LostLink: Web-Based Lost and Found System in Universiti Teknologi PETRONAS (UTP)

by

Siti Nurul Izzah Binti Saharudin

21000991

Dissertation submitted in partial fulfillment of

The requirements for the

Bachelor of Technology (HONS)

(Information Technology)

SEPTEMBER 2025

Universiti Teknologi PETRONAS

32610, Bandar Seri Iskandar

Perak Darul Ridzuan

CERTIFICATION OF APPROVAL

LostLink: Web-Based Lost and Found System in Universiti Teknologi PETRONAS (UTP)

By

Siti Nurul Izzah Binti Saharudin

21000991

A project dissertation submitted to the

Information Technology Programme

Universiti Teknologi PETRONAS

In partial fulfilment of the requirement for the

BACHELOR OF INFORMATION TECHNOLOGY (Hons)

Approved by,

(Dr. Siti Nurlaili Binti Karim)

UNIVERSITI TEKNOLOGI PETRONAS

BANDAR SERI ISKANDAR, PERAK

September 2025

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is on my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or person.

SITI NURUL IZZAH BINTI SAHARUDIN

ABSTRACT

Many students and staff in Universiti Teknologi PETRONAS (UTP) have a common issue of losing their belongings. Inefficiency and unresolved cases have resulted from unstructured lost and found items reporting such as WhatsApp and Telegram platform and asking anyone nearby. A centralized web-based platform called LostLink makes it easier for students and staff to report, search, and retrieve misplaced objects. This project offers features including admin role, claim verification, photo submission, and item description in an effort to save time and reduce stress of individuals for their belongings. Enhances campus connectivity and services through LostLink support SDG 11: Sustainable Cities and Communities and encourages improved communication. With this, this will develop harmony within the community of University Teknologi PETRONAS (UTP).

This report covers the background, problem statement, survey conducted to show feedback from users in UTP community, literature review, methodology that highlights the approach that will be used and tools/technologies to develop LostLink and the flowchart that shows the flow of the system from many aspects, and conclusion along with future work that will happen shortly after this.

After concluding this report, it sums up the plan of the system, LostLink with its goals to address the issue of ineffective and unstructured lost and found item management.

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to my supervisor, Dr. Siti Nurlaili, for their continuous guidance, support, and constructive feedback throughout the planning and development of this project.

I would also like to express my gratitude to the lecturer and staff of Universiti Teknologi PETRONAS for proving knowledge, facilities, and educational setting that made this effort possible. Ever since, I entered Universiti Teknologi PETRONAS (UTP) since foundation studies back in 2021 until my bachelor studies that will end in 2025, I was able to gain many knowledge and skills for these years that able to enhance myself. I was able to gain many positive values that helped me build up to become a stronger, responsible, and better person especially as an adult.

I am deeply grateful to my family and friends for their patience, unwavering support, countless motivation and encouragement throughout this journey. This initiative would not have been feasible without their support.

Finally, I would like to acknowledge my own dedication and perseverance throughout this journey. It took so much effort, self-control, perseverance, and persistent effort to finish this dissertation, and I am pleased with the development and improvement I made along the way.

TABLE OF CONTENTS

CERTIFICATION OF ORIGINALITY iii
ABSTRACT iv
ACKNOWLEDGEMENT v
CHAPTER 1: INTRODUCTION 1
1.1 Background 1
1.2 Problem Statement 1
1.3 Objectives and Scope of Study 2
1.4 Target Users & Platform 3
CHAPTER 2: LITERATURE REVIEW 5
2.1 Overview 5
2.2 Existing Systems and related Studies 5
2.3 Comparative Table 10
2.3.1 Database and Development Frameworks 10
2.3.2 System Function and Features 11
2.3.3 Verification and Security Model 12
2.3.4 Usability and Accessibility 13
2.3.5 Privacy and Data Protection 14
2.4 Theoretical Background 15
2.3.1 System Development Life Cycle (SDLC). 15
2.3.2 Human-Computer Interaction (HCI) and Usability. 16
2.3.3 Authentication and Data Security Concepts. 16
2.3.4 Relational Database Optimization. 16
2.5 Organizational Themes 17
2.4.1 Usability and Accessibility of Platform Design 17
2.4.2 Traceability and Item Retrieval Efficiency 18
2.4.3 Security and Ownership Verification Model 19
2.6 Summary 20
CHAPTER 3: METHODOLOGY 23
3.1 Overview of Methodology 23
3.2 User Requirements 23
3.3 Survey 25

3.3 Agile Approach 25
3.4 Tools/equipment required 31
3.4.1 Platform: Visual Studio Code (VS Code) 32
3.4.2 Programming Language: Python 33
3.4.3 Backend: Python & Framework: Flask 33
3.4.4 Frontend: HTM, CSS, JavaScript 33
3.4.5 Templating Engine: Jinja2 34
3.4.6 Database: MySQL 34
3.4.7 Browser: Google Chrome, Microsoft Edge 35
3.4.8 Deployment (Development): Localhost Server 35
3.4.9 Deployment (Future): Render/Railway/Cloud MySQL 36
3.5 System Modelling and Design 37
3.5.1 Use Case Diagram 37
3.5.2 Flowchart 39
..... 39
..... 41
3.5.3 Entity-Relationship Diagram (ERD) 42
3.5.4 System Architecture Diagram 44
3.5.5 Activity Diagram 46
3.7 Summary 47
CHAPTER 4: RESULT AND DISCUSSION 48
4.1 Overview of Result and Discussion 48
4.2 System Implementation (Backend & Frontend) 48
4.2.1 Backend (Flask): 49
4.2.2 Frontend (HTLM, CSS, and JavaScript) 51
4.2.3 Database Implementation (MySQL) 52
4.2.4 Final System Flowchart 56
4.2 System Interface and Features 61
4.3 Functional Testing and Bug Analysis 66
4.3.1 Functional Testing Overview 66
4.3.2 Test Environment 66
4.3.3 Functional Testing 67
4.3.4 Bug Analysis 69

4.4 System Test Summary 71
4.5 Usability and Feedback Evaluation 71
4.6 User Testing Results and Analysis 72
4.7 Performance Evaluation 78
4.7.1 Test Environment 78
4.7.2 What to measure 78
4.7.3 Performance Result 79
4.7.4 Discussion 80
4. 8 Challenges/Limitations 81
4.9 Summary 81
CHAPTER 5: CONCLUSION AND RECOMMENDATION 83
5.1 Conclusion 83
5.2 Recommendations 84
REFERENCES 85
APPENDICES 87

LIST OF FIGURES

Figure 3.1: Visual Studio Code (VS Code) Icon……..……31

Figure 3.2: Python Icon……...……...32

Figure 3.3: HTML, CSS< JavaScript Icon…….33

Figure 3.4: Jinja2 Icon……...….34

Figure 3.5: User Activity Diagram (Use Case Diagram) ……...……36

Figure 3.6: Reporting Process Flowchart……37

Figure 3.7: Claiming Process Flowchart…….……38

Figure 3.8: Admin Verification & Returning Flowchart…….……39

Figure 3.9: View Reports Flowchart……...……40

Figure 3.10: Remove Reports Flowchart……...……41

Figure 3.11: ERD of LostLink System (Before Implementation) ……42

Figure 3.12: System Architecture Diagram……...……43

Figure 3.13: Activity Diagram……...…45

Figure 4.1: LostLink file structure in Visual Studio Code……55

Figure 4.2: Exmaple coding of /report_item route……57

Figure 4.3: Example of /admin/dashaboard_route……58

Figure 4.4: Example JavaScript Code……...…..…60

Figure 4.5: HTML connection to script.js……..…61

Figure 4.6: ERD in Databse Schema (After Implementation) …….……63

Figure 4.7: Flowchart of Normal User Workflow……67

Figure 4.8: Flowchart of Admin Workflow……...……70

Figure 4.9: Login Page…….……74

Figure 4.10: Registration Page…….……75

Figure 4.11: Home Page……76

Figure 4.12: Report Lost Item Page……...……77

Figure 4.13: Report Found Item Page……...……78

Figure 4.14: Search Items Page……79

Figure 4.15: Claim Page……80

Figure 4.16: Admin Dashboard……

LIST OF TABLES

Table 2.1: Comparative: Database & Development Frameworks……10

Table 2.2: Comparative: System Functions & Features……...11

Table 2.3: Comparative: Verification & Security Model…….12

Table 2.4: Comparative: Usability & Accessibility…….12

Table 2.5: Comparative: Privacy & Data Protection……13

Table 3.1: Functional Requirements…….22

Table 3.2: Non-Functional Requirements…….23

Table 3.3: Tools and Technologies Used……...29

Table 3.4: ERD Relationship Description……42

Table 4.1: ERD Structure of LostLink (Attributes Summary) …….64

Table 4.2: Table Names, Primary Keys & Descriptions……65

Table 4.3: Entity Relationships……...66

Table 4.4: System Interfaces & Description……...73

Table 4.5: Test Environment……...85

Table 4.6: Functional Testing…… ……87

Table 4.7: Bug Analysis…….89

Table 4.8: Usability Evaluation.…….94

Table 4.9: Performance Test Environment……...98

Table 4.10: Metric Description & Tool/Method……...99

Table 4.11: Performance Results……..…….100

CHAPTER 1

1.1 Background

At Universiti Teknologi PETRONAS (UTP), students and staff often misplace their personal items across the campus. Currently, students and staff only rely on informal platform such as Whatsapp and Telegram platform and asking around if they have seen the items. However, this method is significantly inefficient.

With the availability of modern technology in today's life, a web-based platform that is both centralized and searchable provides a more dependable source for these issues of lost and found items. In addition, such a system has the potential to support the operational workflow of UTP security and student support units, who are responsible in managing reports of misplaced items. The platform can enhance administrative accountability and user experience by implementing structured reporting, verification, and traceability.

1.2 Problem Statement

There are many cases where individuals aren't able to retrieve their belongings as it could not reach to the person who found the item. The community of UTP relies on social media platform which in the end, it bought difficulties to monitor and lacking traceability. This leads to significantly time-consuming and increase in stress in individuals as their belongings is what they need the most or a treasure to them.

Next, when there is no centralized system for managing lost and found issue, it makes it difficult to track or retrieve an item as the information is being scattered around which makes it difficult for people to keep track of updates or verify item ownership. By this, this may lead the item to become unclaimed as time passed by as the ownership failed to find the person who founds it or may have given up finding their item.

Lastly, the lack of awareness and user engagement made the issue worse as students and staff are low in motivation to report found items if there is no reliable

and easy to-use system in one place. Need for trustworthy platform that offers a secure verification and builds a sense of trust among users that promotes transparency and accountability in item lost management, fostering a responsible behaviour in an individual who finds the item.

1.3 Objectives and Scope of Study

The objectives are crucial as it act as the goals of the project. One of the objectives under LostLink is to design and develop a user-friendly web-based system that allow the users to report lost or found items. This system of LostLink allows the users to a list of reported items and use keywords search or filtering functions to locate items efficiently. For example, when the user wants to find their lost watch, they can search watch instead to going through many lists of reported items. With this, this will ensure that the user can easily use the system which leads to an increase of effectiveness.

Next objective is to enhance the usability, transparency, and traceability of lost item management through feedback and system tracking data. In order to maintain the platform's usability's and responsiveness to user demands, feedback will be conducted to gather data on the user's use and will be integrated into ongoing functional and interface improvements.

Other than that, is to evaluate the system based on its usability, transparency, and traceability, using both system testing and feedback from real users. This assessment will help determine the platform's effectiveness is achieving its goals and provide insights for further refinement in the next phase of the project.

The final objective is to encourage responsible and claiming behaviour through a structured platform that fosters trust and accountability among users. By providing features such as admin role, security question as a method to secure verification, this able to build trust and accountability.

The scope of this project focuses on addressing the issue with lost and found item management within the campus of Universiti Teknologi PETRONAS (UTP).

In the absence of a centralized platform for this issue, the community become lost and faced a challenge which is the struggle to retrieve their belongings or the finder unable to find the rightful owner of the item. Therefore, the system of Lostlink is designed to become a solution for this issue.

1.4 Target Users & Platform

Target Users

The target users that will be used for this project is the people of UTP as they represent the individuals that were affected in this problem of unstructured lost and found. Therefore, by targeting these people, this will ensure that the system addresses real, recurring problems with a defined and relevant community.

Platform

The platform for this system will be a web-based application making it easy in accessibility as it requires no installation. The decision to develop website allows to create a minimized compatibility problems and enables greater accessibility across many devices.

Core Function:

- Item reporting with photo uploads. Users able to report the lost item by filling in the detail of the item (description) along with uploading the photo of the item, therefore, when the owner search it, they recognize their items immediately because of the photo. This shows that this feature helps to reduce confusion and increases the accuracy of item recovery.
- Keyword search and filtering functions. User is able to key in the word in search without having to scroll through many lists of reported lost item. Other than that filtering function will be implemented to filters based on category or date. With this, this will save time.
- Claim verification using security questions. Security question is crucial to verify the rightful owner. This is to prevent thief happening in the system where when they see the item they want, they would claim and lie that they

are the rightful owner. 8 Other than that, this will build trust among the user regarding on how LostLink investigate this matter in security.

- Admin approval of claims. The claims will be reviewed by the admin to ensure it is the rightful owner before making approval. This is to ensure fairness, secure, and effective is focused in this system

CHAPTER 2

2.1 Overview

The purpose of this chapter is to evaluate relevant studies that have been made, technologies, and conceptual models related to lost and found management systems, web application usability, and data security. LostLink is a web-based lost and found system that puts focus on its audience on Universiti Teknologi PETRONAS (UTP) community as it lacks efficiency and no centralized platform for it. The literatre review provides both theoretical and empirical foundations that support the development of LostLink intended to improve the administration, verification on the claiming item, and recovery of misplaced things within a university environment. This chapter identifies how technology has been used to address issues with item loss, user reporting, and claim verification by examining previous research and current implementation.

2.2 Existing Systems and related Studies

In recent years, there are numbers of systems that have been developed to assist in the management of lost and found items, particularly in universities, workplaces, and public facilities. The reason for this is because they relied on manual tracking and communication between the finder and the owner by traditional paper-based and social media platform that leads to ineffective. These approaches often resulted in misplaced reports, duplication, and lack of accountability. Therefore, with the increasing use of digital and advanced technology that we have today, many researchers have proposed online systems to make the process faster, more organized, and traceable.

In this section, several reviews on related and relevant studies and implementation to analyze their strengths, limitations, and their relevance to the proposed LostLink system.

My Lost and Found Website

The My Lost and Found Website was developed by Ismail (2009) at Universiti Teknikal Malaysia Melaka (UTeM) and was pointed as the first Malaysian attempts to digitalize the lost and found process. PHP and MySQL were used in the development

of the system, which has a straightforward user interface that allowed users to upload pictures and descriptions of lost or found items. The main functions that are included in this My Lost and Found Website are item posting, searching, and listing based on categories such as electronic, stationery, and documents.

Although it represented an important step in transforming from manual to digital management, the system lacked administrative control and authentication. Without verification, any user might submit or claim an item which can cause false ownership and lead to theft occurring on the website, other than that, it can create potential risks of inaccurate data. To add on, users had to manually review postings as the system lacked automated matching or alerting system. Nonetheless, this study was important because I established the foundation for later Malaysian systems that aimed to enhance database functionality and user interaction.

Online Web-Based Lost and Found System

Bataineh, Bataineh, and Al Kindi (2015) developed an online web-based lost and found system at Zayed University, UAE, which automated manual item reporting and combine a structured database to record and retrieve data efficiently. HTLM and JavaScript were used to create the interface, while PHP and MySQL were used to establish the system architecture. The project was unique as they included a usability assessment that measured the interface's efficacy and efficiency using eye-tracking technology. The study they conducted involved university students who performed pre-defined tasks while their eye movements were tracked to analyze focus, clarity, and readability.

According to the results, users found the interface easy to use with clear menu layout, and minimal confusion as it's kept simple. However, the system lacked privacy control and ownership verification on the item which can lead potential risk of theft in action by falsely claiming the item with bad intention of stealing. Without administrative authorization, any user might access item details or assert ownership. Other than that, the design did not include a feedback mechanism or claim progress tracking. To sum up this study conducted by Bataineh, Bataineh, and Al Kindi (2015), it is clear that the project shows excellent usability but weak data protection by all

means, needs to highlight on a necessity of systems that strike a compromise between security and usability.

Python-Based Lost and Found Website

Tiwari et al. (2019) developed a lost and found platform using Python Django, Bootstrap, and SQLite. Compared to previous PHP-based projects, the system used a more contemporary approach which put deep focus on responsive design, search optimization, and secure login. There are features that allow users to upload pictures, list items, and register. The project aimed to reduce reliance on local databases by utilizing Django's scalability and framework flexibility.

However, the study did not include multi-role in the system which means there is no existence of administrative verification status. Since there were no admin dashboards, spam listings and fraudulent users were not monitored. The database was functional but minimal as it has limited relational indexing and no automated notifications. Despite these drawbacks, Tiwari's research advanced the industry by demonstrating how database-driven, scalable applications might be created using Python-based framework, this approach is comparable to LostLink's Flask MySQL stack, which also emphasizes simplicity and security.

Web- Based Lost and Found System

A more thorough architecture was suggested by Kumar et al. (2022) in their paper of Web-Based Lost and Found System. Their paper was published in the MIT International Journal of Computer Science and Information Technology. Their system featured three core modules which are Citizen, Admin, and Police. With these roles, it was one of the only systems with government-level authentication. Starting off with users registered with email verification and ID data, police served as middlemen to verify found or claimed things, and admins who act behind the system by examined and authorized complaints.

The system design showed high potential for data authenticity and accountability, but it was not tailored for campus or small community use like LostLink that target UTP community The process was lengthy and required several verification stages

before clearance. While enhanced security has had negative impact on efficiency and usability. Additionally, the system's absence of real-time communication function and reliance on admin-police coordination limited user interaction. In comparison, LostLink simplifies the verification process by allowing admin-only review rather than involving external authorities which makes the process faster and more useful for academic settings.

