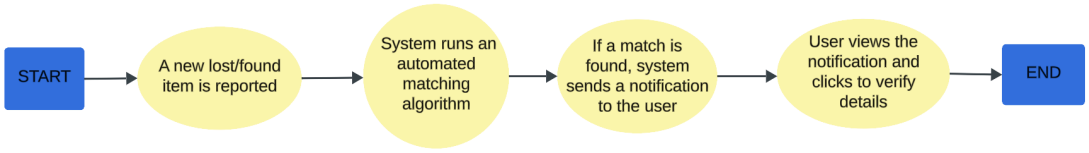
Actors – System

Exceptions - None

Includes - U1

Notes/Issues - None

3.3.5Use Case #5 (Automated Matching Notification (U5))



Author – Riya

Purpose - To inform a user about a potential match between their lost item and a reported found item.
Requirements Traceability – None

Priority - Medium

Preconditions - There should be a minimum of 75% match between the two items

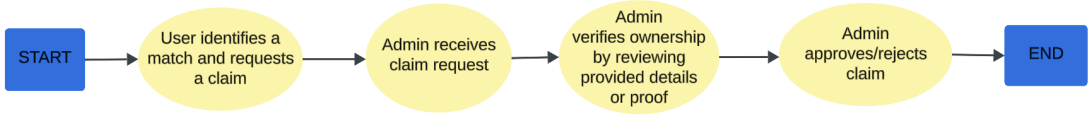
Post conditions - user will be notified successfully about potential matches

Actors – System
Exceptions - None

Includes - U2 or U3

Notes/Issues - None

3.3.6 Use Case #6 (Admin Approval for Claim (U6))



Author – Esra

Purpose - Validate and approve/reject claims for lost and found items with admin oversight.

Requirements Traceability – user_id, name of the item, details of the item for verification, admin_id, reporter_id

Priority - High

Preconditions - Users must be logged in and have submitted a claim with ownership proof.

Post conditions - Claim is either approved, and the item is returned, or rejected with notification.

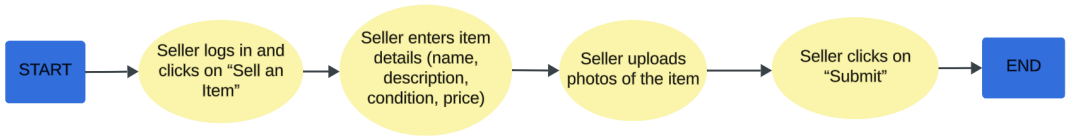
Actors: User and Admin.

Exceptions - None

Includes - U5

Notes/Issues - None

3.3.7 Use Case #7 (Listing an item for sale(U7))



Author – Riya

Purpose - To enable sellers to list an item for sale

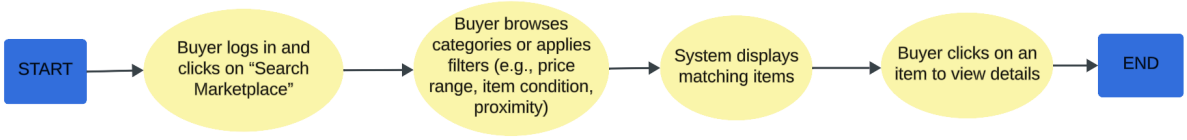
Requirements Traceability – name of item, description of the item ,condition, price of an item, image of the item
Priority - High
Preconditions - The seller should upload a current, accurate picture of the item
Post conditions - The item is successfully listed for selling

Actors – Seller
Exceptions - None

Includes - U1

Notes/Issues - None

3.3.7 Use Case #8 (Search for Items to Buy(U8))



Author – Esra

Purpose - To enable buyers to buy an item

Requirements Traceability – user_id

Priority - High

Preconditions - The user must be logged in
Post conditions - The user got desired search results