An Effective Lost and Found System in University Campus

The FoundeLost system was created by Tan and Chong (2023) for their university campus, Universiti Kebangsaan Malaysia (UKM) and incorporated web and mobile interfaces for usage on college campuses similar to LostLink. This system of The FoundeLost introduced security question verification to confirm the right ownership of the item and automated item status tracking which are lost, found, pending, returned, or disposed. The backend was built by using PHP and MySQL, and Java was used to create the mobile application.

Usability testing was conducted using the System Usability Scale (SUS), and the results showed that users found the platform easy to use and visually appealing. Improvements in data organization and report generation were also recognized in the study. Nevertheless, FoundeLost did not have a complete administrative management layer as the system relied on automated verification without manual (human intervention) approval. To add on, before a claim was approved, consumers contact information was available, which raised privacy issues. The results emphasize how important it is to have both admin-led verification and automatic verification.

A Comporative Study on Lost and Found Management Systems

Neo et al., (2024) conducted a comparative analysis of lost and found systems used across three academic institutions which are Universiti Tunku Abdul Rahman (UTAR, Malaysia), California Polytechnic University (USA), and Nazareth School (Philippines) who built their lost and found systems compared by Neo et al. (2024). Each platform's features, security, usability, and efficiency were evaluated by the study. Results showed that UTAR's system had the most complete functionality,

offering claim forms, staff accounts, and public announcements. However, none of the reviewed systems employed secure claim verification or privacy protection.

This comparative study contributed valuable insight by highlighting that usability and security are rarely implemented together. Many systems were good at one thing but bad at another. The authors came to the conclusion that an efficient system should have authentication, usability, and accessibility which is a combination central to LostLink's design philosophy.

CampusTrace: A Privacy-Focused Lost and Found System

CampusTrace is a university-based lost and found management system with an emphasis on administrative verification and user privacy that was introduced by Abrahim et al. in 2024. The system decreased false claims and enhanced the validaty of reported objects by requiring each claim submission to pass ad adin validation stage prior to ownership confirmation. The system, which was created with PHP and MySQL, encrypted user information that includes phone numbers and email addresses, ensuring the details hidden until approval. It is also added on automated notification function that informed usre sof changes in the status of their claims, enhancing response times and transparency.

Although the study showed robust admin control and good privacy protection, CampusTrace continues to rely on manual verification which slowed claim processing and lacked usability testing. LostLink bulds upon this foundation by integrating admin validation and privacy encryption with a user-tested interface and semi-automated claim tracking to maintain both security and efficiency.

Modern Lost and Found Platform

Shrivastava et al. (2025) proposed a next-generation platform built using React.js, Node.js, and MongoDB that integrated cloud computing and artificial intelligence concepts. In order to automatically match lost and found items, their system recommended employing picture recognition. Despite being highly innovative, this system was only offered as a conceptual design without any real implementation or user testing.

This study highlights modern technologies but overlooked real-world usability and accessibility, especially particularly for small-scale deployments like universities. LostLink draws inspiration from Shrivastava's emphasis but focuses on practical implementation rather than experimental AI features, ensuring accessibility, speed, and security for its users.

Summary of existing studies

Throughout these studies, the evolution of lost and found systems has progressed from straightforward platforms for listing items to more complex and interactive web-based applications. Functionality was given priority in early systems, while usability testing and partial verification were added in later ones. However, despite technological advances, most systems fail to integrate usability, security, and privacy into one at the same time.

While some put their focus on security and structure, others (Bataineh et al., 2015; Tan & Chong, 2023) focus on usage (Kumar et al., 2022). Few provided a good balance between the two. Furthermore, none of the reviewed systems featured a comprehensive admin module that verifies claims, ensured privacy, and provided transparent status tracking.

Therefore, LostLink was designed and built to fill up these particular gaps. Combining administrative control, verified claim processing, and an ISO-compliant, user-friendly interface. LostLink seeks to offer a safe, dependable, and user-friendly solution for managing lost and found items within academic communities.

2.3 Comparative Table

2.3.1 Database and Development Frameworks

System/Study	Database Type	Framework/Technology	Scalability/Efficiency
Ismail (2009)	MySQL (Local)	PHP	Low • Static • Moderate

Bataineh et al. (2015)	MySQL	PHP, HTML, Javascript	Moderate - Automated reporting but limited indexing
Tiwari et al. (2019)	SQLite	Python Django	High scalability - Modern backend
Kumar et al. (2022)	MySQL	PHP	High but overly complex
Tan & Chong (2023)	MySQL	PHP (Web), Java (Mobile)	Moderate - Hybrid deployment
Abrahim et al. (2024)	MySQL	PHP	Moderate - Admin - Heavy processing
Shrivastava et al. (2025)	MongoDB	React, Node.js	Very high - Cloud-based
LostLink	MySQL	Flask (Python), HTML, CSS, JS	High - Normalized DB - Indexed search - Optimized queries

Table 2.1 Shows Comparative with Existing Studies and LostLink

2.3.2 System Function and Features

System/Study	Main Functionalities	Automation/Notifications	Multi-Role capability
Ismail (2009)	Item posting, search, category listings	None	Single user type
Bataineh et al. (2015)	Automated reporting, usability testing	None	Single user type
Tiwari et al. (2019)	Search optimization, login system	None	None

Kumar et al. (2022)	Citizen, admin, police roles	Manual coordination	Multi-role (3 types)
Tan & Chong (2023)	Security questions, auto item tracking	Automated item status	User only
Abrahim et al. (2024)	Admin verification, status notification	Semi-automated	Admin and User
Shrivastava et al. (2025)	AI image match	Proposed automation	Unspecified
LostLink	Reporting, searching, claim verification, admin role (dashboard)	Yes, for user and admin	Admin and User

Table 2.2 Shows System Function and Features

2.3.3 Verification and Security Model

System/Study	Verification Type	Security Features	Weakness Identified
Ismail (2009)	None	Open submission	False claims possible (theft)
Bataineh et al. (2015)	None	Public access	Weak security
Tiwari et al. (2019)	Basic login only	Secure login	No admin oversight
Kumar et al. (2022)	ID verification and police verification	High	Overly compex
Tan & Chong (2023)	Security questions	Medium	No admin review
Abrahim et al. (2024)	Admin validation and encryption	High	Manual workload

Shrivastava et al. (2025)	Conceptual AI verification	Theoretical	Not implemented
LostLink	Security Question and Admin approval	Login authentication and privacy-safe contact	-

Table 2.3 Shows Verification and Security Model

2.3.4 Usability and Accessibility

System/Study	Usability Testing	Accessibility (Platform)	User Experience Strengths
Ismail (2009)	None	Web only	Simple but limited
Bataineh et al. (2015)	Eye-tracking test	Web only	High clarity and user satisfaction
Tiwari et al. (2019)	None	Web (Django)	Responsive interface
Kumar et al. (2022)	None	Web only	Detailed role workflow
Tan & Chong (2023)	SUS testing	Web and Mobile	High user satisfaction
Abrahim et al. (2024)	None	Web only	Clear structure
Shrivastava et al. (2025)	None	Web/Cloud (proposed)	Conceptual innovation
LostLink	Planned heuristic evaluation	Web only (UTP access)	Simple layout and fast task completion

Table 2.4 Shows Usability and Accessibility

2.3.5 Privacy and Data Protection

System/Study	Privacy Measures	Data Handling	Weaknesses
Ismail (2009)	None	Open access	Data easily misused
Bataineh et al. (2015)	None	Public listing	Theft/fraud possible
Tiwari et al. (2019)	Basic login	Local DB	No encryption
Kumar et al. (2022)	ID & admin validation	Restricted	Overly bureaucratic
Tan & Chong (2023)	Limited: contacts visible	Automated only	Privacy risk
Abrahim et al. (2024)	Encrypted data	Manual admin	Slower claim flow
Shrivastava et al. (2025)	None	Conceptual	None
LostLink	Contact hidden until approval from admin	Secure MySQL and verified login	-

Table 2.5 Shows Privacy and Data Protection

A comparison of current lost and found system shows that while eac earlier study made significant contributions, it also had specific flaws that LostLink aims to address. Previous systems, like My Lost and Found Website (Ismail, 2009) and Online Web-Based Lost and Found System (Bataineh at al., 2015), prioritized user interface simplicity and basic functionality but lacked authentication, privacy, and administrative oversight which resulted in potential risk in data accuracy and false ownership claims. Although secure login and multi-role structures were introduced by mid-stage advancement of technology and digital which is Web-Based Lost and Found System (Kumar et al., 2022) and the Python-Based Lost and Found Website (Tiwari et al., 2019), their usability was hidden by overly complex verification procedures.

Meanwhile, modern systems such as FOundeLost (Tan & Chong, 2023) and CampusTrace (Abrahim et al. 2024) made progress in automated tracking and protection for privacy, however, they still depended heavily on either full, automation or manual verification which leads to inefficiency and user confusion.

In contrast, LostLink integrates some of the strengths of these earlier system into a single, cohesive framework. It integrates a Flas-MySQL relational database that supports efficient indexing scalability, and structured relationship between users, items, and claim, addressing earlier issues of data redundancy. Functionally, LostLink combines reporting, searching, and claim verification into a single, user-friendly dashboard that is backed by semi-automated alerts that provide users with real-time information on the status of their claims. The system's dual-layer verification methodology, which avoids the complexity of multi-agency procedures seen in Kumar et al. (2022), to ensure data authenticity while preserving workflow efficiency by combining administrative permissions with knowledge-based security questions. LostLink adopts ISO 9241011 usability principles and accessible interface as LostLink prioritizes effectiveness, efficiency, and satisfaction through a clear and user-friendly. Tan & Chong (2023) and Bataineh et al. (2025) mention that privacy is needed to ensure user contact information remains hidden until admin validation, which is further strengthened by guaranteeing that user contact information is concealed until admin validation.

To sum up, LostLink offers a balanced solution that has usability and security gap found in earlier systems. With combining relational database optimization, verified claim processing, and privacy, the system is built towards more transparent, reliable, and user-oriented platform specifically suited for campus, UTP.

2.4 Theoretical Background

The development of LostLink is grounded in established frameworks from software engineering, usability, and data management to make sure that the system is efficient, secure, and user-friendly.

2.3.1 System Development Life Cycle (SDLC).

In System Development Life Cycle (SDLC), it provides structured frameworks that divide the process into sequential phases starting from planning, analysis, design,

implementation, testing, and maintenance (Dennis, Wixom, & Tegarden, 2015). This model encourages methodical advancement, ongoing feedback from myself and tested users, and reduced development risk. LostLink follows the SDLC approach to make sure that each stage is well defined and recorded and tested prior to release in order to have outcome of stable and maintainable web application.

2.3.2 Human-Computer Interaction (HCI) and Usability.

The design of functional, efficient, and user-friendly systems is highlighted by HCI principles. According to Nielsen (1994), further identified key usability heuristic such as consistency, visibility of system status, and error prevention. Lost Link applies these principles for responsive interfaces, straightforward navigation, and able to increase user satisfaction. With this, this ensures that both students and administrator can easily perform reporting and verification tasks.

2.3.3 Authentication and Data Security Concepts.

Ensuring the security of user data and authentication processes is important for building trust in web-based systems. According to Ullah, Xiao, and Barker (2019), they highlighted that effective security models should balance protection and usability. Although low protection ca led to potential risk and danger of data breach/leakage, but too complicated security processes may deter user engagement. LostLink adopts a layered approach to security by incorporating user authentication, admin verification for claims, and privacy-safe contact handling, to make sure that everything is secure and transparent interactions users and administrators.

2.3.4 Relational Database Optimization.

Relational databases form the backbone of the system of many web-based systems, especially LostLink, due to their structured, normalized data storage design. According to Sakshi and Sharma (2025), relational database optimization greatly improves data retrieval performance and scalability using methods including normalization, indexing, and query optimization to improve data retrieval performance and scalability. In LostLink, these strategies are used within the MySQL environment to

ensure that the data is accurate, reduce redundancy, and preserve effective user, items, and claims table interaction.

2.5 Organizational Themes

2.4.1 Usability and Accessibility of Platform Design

An effective digital palatform must explain high usability to encourage adoption and ensure consistent engagement. Usability includes features that directly affect user satisfaction, such as accessibility, simplicity of use, and the intuitiveness of interface design (Nielsen, 1994). In this section, it wil cover LostLink's implementation decisions.

Platform Selection

Purely web-based systems eliminate installation hurdles and compatibility difficulties, while specialized cross-platform apps which are web and mobile that are found in one of the studies that is being conducted which is FoundeLost system from Universiti Kebangsaan Malaysia (Tan & Chong, 2023). This makes them universally accessible through browsers on any device withing the university network. This strategy optimizes reach while lowering maintenance needs for a campus-based deployment like Universiti Teknologi PETRONAS (UTP), ensuring immediate availability.

The principles of interface design

Effective interface design focuses on clarity, simplicity, and easy to use (navigation). Studies show that users report higher satisfaction when systems feature organized menus, recognizable icons, and visually balanced layouts (Bataineh et al., 2015). In the early system, there is an increasing number of users feeling annoyance and felt the task being done poorly that leads a fail in success rates because of neglected modern interface standards. These ideas are put into practice by LostLink hrough an organized and structutured dashboard with clearly labeled functions for reporting, viewing, searching, and managing items.

Justification of LostLink

LostLink highlights simplicity and accessibility with focus on a user-friendly web based interface. Simplified login, search filtering, and rapid feedback are examples that follow accepted usabilityty guidelines and give the UTP community a unified and easy to navigate platform.

2.4.2 Traceability and Item Retrieval Efficiency

The effectiveness of a lost and found system depends on users who can retrieve and identify items efficiently. Traceability is the capacity to monitor and verify the state of an item from report to claim to recover the item. Achieving this requires precise data management and optimized search mechanisms.

Although this system was not formally requested by UTP security, the verification, tracking, and activity logging features in LostLink align with the typical responsibilities of campus security or student support units, who often oversee lost and found processes in university environments. If it is used in the future, these functionalities can help administrative workflows by offering responsible and trackable item handling

The Role of Rich Content.

System that allows users to put in the detail of the item in descriptive way such as category, color, brand, location, and uploading photos leads to higher claim success rate (Tan & Chong, 2023). Rich content enhances both the accuracy of item matching and the confidence of claimants, allowing easy visual confirmation by the rightful owner.

Search and Query Mechanisms.

Efficient search and filtering functions are important when it comes to lost or found system as it is to reduce the time spent searching for the item by going through scrolling in list of items. By allowing users to filter listings by category, date, or location, web system that use optimized relational database (Sakshi & Sharma, 2025)

able to improve retrieved speed. LostLink implements a keyword-based search system with connection with MySQL indexing to make sure that this function is fast and has relevant results. This allows for accurate item matching without requiring a lot of browsing.

Critique of Item-Tracing Technology (IoT)

RFID tagging and other IoT-based systems enhance physical traceability, but they depend on pre-tagged items and specialized scanning stations (Kumar et al., 2022). This dependency is impartical for campus. LostLink on the other hand, places a higher focus on a centralized, data-driven reporting process that allow users to digitally manage all lost and found actions without being constrained by physical locations.

Justification for Lost link

LostLink integrates best practice for traceability by combining rich data submission and advanced search filters. To add on, this is to ensure quicker and more dependable item recovery. This dual strategy of detailed reporting for accurate and optimized queries for efficiency makes sure fast, more reliable item recovery and provides a complete digital record for administrative oversight.

2.4.3 Security and Ownership Verification Model

Any lost and found platform's capacity to stop false claims and protect user privacy determines how credible it is. Inadequate verification processes undermine trust and increase the risk of misuse. LostLink uses an internal verification methodology with a focus on multi-layer authentication, accountability, and openness.

Critique of External Mediation

Previous studies involving external intermediaries, such as law enforcement authorities (Kumar et al., 2022) were unsuitable for campus system due to their lengthy reaction times and complicated procedure to follow to claim the item. These

approaches conflicted with the goal of providing rapid, self-service item recovery within an academic environment.

Internal Verification Necessity

An effective way to confirm ownership is through internal, knowledge-based authentication which is through security questions or item-specific information (Abrahim et al., 2024). This is to ensure that claims may only be made by users who actually know about the lost item (owner). BY verifying claimants through personal knowledge rather than external documentation, systems maintain efficiency without compromising integrity.

The Role of Admin Oversight

Administrative verification provides an additional layer of accountability and transparency (Ullah, Xiao, & Barker, 2019). Before final clearance, administrators examine submitted claims, double-check item details, and verify authenticity. This human-in-the-loop approach able to decrease false claims and guarantees that private information is kept confidential until it is validated.

Justification for LostLink

LostLink combines both verification techniques which are security questions validation to verify claimant legitimacy and admin that complete item distribution by approve or reject item. This dual-layer model balances security with efficiency, builds user trust, and directly addresses gaps in verification and privacy identified in earlier systems. In order to preserve system integrity and user confidence, this approach ensures that each claim is securely vetted and transparently processed.

2.5 Summary

The literature for this project shows that lost and found systems have come a long way, starting from simple, bare-minimum websites to more structured platforms with

verification, third-party security, automation, and better UI design interface. Although early systems like Ismail (2009) demonstrated that digitalizing the process was feasible, they entirely disregarded security, admin control, and data accuracy as they put focus on other main things such as basic item posting and searching. Mid-era systems such as Bataineh et al. (2015) and Tiwari et al. (2019) enhanced backend structure and usability. However, they still lacked a safe and responsible verification process, which in the end, can cause abuse and fraud claims (wrong ownership).