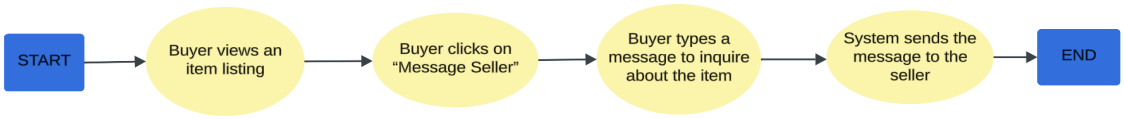
Actors – Buyer

Exceptions - None

Includes - U1

Notes/Issues - None

3.3.9 Use Case #9 (Message a seller(U9))



Author – Riya

Purpose - To allow users to contact a seller for inquiries about an item

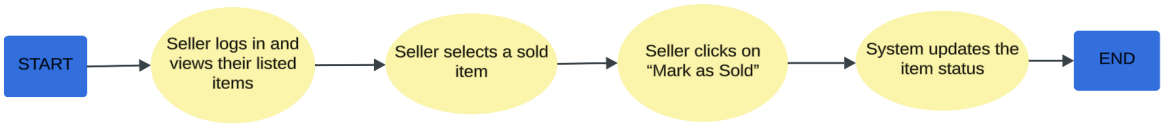
Requirements Traceability – user_id of the buyer, user_id of the seller, item_id
Priority -Medium
Preconditions - The item_id must be valid and accessible
Post conditions - User's message is successfully sent

Actors – Buyer
Exceptions - None

Includes - U7

Notes/Issues - None

3.3.10 Use Case #10 Mark Item as Sold (U10))



Author – Ritul
Purpose - To allow a seller to mark an item as sold, updating its status in the system.

Requirements Traceability – user_id of the seller, item_id.
Priority - Medium
Preconditions - The item_id must be valid and belong to the logged-in seller.
Post conditions - The item's status is successfully updated to "Sold".

Actors – Seller
Exceptions - None

Includes - U7

Notes/Issues - Non

4 Other Non-functional Requirements

4.1 Performance Requirements

1. ***Intelligent Matching Algorithm:***
    - *The feature must implement an intelligent algorithm to match lost and found item descriptions based on keywords, metadata (e.g., location, time), and optional images.*
    - *Matches should be generated within 2 seconds of query submission.*
2. ***Efficient Search Indexing:***
    - *The system must support efficient indexing for search operations, ensuring fast retrieval of lost and found item records even with a large dataset.*
3. ***Document Upload Size Limit:***
    - *Images of lost or found items should not exceed 25MB to comply with the existing size limit.*
4. ***Cross-Platform Compatibility:***
    - *The Lost & Found feature should work seamlessly across all supported platforms (mobile and PC).*

4.2 Safety and Security Requirements

Safety Requirements:

1. ***Confidentiality:***
    - *Users must be able to control which contact details should be shared when a match is identified.*
    - *Anonymous in-platform communication should be enabled for initial contact.*
2. ***Data Protection:***
    - *Information about lost/found items must only be accessible to authorized users (e.g., the item finder and owner).*
3. ***Verification Mechanism:***
    - *The system should implement a verification step where both parties confirm item details before exchanging sensitive information.*

4 ***Mandatory Login****:*

Upon first access of the software, it is mandatory for the user to create his/her unique user ID and password so as to maintain security with respect to the data he/she will be uploading henceforth.

Security Requirements:

1. ***Integrity:***
    - *Ensure the matching algorithm is secure and prevents false positives or mismatched results by using robust input validation and encryption of sensitive data.*

2. ***Availability:***
    - *The feature must be accessible 24/7 for users to report lost and found items without downtime.*

4.3 Software Quality Attributes

4.3.1. Reliability

- *The Lost & Found feature should be rigorously tested to ensure accurate matching and minimal errors.*
- *Continuous uptime and regular maintenance must be ensured to support real-time matching.*

4.3.2. Performance

- *The matching algorithm must retrieve results in ≤2 seconds.*
- *Optimize database queries to handle simultaneous requests from multiple users.*

4.3.3. Interoperability

- *Integrate seamlessly with existing modules such as user authentication and document upload.*
- *Allow compatibility with third-party APIs for enhanced matching (e.g., visual similarity for image matching).*

4.3.4. Usability

- ***For Finders:***
    - *A simple form to upload found item details, including type, description, and image.*
- ***For Owners:***
    - *An easy-to-use interface to input lost item specifications and browse matches.*