Recent solutions such as Kumar et al. (2022), Tan & Chong (2023), Abrahim et al. (2024) advanced their systems in terms of security, different roles in their system, tracking, and privacy. However, even they found it difficult to manage many at once as some relied too much on automation with minimal administrative oversight and some were overly strict and slow when they included third-party which is the police for verification, and some had privacy leaks where user contact info was exposed before verification. To sum up, it is evidence that systems either performed well in terms of security or usability, but rarely both together.

Across all studies, several gaps consistently appear such as weak authentication, lack of admin monitoring, privacy concerns, uneven usability testing, and ineffective search or tracking. No system was able to integrate admin-manages verification, security, and usability into a single, seamless workflow.

This is exactly where LostLink steps in to fill in the gaps from existing studies. By combining a straightforward, aligned interface with dual-layer verification architecture which are security question and admin approval on the claim, privacy-safe, and an organized Flask MySQL backend. In contrast to earlier systems, LostLink strikes balance between security and usability to give a dependable, transparent, and effective solution designed for the UTP campus.This chapter describes the overall methodology adopted to design, develop, and test the LostLink system from initial planning to implementation. The development process followed by evidence tat is conducted by survey, using Agile (part of SDLC) methodology as approach, tools, and techniques used in this project, and flowchart to showcase on how the system will be.

CHAPTER 3

3.1 Overview of Methodology

This chapter describes the overall methodology adopted to design, develop, and test the LostLink system from initial planning to implementation. The development process followed by evidence that is conducted by survey, using Agile (part of SDLC) methodology as approach allowing the system to be developed in iterative sprints with continuous refinement. Tools and techniques used in this project, and flowchart to showcase on how the system will be.

This chapter also presents the tools and technologies that are going to be used throughout development, as well as the system modelling such as use case diagram, flowchart for both roles (Users and Admin), Entity Relationship Diagram (ERD), system architecture, and activity diagram. By combining these components, this able to serve as the foundation of the structured approach to ensure that LostLink was created effectively, methodically, following user requirements.

3.2 User Requirements

The user requirements identify what the system must provide to meet the needs of UTP students, staff, and administrators. These specifications came from an initial survey and identified during the early planning stage. These specifications outline the functions and quality standards that the system must maintain.

Key Words:

FR = Functional Requirements

NFR = Non-Functional Requirements

Functional Requirements

For Normal Users		
No.	**Requirement**	**Description**
FR1	User Registration & Login	Users must be able to register and log in securely using their UTP email credentials.

FR2	Report Lost or Found Items	Users can submit their reports by filling in the description of the item along with photo, and location.
FR3	Search and Filter Items	Users can search for items by using keywords and filter by category.
FR4	Claim Items	User can claim the item and submit it by answering the security question that is given by the person who found the item.
FR5	View Claim Status	After the claim has been made, the item will change the status to "Pending" until admin approval.
For Admins		
No.	**Requirement**	**Description**
FR6	Manage Users	Admin must be able to view and remove users when necessary.
FR7	Review and Verify Claims	Admin must be able to make the decision carefully whether to approve or reject according to the answer that is received.
FR8	Manage Items	Admin can view, edit, or delete reported items.
FR9	View Activity Log	Admin can view the actions within the system to be able to detect any suspicious act.

Table 3.1: Functional Requirements

Non-Functional Requirements

No.	Requirement	Description
NFR1	Performance	The system should load pages within three seconds under normal circumstances.
NFR2	Usability	The system should be easy to navigate, responsive, and visually clear for all users.
NFR3	Reliability	To avoid loss, data must be consistent and routinely backed up.

NFR4	Security	Hashing must be implemented to save user password, and authentication should be required for sensitive tasks,
NFR5	Compatibility	The platform must run on modern browsers such as Google Chrome and Edge.
NFR6	Availability	The system should stay functional during testing and deployment with less downtime.

Figure 3.2: Non-Functional Requirements

These requirements formed the foundation of the LostLink system design and development. These oversaw the Agile development sprints outlined in the next section and ensure that both user-facing and administrative functionality were appropriately established.

3.3 Survey

Survey was conducted with the total of 52 respondents from UTP community. This is to understand their experiences with lost and found items and gather feedback on the need for a structured and centralized system. According to the responds, majority stated that they had lost or seen lost items on campus but were unable to recover or return to the owner. After asking them regarding LostLink, majority stated that it would be a great idea and that they would use the system for better service within UTP community. Refer to Appendix B. The survey results were later used to validate the system design choices, confirming user demand for a centralized lost and found solution. The survey results are provided in Appendix A.

3.3 Agile Approach

For this project, agile methodology will be used as an approach as it allow to work on the task step by step and receive early feedback from tested user which is my supervisor. The task will be in sprints as it allows to focus on feature at a time for more organize management and easy to track the progress.

Planning

The first step into this project of LostLink is to define the problem that is occurring in UTP. The absence of a systematic reporting mechanism was determined to be the main reason why staff and students usually struggle to handle lost and found things. Current procedures, such using social media sites like Telegram or WhatsApp, are informal, disjointed, and opaque, which frequently causes tension and delayed item recovery. 10 The system's essential features and capabilities will be talked about, taking into account user requirements and previous studies. These featured features including searchable listings, item reporting, security question-based claim verification, and admin management tools. The goal was to create an easy-to-use, user-friendly, and effective web-based solution for administrators and users alike. A thorough flowchart was made in order to illustrate the system's operation. This flowchart showed the user's path from item reporting to the approval and claim procedures. In order to guarantee that each feature had a clear operating rationale, it also established roles for administrators and regular users. The choice of technologies and tools was finally made. The frontend was developed using HTML, CSS, and JavaScript, and the backend was developed using Flask and Python. Visual Studio Code was selected as the main development platform because of its integration capabilities and flexibility, and MySQL was selected as the database option.

Sprint 1: Project Setup & Basic Interface

In this sprint, project setup will be focused and to design interface that bought ease for the user to use and navigate. This sprint is crucial as it lays the foundation for the entire development process. Establishing a stable backend structure and organizing project files using a consistent framework allows better collaboration and scalability as new features are added in subsequent sprints.

During this phase, planning the interface layout will be made such as planning the pages that is needed for this system such as the login and registration, homepage, reporting, find, claim, and admin dashboard. These pages are designed to be easy to use and navigate, without users having the difficulty or time consuming in learning on how to use the system.

The interface is designed using HTLM and CSS to create a clean and responsive layout. Flash will be used to configure the basic route and backend structure such as connecting each webpage to s specific URL route, acting as a bridge between the front-11 end which is HTML, CSS, and JS with backend logic (Python), and the interaction with the pages and data.

At the end of this phase, visual components are drafted to get initial feedback from my supervisor, Dr Siti Nurlaili before implementing full functionality in later sprint.

Sprint 2: Login and Registration

In sprint 2, the focus will be developing user authentication system which includes the login and registration pages. These are essential to ensure that only UTP students and staff have access to this system as the system is made only for UTP community.

The registration page is for new users to create their account by filling in their personal details such as their first and last name, UTP email address, ID matric, and password that need to meet the requirements. Input validation is applied to ensure that the details are being filled before they can create the account. Once the user is registered, the user's confidential detail will be stored in the database using hashed password to protect against unauthorized access.

Next is the login where users will login their details to access the system. Flask is used to handle form submission and session management. If the login is successful, the system would bring the user to homepage, meanwhile if the login failed, there will be an error message of "Login Failed" and user have a chance to change their password if they are unable to remember their password.

In this sprint, focus on login and registration is crucial as it needs to ensure that the system is only being used by UTP users. In the end of this sprint, my supervisor will be testing the registration and login to ensure the system is going well and to receive any feedback regarding this.

Sprint 3: Reporting System

In this sprint, the focus will be implementing the feature of item reporting whether it’s found or lost item. This allows the users to report lost or found items through form that needs the details of the item as well as photo uploads.

Using HTML, CSS, and JS to create a clean layout and Flask to handle the form's backend logic, the reporting form is made to be easy to use and includes input fields like item name, description, date lost/found, location, and the ability to upload a photo. This information helps improve clarity and increases the likelihood of accurate identification and claim.

Upon the submission of the form, Flask will capture the entered information and securely stores it in a MySQL database, associating each report with the reporting user’s account. This makes it possible to properly track items and access them later using the system's "Reported Items" feature. In order for the reported item listings to appear, uploaded photographs are saved in a specific folder, and their file paths are noted in the database.

This sprint plays a crucial role in the system establishing the system’s functionality by enabling structured and efficient reporting. This will be tested by submitting a form to check if the reporting feature works and see if the data made it in the database.

Sprint 4: Search & Filtering Function

In this sprint, the focus will be developing the search and filter functionality to help the users to search for their missing item. This feature is crucial to improve the overall usability of the LostLink system and reduce the time users spends browsing through many numbers of items.

With the search function, users can enter keywords such item name or category to see results that match. Additionally, there are filtering options now available, allowing users to select items according to predetermined standards like date, item type, or 13 locations. When the item list is updated in real time or when the form is submitted, these routines are made to work dynamically with it.

From technical side of this project, user input is captured via HTML forms, and Flask manages the backend logic by executing queries against the MySQL database using search or filter parameters. Only things that meet the user's criteria are shown in

the results, which are subsequently presented on a special page. Flexibility and relevance are ensured by filtering logic's optimization to manage partial matches and multiple parameters.

This sprint enhances the system’s efficiency by allowing users to locate items quickly without scrolling through the entire database. It also complements the reporting feature developed in the previous sprint by making submitted data accessible in a meaningful and user-friendly way

Sprint 5: Claim and Approval

In sprint 5, the focus will be towards claim and approval that enable for users to request ownership of reported items and ensures that only valid claims are approved. The approval will only be made by the admin of the system to avoid any bias or theft that may occur. This act as a security and trust to the system.

Users can click on a "Claim" button because they found an item they think is theirs. The initial reporter will ask them a security question, such as "What name is engraved on the item?" This data is gathered via a form and saved for future use. The system admin can then examine the claim request's data in the admin interface. The administrator assesses the validity of the claim by comparing the user's response with the one that is recorded in the system. If accepted, the item will be noted as claimed, and the return can be arranged by both parties. The item is still posted and available for claims in the future if it is refused.

This functionality will be created with MySQL to maintain claim status and approval history and Flask to manage form submissions and decision logic. For the benefit of both users and the administrator, the interface will be maintained straightforward. 14 This sprint's completion improves the system's security, transparency, and dependability while promoting accountability and equity in UTP's lost and found procedure.

Sprint 6: Admin Panel

In sprint 6, the focus will be on the admin of the system which is going to be different than other users as admin has its own role of viewing the user reports and listing, flag any suspicious posts, remove any content that is inappropriate and can bring violation, and approving or rejecting the claims. The admin role plays a vital role in maintaining the system's accuracy, security, and integrity.

In order to maintain role-based control, the admin login is distinct from other user access which is students and staff of UTP that report lost or found item. A dashboard showing all reported items, pending claims, and user activity records is visible to the administrator once they have logged in. From this point on, the administrator can approve or reject item reports, particularly when there are duplicate or improper listings. When a claim is requested, the administrator can examine the claimant's security response and determine whether to accept or deny the claim.

While MySQL manages the storing and retrieval of admin operations, Flask is used to redirect pages that are exclusive to the admin. To cut down on errors and facilitate speedy decision-making, the admin interface was created with simplicity in mind. Every item or claim record has action buttons like "Approve," "Reject," or "Delete," as well as user information and timestamps for traceability.

This sprint is essential to guarantee that all platform interactions are thoroughly examined and verified, as well as that the LostLink system is kept trustworthy and moderated. Additionally, by providing authorized personnel with the necessary tools to efficiently monitor and manage user-submitted content, it gets the system ready for actual deployment.

Sprint 7: Testing

For the last sprint in agile methodology for this system, testing will be conducted to all implemented system features to ensure proper functionality, reliability, and user experience. This is crucial as to identify any potential error that may arise, fix bugs, and confirm that all core function work as expected before moving forward to deployment.

Verifying individual components like login/registration, reporting, searching, claiming, and admin approval were all part of the testing phase. To ensure seamless

component interaction, each module underwent independent testing (unit testing) and subsequently group testing (integration testing). When a user files a report, for instance, testers confirmed that the item is listed accurately and may be searched, claimed, and processed through admin approval. Other than that, a small group of users will be invited to engage with the system and offer input as part of the usability testing process. This will evaluate the form's readability, navigational simplicity, and responsiveness on various screens and devices. The tested users may give feedback and comments to enhance the system.

Through this sprint, LostLink will be defined and polished to ensure that the system performs well in real-world scenarios. Therefore, this sprint mark as a last preparation before going into deployment and present it to my supervisor, examiner, and FYP committee.

After all sprints were completed, the integrated system underwent functional and usability testing to verify stability before final deployment.

3.4 Tools/equipment required

The development of LostLink needs a well-integrated set of tools and technologies to guarantee system efficiency, functionality, and usefulness. Based on compatibility performance, and appropriateness for developing web-based applications, each tool has been chosen.

Category	**Tool/Technology**	**Purpose**
Platform/IDE	Visual Studio Code (VS Code)	Code editing
Language & Runtime	Python 3.x	Server-side logic
Web Framework	Flask 2.x	Routing, templates, form session
Templating	Jinja2	Server-rendered pages
Browser	Chrome/Edge	Testing, UI verification
Frontend	• HTML • CSS	UI, styling, client logic

	• JavaScript	
Backend	• Language Programming: Python Framework: Flask	Functions
Database	MySQL	Persistent storage, FK constraints
Deployment (dev)	Localhost (Flask)	Development & UAT
Deployment (future)	Render/Railway/Cloud MySQL	Multi-user access, scalable

Table 3.3 Shows Tool and Technologies that will be used for this project.

3.4.1 Platform: Visual Studio Code (VS Code)

Figure 3.1: Visual Studio Code (VS Code) Icon.

Visual Studio Code is a Microsoft source code editor that provide lightweight and beneficial with many functionalities tool for user especially programmer. It acts as a universal code editors as it allows to use various programming languages such as Python, Java, SQL, C++ & C and others which making it easy and friendly to develop LostLink system of lost and found as Visual Studio Code provides a unified workspace where frontend and backend can be developed and tested in one platform only instead of using different types of platform for each.

3.4.2 Programming Language: Python

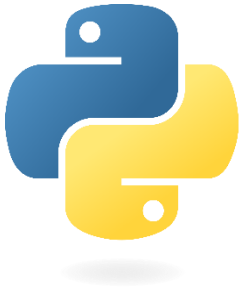

Figure 3.2: Python Icon

Flask is an adaptable and lightweight Python web framework that makes it possible to create web apps quickly. It is employed to manage server-side logic, routing, and front-end-to-database connectivity. A system like LostLink would benefit greatly from Flask's simplicity and modularity.

3.4.3 Backend: Python & Framework: Flask

The backend of LostLink is developed using Python and the Flask framework, which combining it together will be able to handle all system logic. CRUD routines, database queries, authentication, and data validation are all handled by Python. Flask is an adaptable and lightweight Python web framework that makes it possible to create web apps quickly. It is employed to manage server-side logic, routing, front-end-to-database connectivity, and frontend via Jinja2 templates. A system like LostLink would benefit greatly from Flask's simplicity and modularity. With these together, LostLink will be able to deliver fast and secure backend functionality.

3.4.4 Frontend: HTM, CSS, JavaScript

Figure 3.3: HTML, CSS, JS Icon.

For frontend, these three fundamental web technologies will be used to create an interactive user interface:

- HTML: Pages content is organized.
- CSS: Improves responsiveness and aesthetics.
- JavaScript: Adds interactive features like dynamic and form validation.

3.4.5 Templating Engine: Jinja2

Figure 3.4: Jinja2 Icon

Jinja2 is the templating engine used by Flask to generate dynamic HTML pages. It enables data from the backend to be inserted into templates using placeholders, loops, and conditional statements.

Jinja2 makes it possible for LostLink to have features like item listings, search results, review claims information, and page customization depending on whether the user is an admin or a regular user. Using Jinja2 ensures cleaner code separation, enhances maintainability, and less repetition across several HTML pages are all ensured by using Jinja2.

3.4.6 Database: MySQL

Figure 3.5: MySQL Icon.

User accounts, reported items, and claim history are among the many types of system data that are stored in MySQL, a relational database management system (RDBMS). For managing an expanding dataset, it offers secure data processing and effective query performance.

3.4.7 Browser: Google Chrome, Microsoft Edge

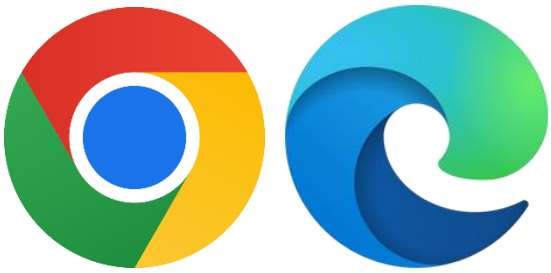

Figure 3.6: Google Chrome and Microsoft Edge Icon.

When an application is being developed, its interface is tested and visualized using web browsers such as Google Chrome and Microsoft Edge. They assist developers with element inspection, frontend behaviour debugging, and making sure that the site is responsive on various screens and devices.

The selected toolchain ensures smooth integration between the frontend, backend, and database layers, enabling rapid debugging and modular feature expansion.

3.4.8 Deployment (Development): Localhost Server

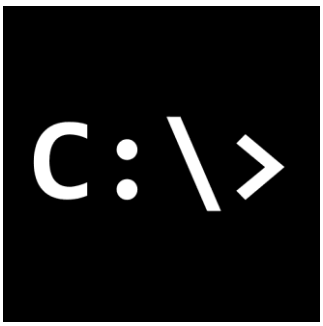

Figure 3.7: Command Prompt.

During development, Flask's built-in-development server which is https://127.0.0.1:5000/ is used to deploy the system locally. This environment allows rapid testing of all system features such as reporting items, searching, viewing, and verifying claims before conducting public deployment that will enable the user from other device to access LostLink. Easy debugging and secure experimentation are guaranteed by localhost deployment, which doesn't impact real users or external data.

3.4.9 Deployment (Future): Render/Railway/Cloud MySQL

LostLink is built to accommodate future deployment on cloud-based platforms like Render, Railway, or Cloud MySQL providers, even though the project starts by operating on a local development server. These platforms allow multiple users to access the system simultaneously and provide automated scaling, uptime monitoring, hosted MySQL databases. The system architecture is entirely compatible with online hosting for future enhancement, even if cloud deployment was not performed owing to project scope and constraints.

3.5 System Modelling and Design

3.5.1 Use Case Diagram

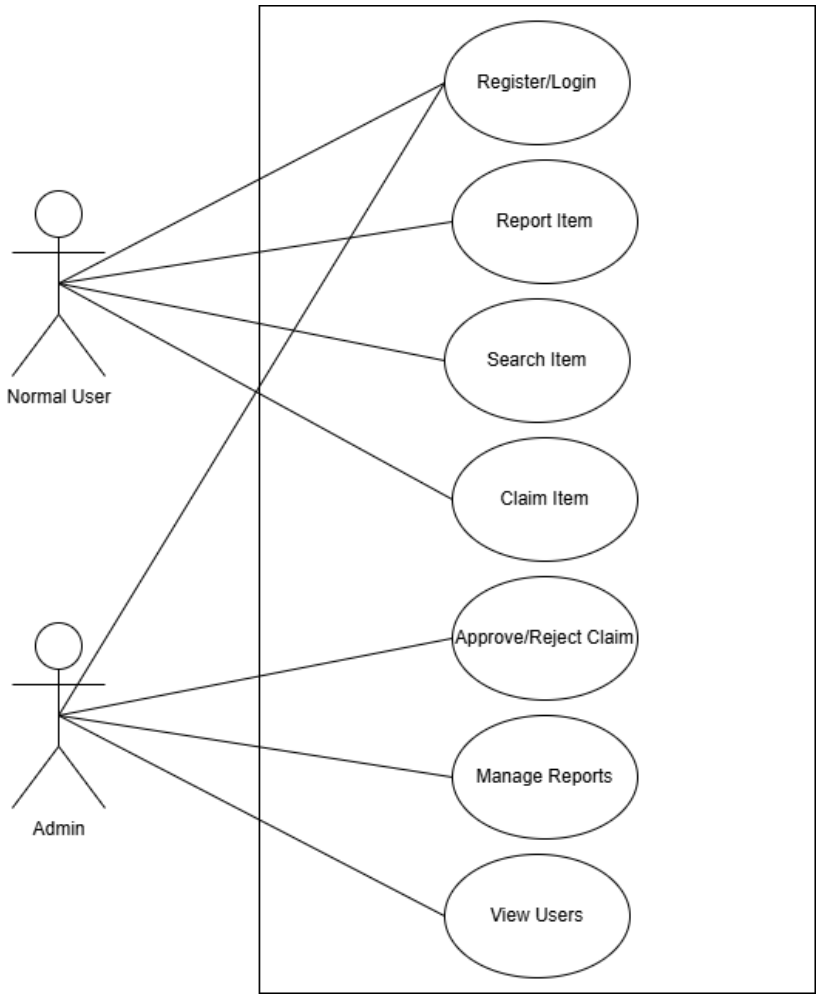

Figure 3.5 Shows User Activity Diagram

The use-case diagram shows the main functionalities which are login/register, report, search, claim, and admin role as well as showcasing the interaction of normal user (students and staff of UTP) and admin with the system, LostLink.

Normal User (UTP User)

The normal represents the user in UTP which are students and staffas the system is made for UTP community. Their main use cases include:

- **Login/Register:** Users needs to fill in their credential before entering the system or create their account if haven't. The system only accept UTP users and will receive invalid if its not UTP account.

- **Report Item:** Users report lost or found item that includes description of the item, security questions, and upload photos features.

- **Search Item:** Allow the user to search for their item by using search and filter functionality.

- **Claim Item:** When the user sees item that seems to be theirs, this function allows the user to claim the item by answering the security questions and wait for admin approval.

The primary user-facing element of LostLink is interactions, which allow the users to engage in the reporting and claiming process.

Admin

The administrative manages and verifies all user activities to maintain system reliability. The admins use cases includes:

- **Approve/Reject Claim:** Admin reviews the claim request made by normal users and see if it's the right owner or not. If it the right owner, the admin will approve, otherwise the admin will reject the claim.

- **Manage Reports:** Allow the admin to view, edit, or delete item or duplicate reports that may occur.

- **View Users:** Shows list of registered users in LostLink for monitoring and maintenance purposes.

The admin's efforts ensure data integrity, prevent misuse of the system, and maintain the system's legitimacy. This use-case diagram effectively captures the intended functionalities of LostLink during design phase. This sets the stage for other modelling such as flowchart, ERD, and system architecture to showcase on more detailed of the system and better understanding.

3.5.2 Flowchart

Flowchart is created to showcase the flow of the system from users and admin point of view. As for users, there are three flowchart which are from the founder's perspective, the owner who lost their item perspective, and claiming perspective. As for admin's part, there are two parts which are

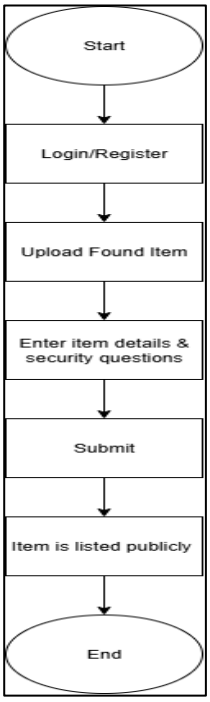

Figure 3.6 Reporting Process

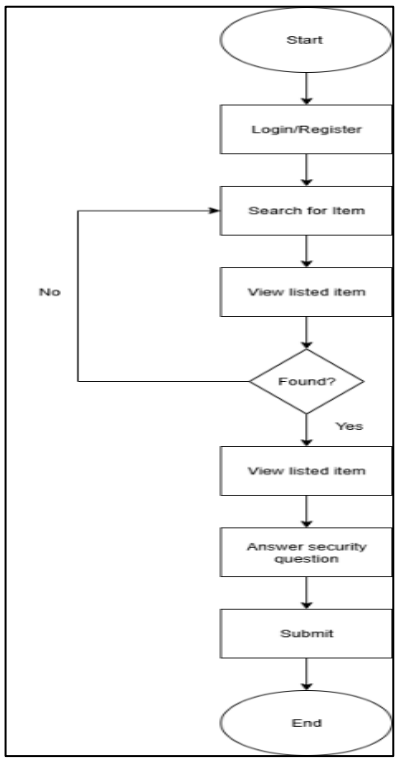

Figure 3.7 Claiming Process

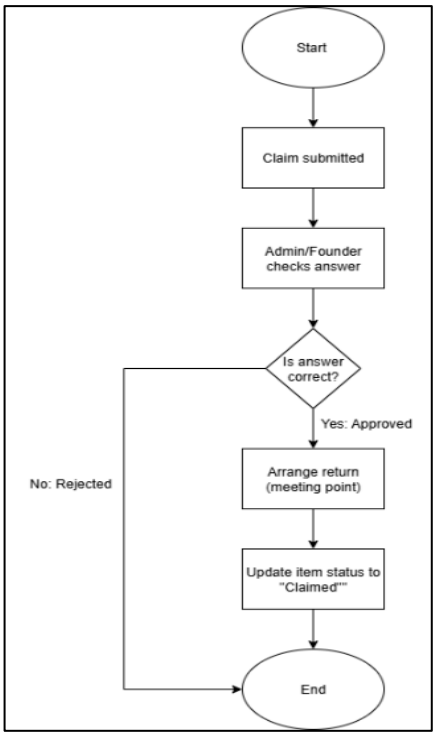

Figure 3.8 Admin Verification & Returning Process

Figure 6(a) starts from the user login/register the system to ensure authenticated access. Then the user will navigate to upload missing item that they have found and starts to enter inputs regarding the item, uploading photo of the item, and set security questions for security purposes. After that, the user submit the form, and the item will be displayed publicly in the system where the founder may find the item soon. That's the end for this figure.

Figure 6(b) starts from user login/register the system to ensure authenticated access. Then the user who is the owner of the item searches for their lost item through many lists of lost items. Since the system have the function of search keyword and filtering, the user can type in their item for shortcut for each finding and less time-consuming. Once the item is found, the user will then start answering the security questions to proof that they are the owner of the item and then submit the form.

Figure 6(c) is about submitting the claim that starts from the user claiming the item when they have finally found their item along with answering the security questions. Once it is done, the admin will then checks the answer submitted from the user and see if it the real owner based on how they answer the question or shows proof that is

it theirs. After admin approved the claim, meeting will be arranged between the founder and the owner to retrieve back the item and settle the case. After it is done, the user must update the status to “Claimed” to show that the case of an item is successful being done.

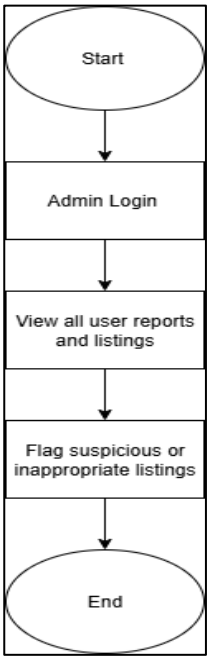

Figure 3.9 View Reports

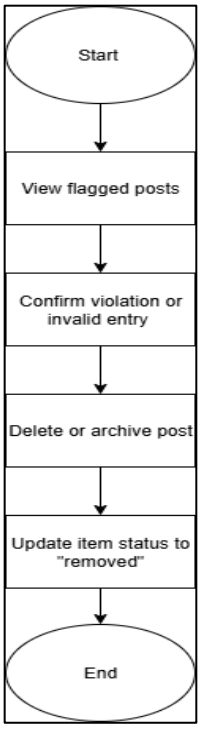

Figure 3.10 Remove Reports

For this part, it shows the admin’s point of view of the system. For figure 6(d), Admin starts login by filling in their email/username and password and the system will bring the admin to their side of the system which is different from regular users (students and staff). The admin has the ability to view all user reports and listings and to see if they are any suspicious or inappropriate listings in the system, if there any, the admin will flag that specific post for further actions.

For figure 6(e), it is about admin viewing the flagged posts and confirm that the post bring violation and inappropriate to the system. With this, the admin has the ability to remove this kind of post.

3.5.3 Entity-Relationship Diagram (ERD)

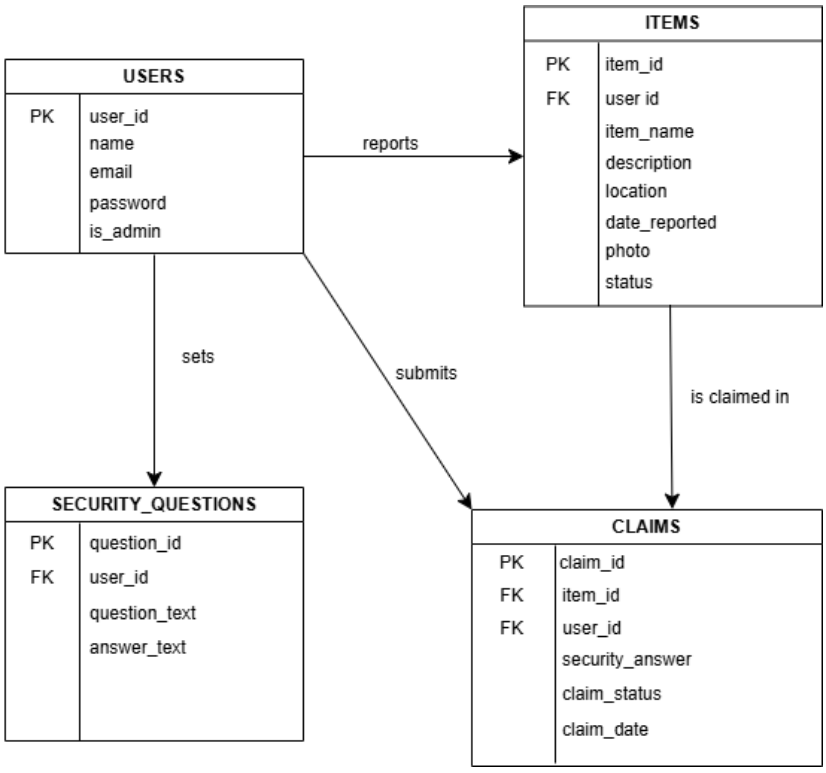

Figure 3.11 Shows ERD of the System LostLink

The Entitiy-Relationship Disgram (ERD) shows the conceptual database design of the LostLink system before implementation. It defines four core entities which are Users, Items, Claims, and Security_Questions. The relationship between them, ensuring data consistency, normalization, and integrity within the MySQL database.

1. **USERS Entity**

   The USERS table stores all registered users, including both normal users which are students and staff of UTP and admin of LostLink. Each user is identified by a unique primary key which is user_id. According to above figure, there are attributes such as name, email, and password store personal credentials, while is_admin differentiates between normal users and system administrators.

   This entity serves as the parent table for other dependent entities.

2. **ITEMS Entity**

   The ITEMS table records all lost and found items reported by users. The attrbutes contain inside ITEMS table are item_name, description, location,

date-reported, photo, and status. User_id act as foreign key that connects each item to the user who reported the item, therefore this establishes a one-to-many relationship between the USERS and ITEMS tables. With this, this ensures that each user can report more than one item, but only one user can report each item.

3. **CLAIMS Entity**

   The CLAIMS table manages the ownership verification process when users attempt to claim lost items. Each claim is identified by claim_id with its foreign keys of user_id and item_id which connects to the ITEMS and USRES tables respectively. The verification response, admin decision, and claim submission date are recorded via additional fields like security_answer, claim_statis, and claim_date. This creates relationship in which an item may have more than one claim request. But each claim is associated with a single user and a single item.

4. **SECURITY_QUESTIONS Entity**

   The SECUIRTY_QUESTIONS table stores each user's verification questions to ensure ownership authenticity throughout the claim procedure. Each record is connected to the USERS table by the foreign key user_id and includes a question_text and its matching with answer_text. This forms a one-to-one relationship between USERS and SECURITY_QUESTIONS, as each user sets one question for identity verification.

Relationship	**Cardinality**	**Description**
USERS to ITEMS	One-to-Many	A single user can report multiple items
USERS to CLAIMS	One-to-Many	A single user can submit multiple claim different items

ITEMS to CLAIMS	One-to-Many	A single item can receive multiple claim requests from different users
USERS to SECURITY_QUESTIONS	One-to-One	Each user sets one unique security question for verification

Table 3.4 Shows ERD in Database Schema

3.5.4 System Architecture Diagram

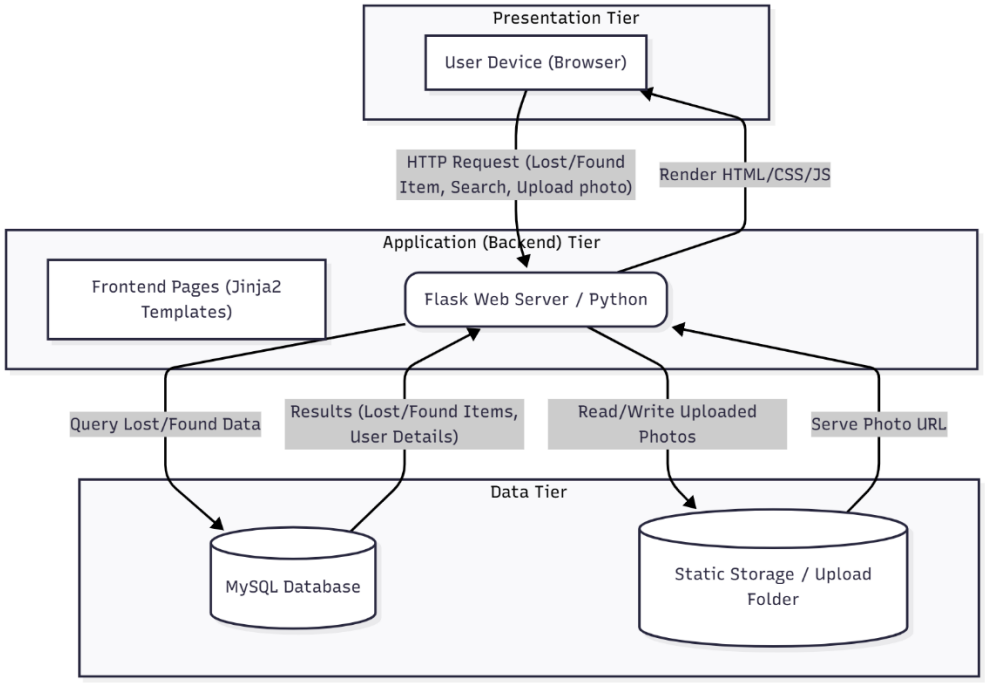

Figure 3.12 Shows System Architecture Diagram of Lostlink

The presentation, application, and data tiers make up the LostLink system's three-tier architecture. This structure guaranteed scalability, maintainability, and organized separation of system functions.

1. **Presentation Tier**

   Presentation tier is put on top of the layer that represents as user that interact with the system. The users is able to perform actions such as login/register, reporting, searching, and claiming lost items with the help of this component

that developed the interface (frontend) which are HTML, CSS, and JavaScript. The browser displays rendered pages in response to HTTP requests sent to the Flask backend.

2. **Application (Backend) Tier**

   The middle layer of the system architecture is Application that is powered by the Flask web framework (Python) which manages all routing, business logic, and system operations and functions. Other than that, it handles authentication users to ensure privacy and security, processes claim, and coordinates data flow between the frontend and the database. To add on, Flask serves function for image uploads and static photo URLs from the upload directory.

3. **Data Tier**

   In data layer, there are two storage components which are MySQL Database, which keeps all structured data such as user credentials, claim records, and item details and Static Upload Folder, which keeps posted of picture of found or lost items. With this both components, its able to communicate with Flask backend to provide reliable data retrieval and image access.