4.3.5. Testability

- *Create detailed test scenarios to evaluate:*
    - ***Matching Accuracy:*** *Simulate reports of similar and dissimilar items.*
    - ***Search Performance:*** *Stress-test the system with a high volume of concurrent searches.*
    - ***Data Protection:*** *Verify encryption and secure sharing mechanisms.*

5 Other Requirements

Database Requirements:

- *The database should be designed with security in mind and should protect sensitive data, such as user contact information, item descriptions, and any uploaded images of lost or found items.*
- *The database should support advanced search and filtering capabilities to efficiently match lost and found items based on user-provided specifications.*
- *The database should be able to handle concurrent requests from multiple users, such as uploading lost or found items, searching for matches, and contacting matched users.*
- *Regular backups should be scheduled to ensure data recovery in case of errors or failures, maintaining user trust and system integrity.*

Internationalization Requirements:

- ***Character Encoding:***
  Use UTF-8 or UTF-16 encoding to support diverse characters in item names, descriptions, and user messages, accommodating multiple languages if necessary.
- ***Language-Independent Code:***
  Ensure that all text displayed in the app or website, including field labels, notifications, and messages, is externalized to resource files for easier localization or translation.
- ***Keyboard Input:***
  Allow users to input specifications (e.g., keywords like “black bottle with logo” or “left AirPod”) using various keyboard layouts. Ensure that input fields handle special characters and symbols gracefully.

Legal Requirements:

- *A mechanism for preserving uploaded data (e.g., lost or found items) should be implemented. For instance, lost and found item records may need to expire or archive after a specified time.*
- *Explicit consent should be obtained from users before sharing contact information with a matched user. Users must also be informed about how their data will be stored and used in the matching process.*
- *The application should implement measures to verify the authenticity of users and prevent misuse (e.g., uploading fraudulent found items or false claims).*
- *Provide a disclaimer or terms of service stating that the platform only facilitates connections between users and does not assume responsibility for the validity of claims or ownership disputes*

Appendix A – Data Dictionary

Listing Class:

Element Name	*Description*	*Attributes*	*Operations*
Listing	Represents an item listed for sale or trade by a user.	listing_id=string user_id=string description=string category=string image=string price=float condition=string	create_listing() update_listing() delete_listing()

User Class:

Element Name	*Description*	*Attributes*	*Operations*
User	Represents a platform user who can perform actions like creating listings and managing claims.	user_id=string email=string password=string name=string phone_number=string	register_user() login() update_user_details() reset_password() delete_account()

Admin Class:

Element Name	*Description*	*Attributes*	*Operations*
Admin	Admin responsible for overseeing user activities and approving claims.	admin_id=string email=string password=string	approve_claim() reject_claim() review_listing() suspend_user_account()

Notification Class:

Element Name	*Description*	*Attributes*	*Operations*
Notification	Represents notifications sent to users about automated matches or updates.	notification_id=string user_id=string message=string timestamp= datetime	send_notification() mark_as_read()

Appendix B - Group Log

• Offline meetings were hosted on a regular basis to discuss the software ideas, consistently monitor the completion of tasks assigned to every team member.
• Apart from this, a WhatsApp Group was also created to discuss daily updates over the documentation, along with brainstorming over some sudden change of thoughts.

Date, Time & Duration	*Meet Type*	*Ongoing Discussions in the meet*
12/01/2025	*Offline Meet*	*Discussion over designing the website/App upon chosen idea along with finalizing the computer stack for development of the website.*
15/01/2025	*Offline Meet*	*Discussion upon chosen idea and deciding on the technology stack to be used for the software's frontend and backend development.Concluding which we finalized to use Angular.js as frontend and Django for backend of our website.*
19/01/2025	*Online Meet*	*Meeting regarding work distribution for SRS documentation .The documentation was divided into four subparts , with 2-3 members assigned to each subpart for completion .It was also decided that all members would stay updated on the progress and content of each other's assigned subparts.*
21/01/2025	*Offline Meet(with TA)*	*Meeting with TA to give him updates on the progress of the documentation and discuss the core idea of "DealSimplified" .Additional inputs and suggestions for modifications were also gathered during the discussion.*
24/01/2025	*Online Meet*	*Final meet amongst the team members as a finalized draft of SRS documentation was ready from team's side for the submission by the due date.*