To sum up these three tier in system architecture for LostLink, it enhances the system's modularity to make it ease to maintain and expand for future versions for improvements which may be adding mobile support or cloud-based server.

3.5.5 Activity Diagram

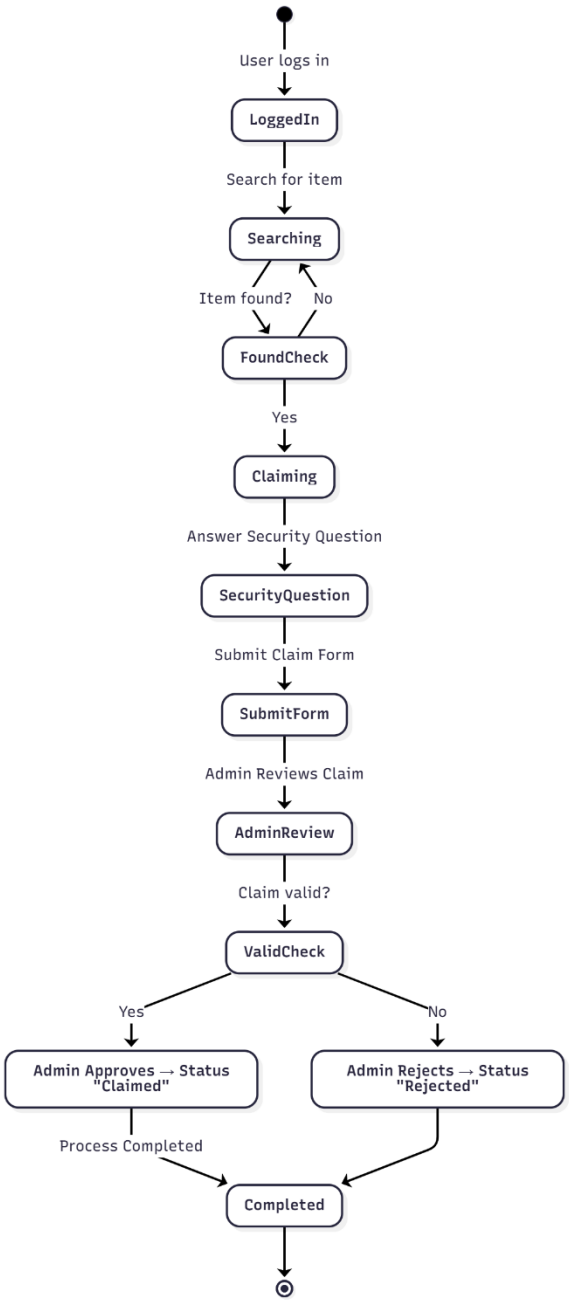

Figure 3.13 Shows Activity Diagram on the flow within LostLink

The activity diagram shows the step-by-step workflow of a user thorough the system. In this diagram, the focus will be on claiming items process as it involves both normal users and admin for approval on the claim request.

The process begins with user's login/register their account to gain access of LostLink, afterwards, the user searches for their item either in view items or search items with keyword and filtering function that reduces time-consuming.

If the is not found, the user can refine their search and try again. For example, by typing “Bottle” and select category on found from there, the items shows up and the user able to claim the item. If the item is found, the user clicks “Claim Item” and answer security questions with any answer they would like to put.

After the user answers the security question, this leaves for the admin to take on their part which is reviewing the answer to see if it’s the right ownership.

- If the claim is valid, the admin approves the claim and will updates to the user through notification that their claim is approved and provide contact of the founder. The item will be showed as claimed from pending.
- If the claim is invalid, the admin rejects and user will receive notification regarding rejection on their claim request. The item will be showed from pending back to found to notify to users that the item is still on search.

After the admin’s decision, the system updates the record in the database and marks as claimed. This activity diagram ensures traceability and transparency in claim verification by showing every possible path and decision taken within the LostLink on claiming process.

3.7 Summary

In this chapter, it discussed the overall methodology applied in developing the LostLink system. The final year project was managed in iterative sprints using an Agile methodology, which allowed for design flexibility and ongoing improvement based on feedback. The frontend, backend, and database components were integrated with the help of tools and technologies such as Flask, MySQL, HTML, CSS, and JavaScript.

Using diagrams such as the Use Case, Flowchart, Entity-Relationship Diagram (ERD), System Architecture, and Activity Diagram, which each shows how data and processes interact within the system. Additionally, a locally hosted MySQL database for development and its possible migration to a remote or cloud environment for deployment were also described in the Database Hosting Setup in section 3.5. Overall, this chapter of Methodology ensures that LostLink was systematically designed, effectively executed, and scalable for upcoming enhancements and practical use.

CHAPTER 4

4.1 Overview of Result and Discussion

In this chapter, its focus will be on the outcomes that are obtained after the implementation and testing of the LostLink system. The results that were able to receive from going through categories of interface outcomes, functional testing, usability feedback, performance evaluation, and comparative discussion. Each section shows how the developed system, LostLink, can solve the problems that is mentioned in the problem statement which were the lack of monitor and traceability, no centralized system to manage this issue of lost and found, and the lack of awareness. Other than that, in this chapter, it will show the goals that were able to achieve that was mentioned in Chapter 1.

4.2 System Implementation (Backend & Frontend)

The implementation of LostLink was carried out by using Flask framework for backend operations and HTML, CSS, and JavaScript for frontend. The integrated development environment (IDE) used for the development was Visual Studio Code, which provided efficient debugging tools, extension management, and support for Python and web technologies.

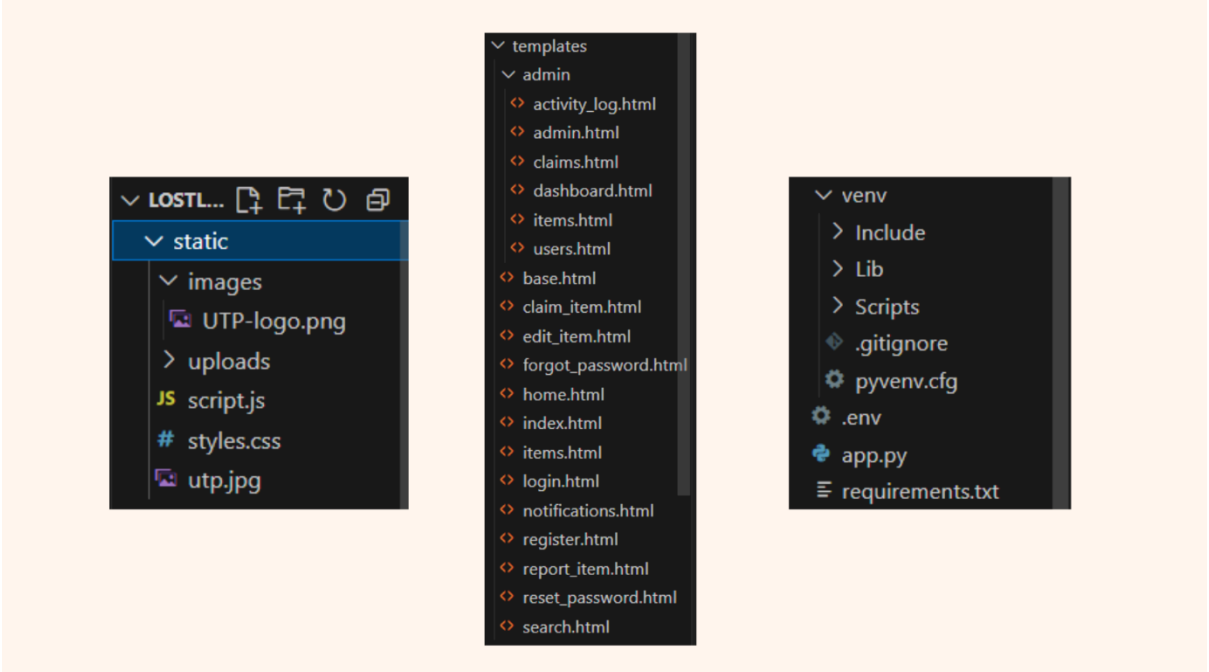

Figure 4.1 The structure of LostLink in Visual Studio Code (VSC)

Figure 4.1 shows the Visual Studio Code environment showing the primary application structure. The system is divided into logical modules consisting of App.py, templates/ for Jinja2, static/ for CSS and JavaScript, and uploads/ for user-submitted photos comprise the system's logical modules.

4.2.1 Backend (Flask):

The backend of Lostlink was developed using the Flask Python framework, which was selected for its lightweight architecture and ease of handling web routes, sessions, and database interactions. The backend acts as the system's logic layer, managing user authentication, and CRUD operations, and data validation.

Using the Flask that is integrated with a MySQL database using flask_mysqldb because it allows the backend and database server to communicate directly. The database connection is configured locally within the development environment which allow quick testing and effective data handling. Database credentials, including host, username, password, and database nae, are safely managed using environment variables kept in the .env file.

The backend also has various other advances functionalities such as password hasing via wekzeug.security, activity logging regarding the actions of users and admins, flash notifications for system alerts, and session-based control to distinguish between normal users and administrators. According to Figure 4.x: Visual Studio Code workspalce open to show the overall structure of the backend, main application file App.py, templates/ directory for Jinja2 files, and the static/ folder container CSS, images, and scripts JS.

Each route served a particular function within the system, and hence, the separation of concerns is quite distinct. An example that can be taken is the /report_item route handles reporting by the user and stores the record in the database:

Figure 4.2 Shows an example coding of /report_item in app.py

The backend also included admin routes protected by decorators, such as the figure below this:

Figure 4.3 Shows an example of /admin/dashboard in app.py

Access control is applied with a customer decorator called @admin_required that ensures only users who have the is_admin flag in the database have access to management pages.

To add on, the backend logs each major user action such as reporting, claiming, approval, or rejecting using a customer log_activity() function which records activity in the activity_log table.

To sum up, Flask's modular structure allowed the development of a responsive and secure lost and found management system, effectively integrating database operations, authentication, and admin control within lightweight backend environment,

4.2.2 Frontend (HTLM, CSS, and JavaScript)

The frontend of LostLink was developed by using HTML, CSS< and JavaScript for its simple-to-use, responsive interfaces targeted for both users and administrators. The interface design focuses on accessibility, clarity, and user experience to help users easily navigate, login, register, report, search, claim, and approval of the claim from admin's side.

The frontend pages are organized under the templates/ directory and rendered dynamically with Jinja2 templating, embedding Python data directly from Flask into HTML files. This ensures real-time synchronization between the backend and frontend. An example that can be taken is when a new item is reported or gets claimed, it will be shown on the interface in real time without the need for refreshing or hard coding as it takes straight to the item being in the view items after clicking the submit report button.

Each main page, such as login.html, home.html, report_item.html, and admin/dashboard.html, extend from a common base.html template to ensure consistency in layout and navigation bars in all pages. This structure minimizes code duplication and simplifies maintenance.

The Cascading Style Sheets (CSS0 files stored under the static/style.css which define the system's visual appearance, produced by a pastel blue color scheme that provides a modern, minimalistic feel to the system's interface. According to Figure 4.x, it shows how the frontend templates are represented inside the Visual Studio Code, making use of a div layout approach as well as series of Jina2 syntax blocks that are responsible for providing the structure relating to how the content should be represented.

Lastly, JavaScript, or known as JS, is used to enhance interactivity and user experience. The script.js file is included in the static/ directory same as style.ss and referenced by the base.html file. The script handles UI interactions such as closing flash notifications, basic form validation, and smooth transitions for alerts.

```
// Flash close button handler
document.addEventListener("click", function (e) {
  if (e.target.classList.contains("close-btn")) {
    const flash = e.target.parentElement;
    flash.style.opacity = "0";
    flash.style.transform = "translateY(-10px)";
    setTimeout(() => flash.remove(), 300);
  }
});
```

Figure 4.4 Shows an example of JavaScript

The base.html file includes this script using the following reference:

```
<script src="{{ url_for('static', filename='script.js') }}"></script>
```

Figure 4.5 Shows the connection to scripts.js from another file (HTML)

This enables all pages that extend the base layout to automatically load the JavaScript routines. The frontend ensures that LostLink delivers a unified and responsive user experience when combined with Flask's Jinja2 templating and structures CSS design.

4.2.3 Database Implementation (MySQL)

The database for LostLink was implemented using MySQL Workbench to host it locally. All user data, reported items, claim fillings, activity logs, and system alerts are stored in one single location. The Flask backend connects to this database via the flask_,mysqldb library and performs SQL operations like inserting, updating, and connecting to this database, and retrieving data.

Users, items, claims, activity_log, and notification are the five primary tables that make up the relational structure of the database. Each table is created with primary and foreign key constraints to preserve referential integrity and guarantee consistency between user activities and system records. The database was normalized up to the third normal form (3NF) to decrease the redundancy and improve query efficiency.

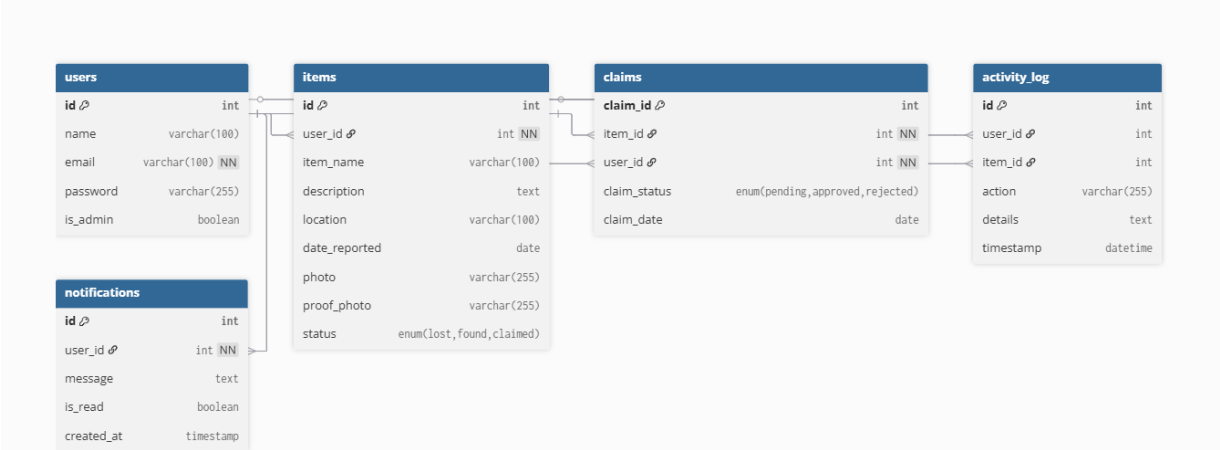

Figure 4.6 Shows ERD in Database Schema (After Implementation)

ERD Structure of Lostlink

Entities	**Attributes**
USERS	- Id (PK) - Name - Email - Password - Is_admin
ITEMS	- Id (PK) - User_id (KF) - Item_name - Description - Location - Date_reported - Photo

	- Proof_photo - Status
CLAIMS	- Claim_id (PK) - Item_id (FK) - User_id (FK) - Claim_status - Claim_date
ACTIVITY_LOG	- Id (PK) - User_id (FK) - Item_id (FK) - Action - Details - Timestamp
NOTIFICATIONS	- Id (PK) - User_id - Message - Is_read - Created_at

Table 4.1 Shows ERD Structure of LostLink

Table Name	**Primary Key**	**Description**
users	Id	Stores user credentials such as name, email, hashed password, and user role (is_admin).

Items	Id	Includes information about every lost and found item, including the item's name, description, photo, location, date of reporting, and reporter (user_id).
claims	Claim_id	Manages claim submission by users, including claim date, status whether pending, approved, rejected.
activity_log	Id	Tracks all actions performed by users or admins.
notifications	id	Stores system-generated messages for each user.

Table 4.2 Shows the Table Name, Primary Key, and Description

Entity Relationships

Relationship	**Description**
users to items	1:N: one user can report many items
users to claim	1:N: one user can make many claims
users to notifications	1:N: each user can receive many notifications
users to activity_log	1:N: each user may perform many actions
items to claims	1:N: an item can have many claim attempts
items to activity_log	1:N: an item can appear in many logged actions

Table 4.3 Shows Entity Relationship

According to 4.x, it shows Entity Relationship Diagram (ERD) of the LostLink database showing the relationship between users, items, claims, activity logs, and notifications. There is a one-to-many relationship between the users and item as user may report more than one lost or found item. Claims and the activity log connect both the users and items to ensure traceability in all claim submissions and administrative actions. The notifications is related only to the users, enabling the system to present personalized notifications for either claim updates or administration decisions.

4.2.4 Final System Flowchart

4.2.4.1 Normal User Workflow

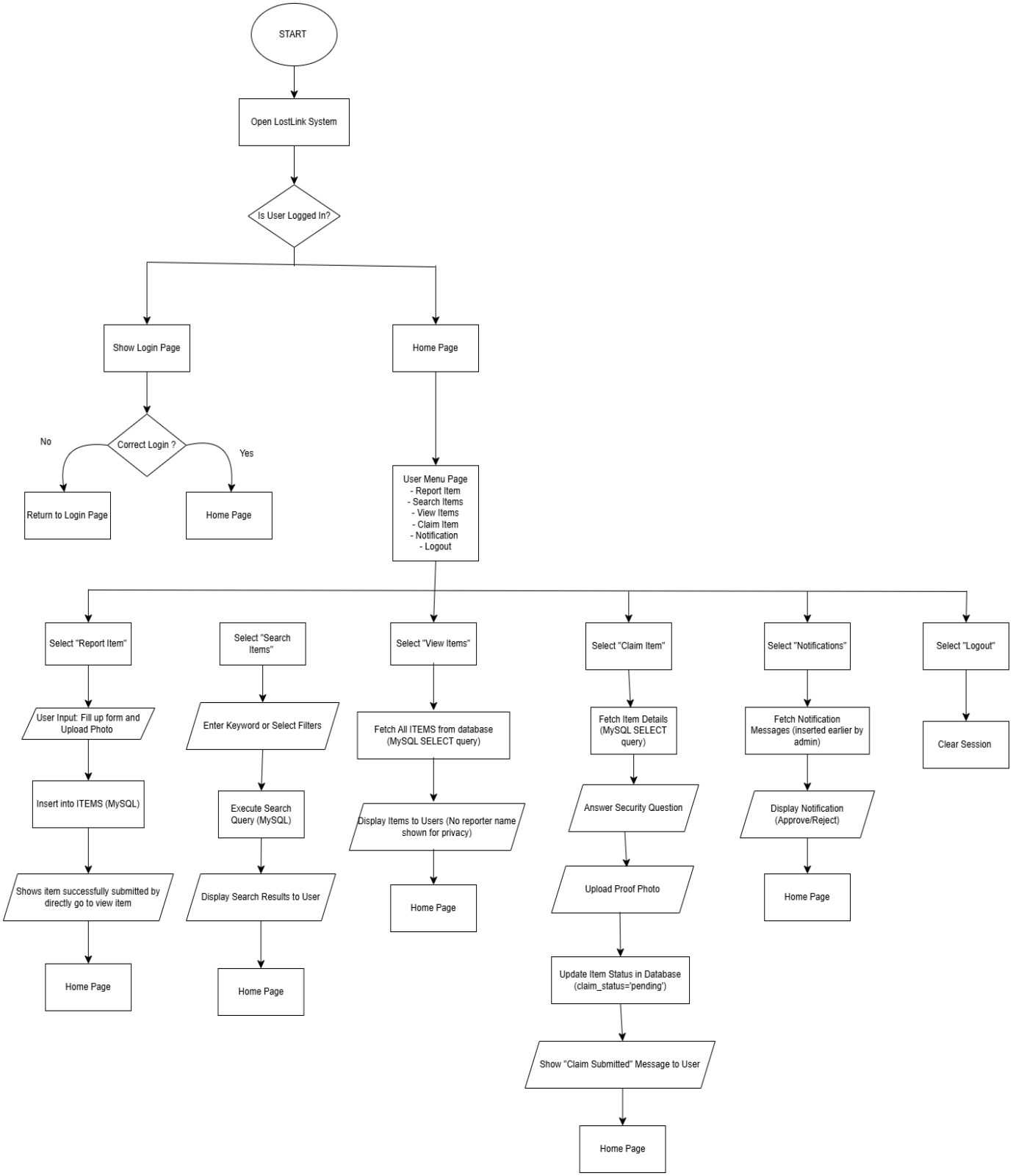

Figure 4.7 Shows Flowchart of Normal User

Explanation:

This flowchart illustrates the workflow for normal users using LostLink system. It outlines the sequence of actions taken by the user starting from login/register until end session. The main activities include reporting items, searching items, viewing items, and claiming item. All system processes are guaranteed to follow an organized path and exhibit consistent behavior thanks to the flow. Traceability, precise data management, and seamless user-system interaction are all made possible by this procedure.

1. **Login/Register**

   For login/register part, it is for users accessing the LostLink system. Usres enter their registered email and password, which the system validates against the stored hashed credentials in the database. In order to ensure secure access control, invalid login attempts send the user back the login page with an error message. As a normal user, a successful login opens an active session and takes them to Home Page. In addition to preventing unwanted access, this method ensures authentication fidelity.

   When it comes to registering, the system ensures that only UTP community can access the system as LostLink is made for UTP community only. The system puts strong security when it comes to credentials as the requirement to register is to put strong passwords of eight lengths, at least one capital letter, one small letter, one symbol, and 1 number. Other than that, the system has “Confirm Password” to ensure that the user remember the password they entered.

2. **Report Item Flow**

   When the user selects “Reporting Item”, the user fills in the form as well as uploads a photo of the item that is lost and submits the report. While that, the system validates and stores the record in the database. The flow ensures that each report is captured accurately. After the user submits the report, it immediately takes the user to view items to show that the item is now displayed in the system.

3. **Search Items Flow**

   When the user selects “Search Items”, the system is able to use the function of search keywords and filtering to search the items with ease instead of scrolling through the system that can take up unnecessary time. Efficient and responsive search operations are guaranteed by the structured flow. This facilitates rapid item discovery, which is crucial to lowering user annoyance and enhancing system usefulness.

4. **View Items Flow**

   When the user selects “View Items”, the system fetches the latest items from the database and displays the latest items from the database and to the user. This process guarantees transparency and each browsing of lost and found entries. Other than that, it offers a standardized interface that helps users find pertinent items.

5. **Claim Item Flow**

   When the user seems to find the item that belongs to them, they will click the “Claim Item” button and security questions will be displayed (input by the person who found the item). The user will have to fill in the answer to the security question along with uploading photos to strengthen the ownership of the item. After the user is done and click submit, the system will update the item as “pending” for admin review. This structured process ensures that claims are verified fairly and linked to valid evidence. It forms the foundation for the admin’s approval and notification process.

6. **Notification Flow**

   Once a claim is accepted or denied by the admin upon the claim that has been made, notifications will be shown to the user regarding the claim whether approve or reject. This method ensures that users are promptly notified about the status of their claims and system updated.

4.2.4.2 Admin Workflow

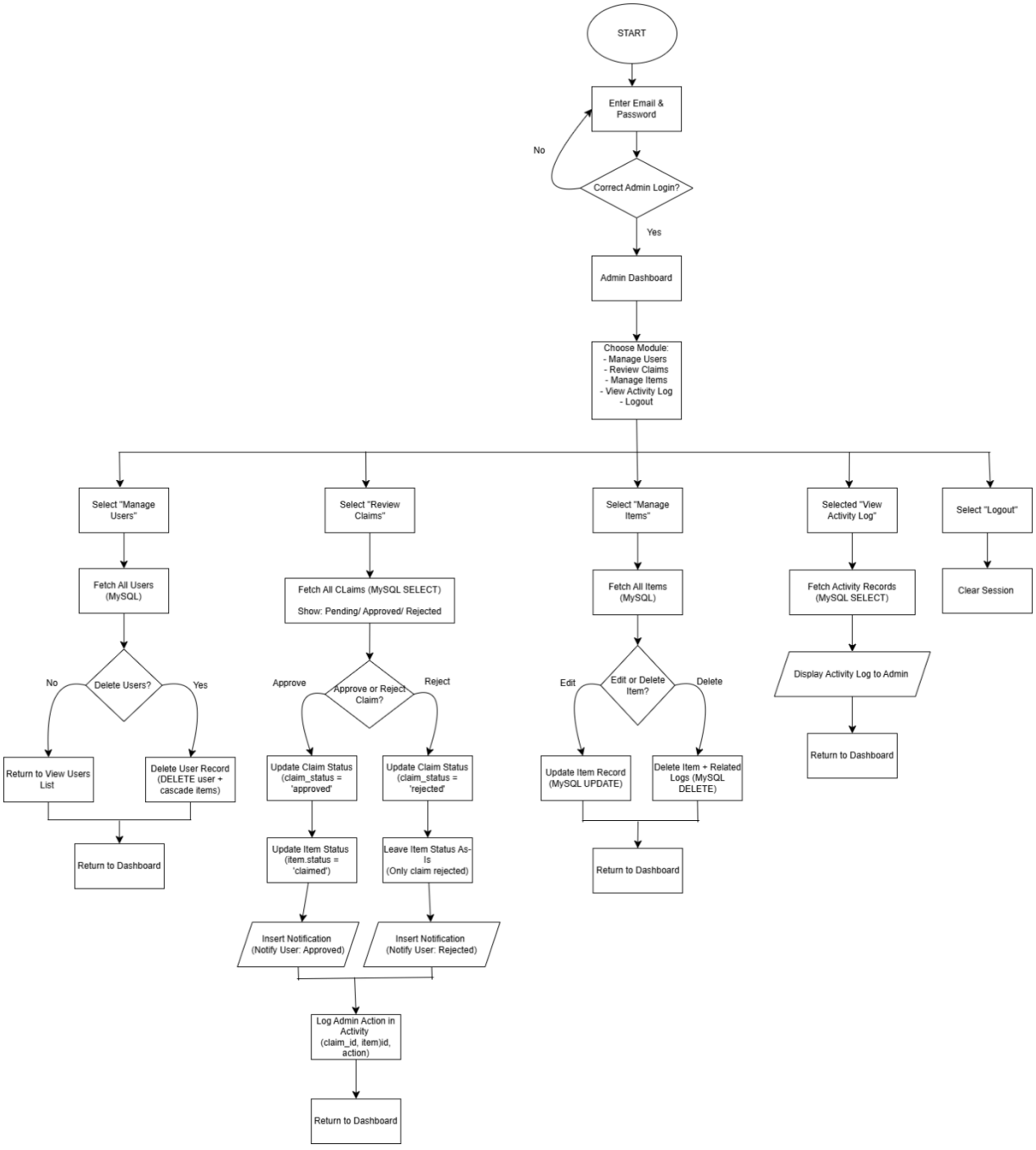

Figure 4.8 Shows Flowchart of Admin in LostLink

Explanation:

1. **Manage Users Flow**

   When the admin selects “Manage Users” the admin is able to view all users, filter them by role and name by using keyword search, and able to deletes accounts when necessary. The system manages the deletion and cascades removal of related data. This workflow maintains data cleanliness of the database.

2. **Review Claims Flow**

   When the admin selects “Review Claims”, this is where the admin act as important role whether to approve or reject the claim and trigger updates to the item and claims status with generating notifications and logging the action. The admin must act professionally and see carefully whether it is the right ownership to the item.

3. **Manage Items Flow**

   When the admin selects “Manage Items”, Admin is able to view full list of the item and can use the function od edit or delete the item. The system then updates the database accordingly. This workflow supports data consistency and ensures proper maintenance of item records within LostLink.

4. **Activity Log Flow**

   As admin of the system, LostLink, the admin has access to activity log, where all user and admin actions are recorded. The system retrieves and displays log entries, helping admin to monitor the system usage and detect any suspicious and irregular action occurring in the system. This ensures accountability and transparency in the system.

4.2 System Interface and Features

There were five sprints that were outlined In Chapter 3, the LostLink web-based system was successfully deployed in a testing environment using Flask (Python), MySQL Workbench, and HTML/CSS/JS.

Below are the core functionality and user experience.

Figure No.	Interface	Description
Figure 4.9 and Figure 4.10	Login and Registration Page	Enables students and staff of UTP to access the system by using UTP credentials. Error message is received for invalid input.
Figure 4.12	Home Page	Main page of the system where contains the function of report, search, view, and logout.
Figure 4.13	Report Item Page	Enables users to submit items that is lost or found with information of the item and photo uploads. Data is automatically kept in the database.
Figure 4.14	Search and Filter Page	Enables users to effectively find items by filtering by name of item, date, category, or keyword.
Figure 4.15	Claim Page	Displays the item's details along with security question form to confirm ownership to avoid theft that can occur.
Figure 4.16	Admin Dashboard	Enables the administrator to handle reports, approve or reject claims, and ensure there are no inappropriate action in system.

Table 4.4 Shows System Interface and its description

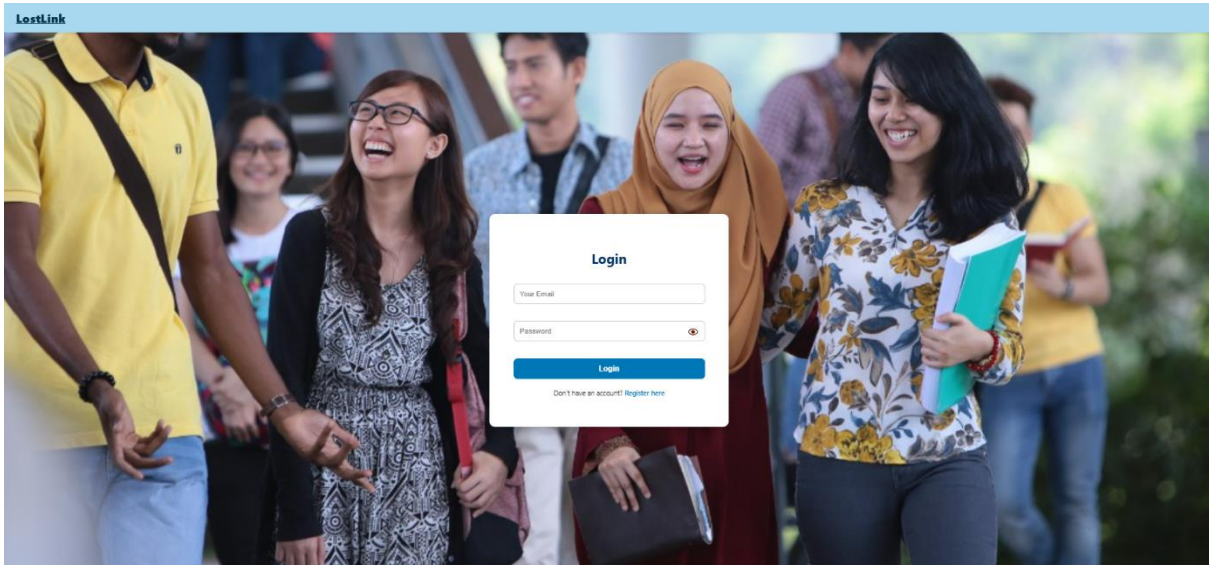

Figure 4.9 Login Page

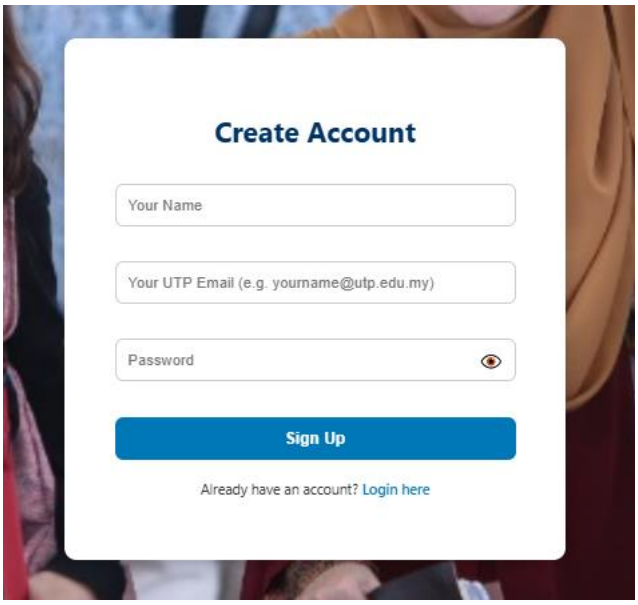

Figure 4.10 Registration Page

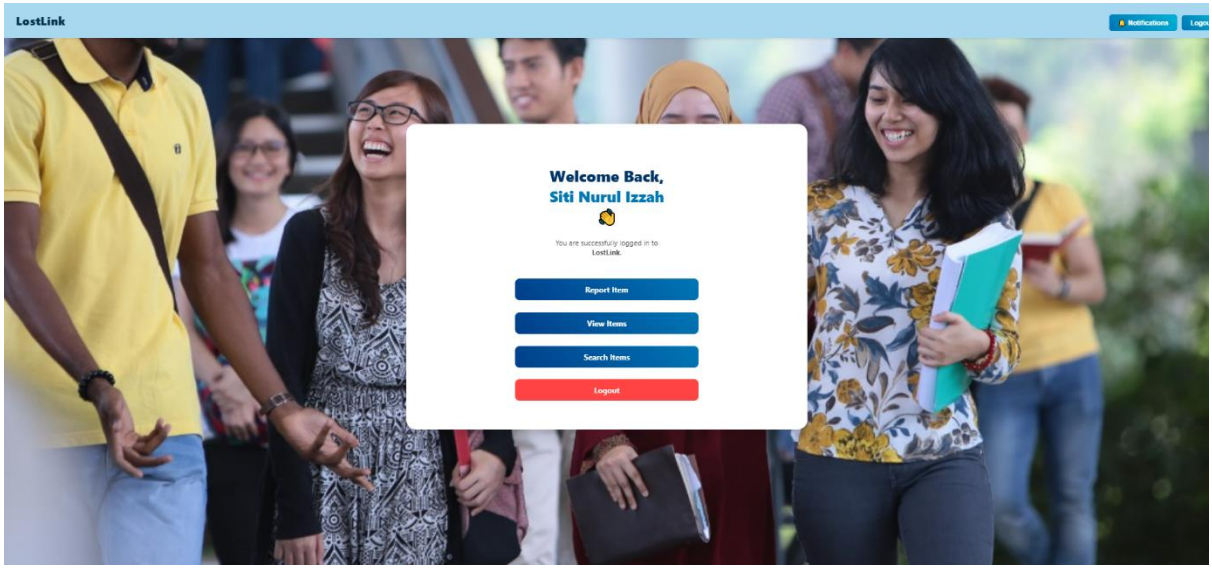

Figure 4.11 Home Page

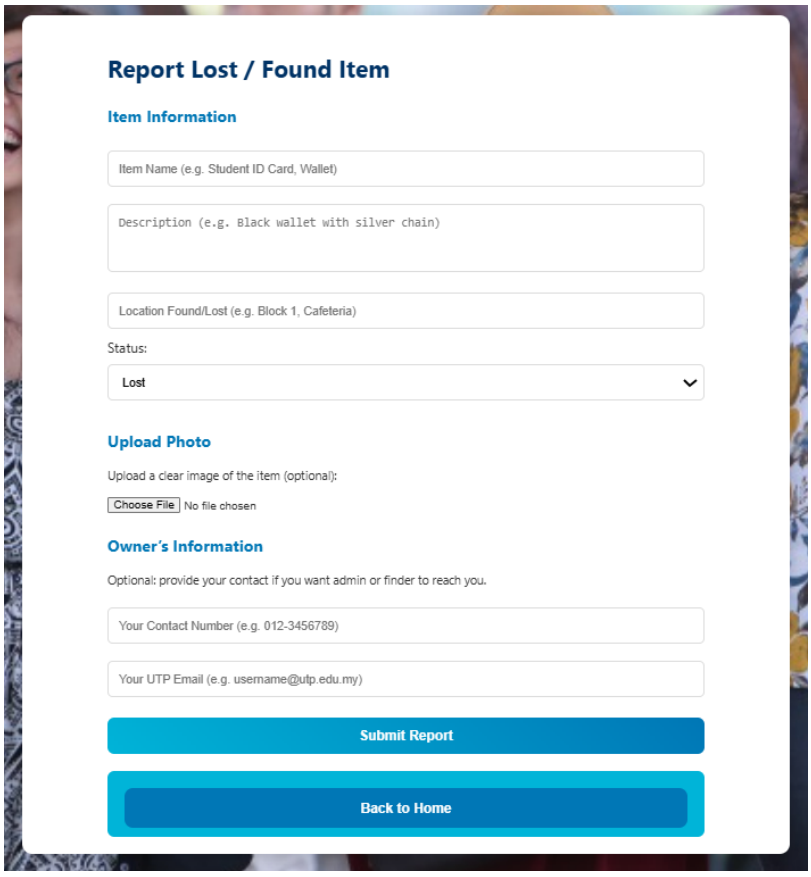

Figure 4.12 Report Lost

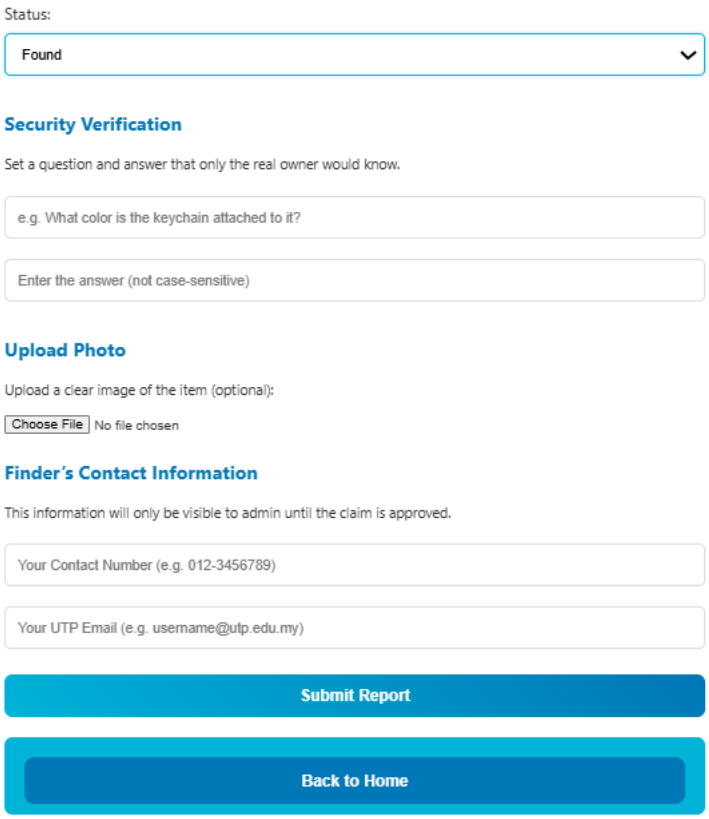

Figure 4.13 Report Found Item

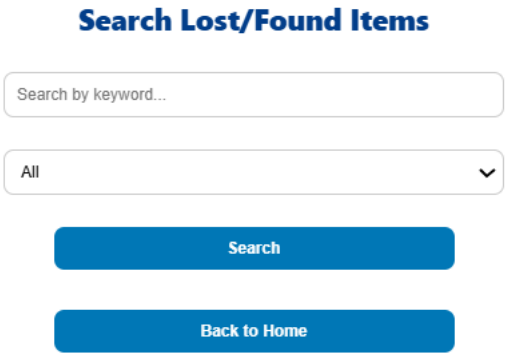

Figure 4.14 Search Items

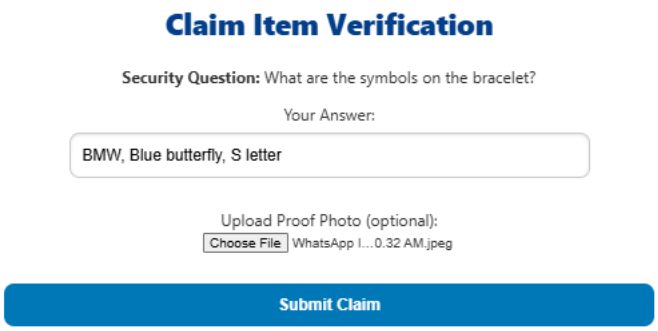

Figure 4.15 Claim Page

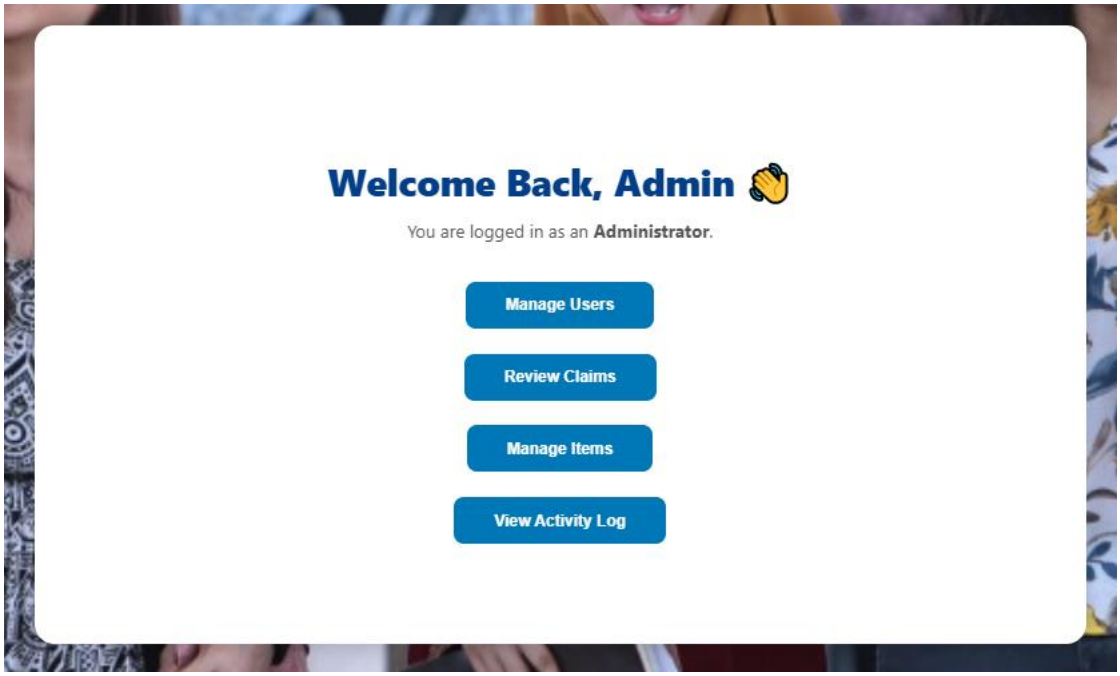

Figure 4.16 Admin Dashboard

Observation:

The user interface follows a minimalist layout that allows a responsive and consistent for user experience to use. While going through many testing, feedback was received to make improvement on the design and function from user's perspective. Refer to Appendix B for other interfaces.

4.3 Functional Testing and Bug Analysis

4.3.1 Functional Testing Overview

To ensure full functionality, the system underwent several testing cycles to cover all functions which are login, reporting, viewing, searching, claiming, and admin verification.

The main objectives were to verify that:

1. All core functionals which are User, Report, Claim, and Admin, able to operate correctly.
2. User interactions with the database are reflected accurately.
3. Every functional requirement specified in the project scope is met by the system.

4.3.2 Test Environment

Component	**Specification**
Frontend	HTML, CSS
Backend	Python (Flask Framework)
Database	MySQL Workbench
IDE / Tools	Visual Studio Code
Browser	Google Chrome
Testing Method	Manual Black-Box Testing

Table 4.5 Shows Test Environment

The testing of the LostLink system is being done out in a defined environment using specific tools and technologies to ensure system reliability and performance. HTML

and CSS were used in the frontend development process to create a responsive and user-friendly interface such as login pages, report item, claim verification, and admin dashboard, by providing the structure and visual design for all user interfaces. As for backend, Python and the Flask framework were used to create the system's backend which handled the server-side logic, routing, and communication between the user interface and the database. Example of this is login authentication, item reporting, claim verification with security questions, and admin approvals whether approved or rejected. The database management system was MySQL Workbench with the function of storing essential data such as user accounts, reported items, and claim information in efficient and secure way. Visual Studio Code (VS Code) were chosen as platform for LostLink's development and testing. The reason for this is that Visual Studio Code (VS Code) is compatible with Flask and MySQL and has built-in debugging capabilities. The system was developed and tested through Google Chrome to guarantee cross-browser compatibility and responsive performance. Testing was conducted using a manual black-box approach, with the focus on verifying that each feature produced the correct output in response to user actions without inspecting the internal source code. With this setting, it able to offer a reliable and practical configuration for assessing the system's overall usability, functionality, and overall effectiveness.

4.3.3 Functional Testing

No.	Function Tested	Test Description	Expected Result
F1	**User Registration**	New users fill in their information - Name - Matric ID - UTP email - Password	New user account is created successfully and stored in the database.

F3	**User Login**	User enters valid credentials	User is authenticated, session is created, and user is redirected to the Home Page.
F4	**Report Item**	User reports a lost/found item - Description - Upload Photo	Item details and photos successfully stored in the database.
F5	**Search Item**	Users search for items using keywords or filters.	System displays relevant items by giving input in keyword or selected filters.
F6	**Claim Item**	User submits claim with correct security answer	System gives security questions. System validates the security answer and proof photo. The claim is then marked as "pending" for admin approval.
F7	**Edit Item**	User able to edit description or photo from previous reported item.	Item description or photo is updated successfully in the database. It will show in view items.
F8	**Delete Item**	User deletes reported item.	Will be removed from the database and will not appear in the items list.
F9	**Admin Login**	Admin enters valid email and password.	Admin is authenticated, session is created, and the system shows straight to the Admin Dashbaord.
F10	**Admin Management**	Admin reviews claim, approves or rejects, view users that is in the system, and manage reports.	Able to manage users, review claims, manage items, and view activity logs.

F2	**Logout**	User/Admin click "Logout"	User/Admin session ended and system goes straight to the login page.

Table 4.6 Shows Functional Testing

4.3.4 Bug Analysis

Bug ID	Description of Issue	Cause Identified	Fix Applied	Status
B1	MySQLdb.OperationalError: Unknown database 'lostlink'	Database was not created before running app.	Executed CREATE DATABASE lostlink; in SQL.	Fixed
B2	TemplateNotFound: report.html was not found	Missing template file in /template/ directory.	Added correct report.html in Flask template folder.	Fixed
B3	Uploaded images did not display.	Wrong static path configuration.	Images were moved to /static.uploads/ and url_for() was changed.	Fixed
B4	Admin privileges did not work.	Missing is_admin column in user's table.	Added ALTER TABLE users ADD COLUMN is_admin TINYINT(1)	Fixed
B5	Dark overlay issue on homepage	Error in CSS with overlay styling	Adjusted CSS to remove unnecessary dark background.	Fixed

B6	“1” displayed in edit form.	Incorrect index reference in edit_item.html	Updated references to Jinja variables.	Fixed
B7	Clicked logout but user was redirected to home instead of login page	URL_for() has the wrong endpoint Wrongly redirect home instead of login page.	From URL for (‘home’) to URL for (‘login’) route.	Fixed
B8	Flash Messages (“Access denied”) appeared on claim page even after new action.	Global flashes from base.html showed on every template.	Hidden flash-wrap in claim_iem.html.	Fixed
B9	Claim button appeared for lost item (should only be applied on found item).	Conditional logic does not restrict status.	Jinja2 condition has been updated to only display the button if status ==”found”	Fixed
B10	Approved items did not appear in “Approved” tab in admin page of manage claim.	SQL query filtered only for pending records.	Changing the query in SQL to include pending, approved, and rejected.	Fixed
B11	UI card layout too narrow in Manage Claims.	CSS max-width restriction.	Adjusted the .claims-wrapper width and centering.	Fixed

Table 4.7 Shows Bug Analysis

4.4 System Test Summary

The LostLink system went through a full cycle of testing to ensure and confirm that each module that is in sprint of agile is being performed according to the system objectives defined in Chapter 1. Every feature such as registration, login, reporting, searching, claiming, and admin management was tested numerous times to ensure there is enhancement over time.

The test confirmed that the system successfully handled all essential functions such as data submission, retrieval, and validation. Form validation errors and database connectivity problems were among the minor faults found during the early rounds of testing and fixed. After implementing these fixes, the system achieved stable operation across multiple browsers such as Google Chrome and Microsoft Edge.

To sum up, the results explains that LostLink is functionally complete, stable, and ready for real-user evaluation.

4.5 Usability and Feedback Evaluation

In this section, this will show real users which are students and supervisors, their interactions with LostLink, and how their feedback helped u evaluate its usability, clarity, and performance. Therefore, the purpose of this section is to evaluate the user experience (UX) and usability of LostLink by all means the ease of use in navigating, report, search, and claim items.

The total of participants for testing is 10 users which consists of 9 students and 1 supervisor (lecturer). All users are able to test 2 types of roles in the system which are normal users that can view function of report, claim and search, and admin role that can view users, manage claim, and manage item. The testing method is hands-on live session where users interacted directly with the system and filled out a short post-test feedback form.

Due to time constraints, usability testing was conducted using a video-based evaluation method. The total of ten participants consists of nine students and one supervisor who is a lecturer). Following the demonstration of the system, the participants answered scale questions from strongly disagree to strongly agree and

open-ended questions on Google Form. This approach enabled speed collection of feedback on usability perceptions without requiring physical system access.

Criteria	**Average Score /5**	**Interpretation**
Ease of Navigation	4.71	Users found the navigation clear, structured, and easy to follow. When switching between pages, there is no confusion.
Layout Clarity	4.78	The UI is really clear and easy to use. Users stated the user interface was clear and easy to use.
Response Time	4.50	The system is responsive and loads fast. Users rarely experienced lag during interactions.
Claim Verification Trust	5.57	The security question and proof requirement made users feel secure and reliable.
Overall Satisfaction	5.00	All users gave the system a perfect user-friendly rating, showing a very high level of satisfaction and acceptability.

Table 4.8 Shows the Evaluation

4.6 User Testing Results and Analysis

In this section, it's going to be focusing on evaluating the usability, clarity, and overall effectiveness of the developed LostLink system, real users from the Universiti Teknologi PETRONAS (UTP) community consists of students, lecturer, and staff participated in a user testing session. The survey used Google form to conduct user testing acceptance and consisted of both closed-ended and open-ended questions with the purpose of gathering information about user experience, system satisfaction, and general expectations for a centralized lost and found system. The survey was successfully able to collect 14 participants consisting of students, lecturers, and staff, ensuring feedback from both normal users and admin roles in LostLink system.

The purpose of this user testing is to evaluate LostLink's general system accessibility, reporting, and claim process, feature clarity, and convenience of use in real-world lost and found situations. The results collected through the survey provide

insights into user needs, identify any usability gaps, and validate whether the system successfully addresses the problems highlighted in the problem statement. The main conclusion from the survey responses is collected and examined in the parts that follow.

Section A: Introduction

1. **Age distribution of Respondents**

   Most respondents fall within the 20-24 age group (85.7%), and others with smaller portions aged 25-29 (7.1%) and above 30 (7.1%).

   This distribution is consistent with the average demographic of undergraduate students at UTP, who are mainly between the ages of 20 and 24. Additionally, as this group comprises the majority of users who interact with lost and found events on campus, the feedback gathered is highly relevant.

2. **Role of Respondents**

   The survey shows that the majority are students with 78.6, while staff are 21.4%. The mix bought beneficial because LostLink is designed mainly for students but also staff/lecturer involvement for verification and claim approval. Responses from both groups ensure that the system satisfies the needs of both admin users participating in the lost and found process and real end users.

3. **Lost/Found Experience Among Users**

   From the survey responses, 85.7% of participants reported that they have lost or found item in UTP, while only 14.3% answered “No”. This shows that the majority of users have personal experiences issues related to misplaced belongings, demonstrating how widespread and significant the lost and found issue is on campus. The large percentage supports the necessity of LostLink by explaining that the system solves a genuine and persistent issue that UTP employees and students deal with.

Section B: Perception After Watching the Video

1. **The system interface looked simple and easy to understand.**

   Feedback for interface clarity has received positive feedback with 78.6% of respondents gave a rating of five, 21.4% gave a rating of four, and non-gave a rating of three or less. This shows that the UI design, which focused on simple form structure, pastel colors, clear layout, and consistent navigation, was successful in producing an interface that users find intuitive. Even for new users, the system is user-friendly, as seen by the lack of negative scores.

2. **The navigation flow was clear and logical.**

   Feedback on navigation flow has received 71.4% of respondents gave a rating of five, and 28.6% gave a rating of 4. User reported that the switched between pages such as from home to report, to view, to claim, and lastly notifications felt structured and predictable. This implies that workflows and button placements are in line with user expectations and facilitate clear, seamless interaction.

3. **The report and search features seemed easy to use.**

   Feedback for this feature found these core features easy to use, with 64.3% rating five and 35.7% rating four. This highlights that the main functions of LostLink such as reporting and searching items are well designed, easy to complete, and free from unnecessary steps. These positive evaluations confirm the system's capacity to efficiently assist the most important lost and found operations.

4. **The claim verification (security question) increased my trust.**

   Feedback on claim verifications gathers that the security questions were strong, with 64.3% rating five, 28.6% rating four, and only 7.1% rating three. There are no ratings for below three. This shows that adding security verification is effective in assuring users that claim approvals are handled by admin with careful examination. This decreases the possibility of fake claims and increases user confidence in using the system.

5. **The system appeared responsive and fast during the demo.**

Feedback on system being responsive and fast during demo gathers that majority of respondents with 78.6% rated five, while the remaining 21.4% gave a four rating. This supports the idea that LostLinkk is stable and minimal delay during the demo session. The system's dependability is strengthened by the consistency between measured performance outcomes and user perception.

6. **The admin interface looked manageable and clear.**

   Feedback on admin interface gathers that there is a high positive evaluation regarding the admin dashboard, with 85.7% rating five, and remaining with 14.3% for four. This implies that the dashboard's clear access to manage users, review claims, and items management functions made it simple to use even for administrators. The high ratings confirm that the admin's interface simplicity and ability to facilitate efficient moderation.

7. **How fast does the system load?**

   Feedback regarding the system load gathers that most respondents rated the system's loading speed highly with 57.1% for five ratings, and 35.7% rating it a four. Only one response (7.1%) rated it a three, and no ratings fell below that. This shows that users view LostLink as quick and responsive, which is in line with the technical performance findings, according to the testing on one of the functional assessed using Chrome Developer Tools. The majority agreement confirms that page transitions, searches, and form submissions perform smoothly, contributing to a positive user experience.

8. **Overall, LostLink seems user-friendly.**

   This testament received a 100% rating of five which means that every single respondent felt LostLink is user-friendly. This is a significant validation point that demonstrates the system's exceptional usability form a functional and design standpoint.

9. **I would personally use a system like this in UTP.**

   The number of respondents that are willing to use LostLink is significantly high with 92.9% rating five, and 7.1% rating four. This shows strong acceptance and demand for the system. Neary all of the respondents said they

would personally utilize LostLink If it were put into place, indicating a significance potential for campus acceptance

Section C: Open-Ended Feedback

1. **What did you like most about the system?**

   The majority of responders commented on LostLink for being simple, easy to use, and straightforward to navigate. Many users emphasized how organized and visually appealing the interface is with pastel colur scheme that gives the platform a contemporary, user-friendly vide. The security question feature, reporter's anonymity to ensure privacy, and the reporting and claim process's clear structure were also valued by participants. The system feels functional, easy, and well-designed, according to a number of users, which leads to comfort and trust.

2. **Which part looked confusing, or could it be improved?**

   Overall, most respondents indicated that the system was not confusing. However, a few noted minor areas for improvement:

   - Adjustment to element sizes (box)
   - Reduce the number of buttons and do upper menu instead

   However, there are a few users raised concerns or suggestions related to features that already exist in the system such as adding feature of uploading photo for claim verification, description where user can type in the description of the item in keyword, the next process after the item has been approved. This may indicate that certain features were not fully noticed during the short demo video, suggesting a need for clearer visibility or improved user guidance in future iterations.

3. **Any features you think should be added?**

   Participants suggested several possible enhancements, demonstrating their significant belief in LostLink's capacity to expand. These include:

- Menu Bar
- Enhanced categorization by type, category, or location
- More detailed verification steps for claims
- Direct communication features (linking to Microsoft Teams)

Some respondents also mentioned “none”, showed they were satisfied with the current feature set. However, there are a few suggestions that LostLink already provides. This is likely due to the limitation of the demo that may not have fully showcased these components. A more thorough and comprehensive system tour would be beneficial for upcoming usability assessments.

4. **General comments**

The general comments were full of positive as respondents frequently described LostLink as:

- Very useful
- Very good for future use
- Can solve the problem that is still happening in UTP
- The system is good and has potential to be commercialized
- Clean, secure, and well-designed

In contrast to depending on dispersed Telegram or WhatsApp village groups, a number of users expressed joy and said the system will make life easier in UTP community. Other mentioned that the system feels “future-ready” as there are comments stated with the potential to be formally adopted or commercialized.

Overall, the open-ended responses show that LostLink is well-linked by staff and students and effectively points out the current lost and found process and is well-received by students and staff.

Refer to Appendix X for the full survey responses.

4.7 Performance Evaluation

The performance of LostLink was evaluated based on system response on speed, database operation time, and stability across browsers. Google Chrome and Microsoft Edge were used on a typical laptop with a reliable Wi-Fi connection to conduct the test.

4.7.1 Test Environment

Item	**Specification**
Device	Personal Laptop: Windows 11 Home (64-bit)
Processor	11th Gen Intel(R) i7
Operating System	Window 10 Pro
Backend Framework	Python Flask (version 3.x)
Frontend Stack	HTML, CSS, JavaScript, Jinja2 templates
IDE	Visual Studio Code
Browser	Google Chrome
Database	MySQL (local server, MySQL Workbench 8.0)
Network	UTP Wi-Fi
Environment Mode	Localhost (127.0.0.1) – not remote or cloud-based deployment

Table 4.9 Shows the Test Environment (Locally)

4.7.2 What to measure

Metric	**Description**	**Tool/Method Used**
Page Load Time	The amount of time it takes for a page to fully load.	Chrome Inspector

Search Query Time	Time taken from initiating a search appear.	Stopwatch
Form Submission Time	Time to submit a "Report Item" form	Stopwatch
Claim Verification Time	Delay between claim request and confirmation message or admin update	Stopwatch
System Stability	Whether the system continues to function during prolonged testing	Observation-Based

Table 4.10 Shows Metric, Description, and Tool/Method

These metrics were chosen as they show the most common and performance-sensitive operations. The evaluation's key goal is to make sure that every process runs smoothly and that the system remains stable throughout continuous usage.

4.7.3 Performance Result

To assess LostLink's speed, stability, and effectiveness under normal operating conditions, performance testing was done. The environment that will be testing is on Google Chrome and Edge browsers to visit Windows 11 laptop running Flask (Python 3.x) with MySQL as the database.

Test Case	**Average Time (s)**	**Result**
Login Page Load	0.042 seconds	Page loads normally
Report Item Form	0.40 seconds	No error submission which is successful
Search Function	0.33 seconds	Quick retrieval of findings
Claim Verification	0.39 seconds	Claim filled and stored correctly
System Stability	-	System remained stable with no crashed

Table 4.11 Shows Performance Result of the System

Response times below two seconds were attained in every performance test, which is withing acceptable limits for a web-based application. The system's dependability was further confirmed by the fact that it stayed steady during continuous operation.

4.7.4 Discussion

The performance evaluation results show that LostLink performs efficiently under normal usage conditions. The two-second benchmark, which is frequentl used to determine acceptable web application performance, was far below all observed response times. The frontend assets, routing, and session setup are lightweight and optimized for local execution, as evidenced by the Login Page Load, which averaged 0.042 seconds, shows the fastest metric. The fast performance also reflects the efficiency of Flask's minimalistic framework and the small file sizes used in the interface.

The report Item Form and Claim Verification processes, which require direct interaction with the MySQL database, also perform well with average times of 0.40 seconds and 0.39 seconds. These numbers imply that there are no bottlenecks in the database queries, image uploads, or form validations. The performance verifies that the SQL operations, table structures, and indexes are suitable for the application's present scale.

The search Function produces results with 0.33 seconds, which proving that LostLink can efficiently retrieve and filter items even when several attributes such as keywords or categories are involved. This supports one of the system's core objectives, which is by helping users find lost items efficiently and easily without unnecessary delays.

System stability was consistent throughout the evaluation. Throughout numerous tests on Google Chrome and Microsoft Edge, there were no crashes, freezes, or strange behaviours. This result is crucial as it validates that LostLink can maintain to be dependable throughout routine usage cycles, which includes item viewing, report submission, and claim verification.

However, it is crucial to remember that the performance tests were carried out in a local environment (localhost), which has low server load, network latency, and concurrent user traffic. If the system were to be deployed on a public cloud server or used by many concurrent users, they may be changes on the performance. Future work could include stress testing, cache implementation, or cloud-based deployment to measure scalability more accurately.

Overall, the results confirm that LostLink is fast, stable, and efficient for the level of development it is in. The system offers a responsive user experience that is in line with contemporary web application standards and system meets all expected performance requirements.

4. 8 Challenges/Limitations

Deployment Limitation

During the deployment phase, the system was first set up to be hosted online during the deployment phase using PlanetScale to make it remote MySQL database and Render, for the Flask backend. However, integration issues occurred due to SSL/TLS requirements enforced by PlanetScale, which causes conflict the connection with the Flask-MySQLdb library's default on Render. The cloud connection was not able to safely establish within the project timeframe, despite several setup efforts.

As a result, Flask's development server and a local MySQL database were used to deploy the LostLink system locally. This configuration maintained full system functionality which consists of authentication, item management, and administrative controls. Withput sacrificing any project goals or functional results, the decision to go with local hosting guaranteed reliable performance and simpler debugging.

4.9 Summary

This chapter showed the outcomes of the LostLink system's implementation, testing, and evaluation. Flask and MySQL were used to successfully create the system, and HTML, CSS, and JavaScript were used to support a clear and responsive font-end interface. The implemented workflows such as reporting items, searching, viewing,

claiming, and administrative management were able to be worked as planned, demonstrating that the system meets the core requirements defined earlier in the project.

Functional testing showed that all major features operated properly and without any serious problems. User testing involving 14 participants revealed strong positive feedback, with high ratings in navigation clarity, interface simplicity, usability, and system responsive. Many users described LostLink as Benefiel and extremely relevant for the UTP community in open-ended questions. The two main minor recommendations for improvement were improving the visibility of specific features and streamlining the explanation steps in the claim procedure.

Performance evaluation also indicated that LostLink loads quickly and performs smoothly under local testing conditions with fast page transitions and no stability issues faced after repeated use. Overall, the findings show that by offering a centralized, user-friendly, and effective lost and found management system, LostLink effectively resolves the issues mentioned in the problem description and accomplishes the project's objectives.

CHAPTER 5

5.1 Conclusion

This project set out to build LostLink, a web-based Lost & Found management system with the goal to fix the issues of disorganized, manual, and slow-tem recovery procedure that is currently used on campus. The system was successfully developed using Flask, MySQL (local), HTML/CSS, Jinja2, and vanilla JavaScript, which is completely functional for both regular users and administrators.

The system demonstrated throughout development that the entire workflow, from reporting items to confirming claims, can be streamlined using a consolidated platform. Users can submit reports, upload photos, track statuses, and receive updates without needing WhatsApp groups or physical bulletin boards. With a special dashboard, administrators may also effectively monitor reports, verify claims, and keep records of user's activity logs.

Testing showed that the system works reliably in a local environment, with stable database interaction, proper access control, and smooth form validation. The project's goals were still accomplished even though compatibility and environmental problems prevented remote/cloud deployment from being completely implemented:

- Functional digital platform was built
- Core lost-and-found processes were automated
- User experience became more structured and transparent
- Admin operations became more efficient and trackable

Additionally, the systems' traceability, verification, and logging features show potential to support the operational workflows that is managed by UTP Security or Student Support units, emphasizing its relevance beyond technical implementation and into practical campus administrative.

Overall, LostLink fulfills the project objectives and effectively illustrates a workable solution for UTP's lost & found management issues.

5.2 Recommendations

The system is functional, but it can yet be improved. Future work should incorporate the following enhancement:

Although the system functions effectively in its current form, several enhancements can further strengthen its usability, reliability, and real-world adoption. One of the most crucial improvements is deploying LostLink to a cloud hosting platform, as the current solution only works in a local environment. Real-time item reporting and greater accessibility for the UTP community would be made possible by hosting the system like Render, Railway, or AWS in conjunction with a managed MySQL solution like PlanetScale or AWS RDS. The system would also benefit from the inclusion of push notifications or automated email alerts to ensure that users are promptly notified of claim updates or item status changes without having to manually check the site. To add on, integrating lightweight image recognition models as it could help the system suggest possible matches between lost and found items, increasing item identification accuracy and efficiency.

To ensure consistent and safe approvals, the claim verification workflow can be strengthened by adding more evidence questions, requesting multiple supporting photographs, or giving administrators a structured verification checklist. A mobile-friendly Progressive Web App (PWA) that allows offline caching and allows users to add the system to their home screens for faster access could be developed for LostLink to increase accessibility. Lastly, reporting and claiming procedures would be greatly streamlined by the usage of QR code features. QR codes can be attached to found items for quick guide users directly to the LostLink platform. Additionally, the system can be expanded to support operational workflows of UTP Security or Student Support units, these departments handle lost and found processes on campus. Integrating administrative features tailored for these units such as reporting tools, item intake logs, dedicated dashboard that are able to increase real-world adoption and make LostLink more aligned with existing campus procedures. These improvements would collectively enhance usability, strengthen security, and encourage broader system adoption among the UTP community.

REFERENCES

Abrahim, N., Rahman, S., & Ibrahim, H. (2024). CampusTrace: A privacy-focused lost and found management system. *Journal of Information System Applications*, 12(1), 45-54.

Alharbi, R., & Khan, S. (2021). Web application authentication and security measures: A review. *International Jornal of Computer Applications, 182(21)*, 18-25. https://doi.org/10.1007/s41870-021-00611-3

Bataineh, E., Bataineh, B., & Al Kindi, S. (2015). Design, development and usability evaluation of an outline web-based lost and found system. *International Jornal of Digital Information and Wireless Communications, 5(1)*, 75-81.

Codd, E.F. (1970). A relational model of data for large shared data banks. *Communications of the ACM*, *13(6)*, 277-387. https://doi.org/10.1145/362384.362685

Davis, F. D. (1989). Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS Quarterly, 13(3)*, 319-340. https://doi.org/10.2307/249008

Dennis, A., Wixom, B. H., & Tegarden, D. (2015). *Systems analysis and design: An object-oriented approach with UML* (5th ed.). John Wiley & Sons. https://books.google.com.my/books?1d=rbLrBgAAQBAJ

International Organization for Standardization. (2018*). ISO 9241-11: Ergonomics of human-system interaction, Usability: Definitions and concepts*. ISO. https://www.iso.org/standard/63500.html

Ismail, I. (2009). *Development of a searching module in My Lost and Found website* [Undergraduate thesis, Universiti Teknikal Malaysia Melaka].

Kumar, S., Bhatnagar, D., Gahlot, D. Gola, B. S., & Rajput, G. (2022). Web-based lost and found system. *MIT International Jornal of Computer Science and Information Technology*, 11(1), 7-10.

Neo, J., W.., & Chan, P. (2024). A comparative study on lost and found management systems in academic institutions. *IEEE Access*, 12, 230-237. https://doi.org/10.1109/WAIE63876.2024.00049.

Nielsen, J. (1994). *Usability engineering. Morgan Kaufmann*. https://books.google.com.my/books?1d=95As2OF67f0C

Sakshi, S., & Sharma, A. (2025). *Relational database performance optimization techniques. In S. Bhalerao (ed.), Proceedings of the International Conference on Recent Advancement and Modernization in Sustainable Intelligent Technologies & Applications (RAMSITA-2025)*(pp. 113-124). Atlantis Press. https://doi.org/10.2991/978-94-6463-716-8_10

Shrivastanva, R., Gupta, N., & Yadav, P. (2025). Modern lost and found platform using AI and cloud technologies. *Journal of Computer Applications and Technological Innovation*, 9(2), 188-195.

Tan, S. Y., & Chong, C. R. (2023). An effective lost and found system in university campus. *Journal of Information System and Technology Management*, 8(32), 99-112.

Tiwari, A., Gupta, S., & Singh, P. (2019). Design of Python-based lost and found website for college campus. *International Research Journal of Engineering and Technology, 6(5)*, 1-8.

Ullah, A., Xiao, H., & Barker, T. (2019). A study into the usability and security implications of text-and image-based challenge questions in the context of online examination. *Education and Information Technologies, 24(1)*, 13-29. https://doi.org/10.1007/s10639-018-9758-7

APPENDICES

Appendix A

A.1 Experience with Lost Items

Q1. Have you lost your belongings around the campus?

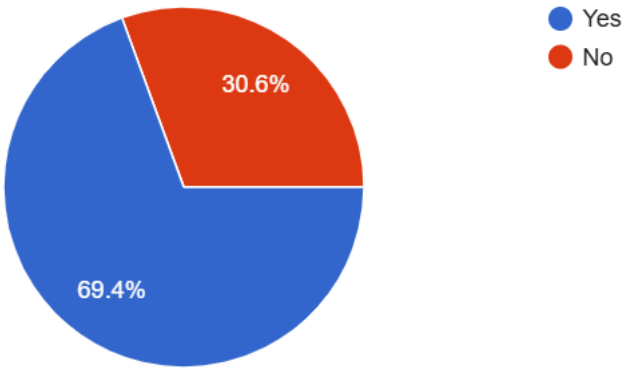

Figure A1: Survey Question 1 Result

A.2 Experience with Found Items

Q2. If yes (according to no.1), did you find your lost item?

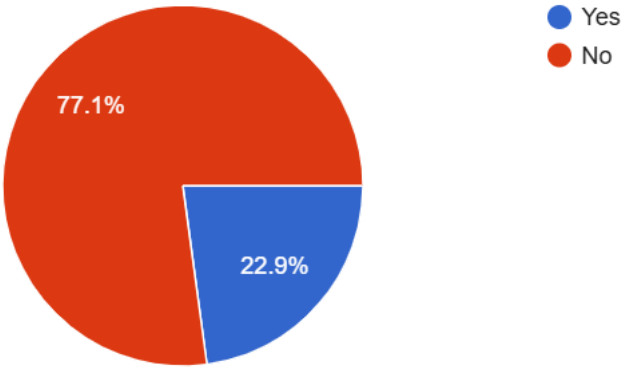

Figure A2: Survey Question 2 Result

A.3 Difficulty Returning Lost Items

Q3. Have you seen someone's else belongings being lost around the campus?

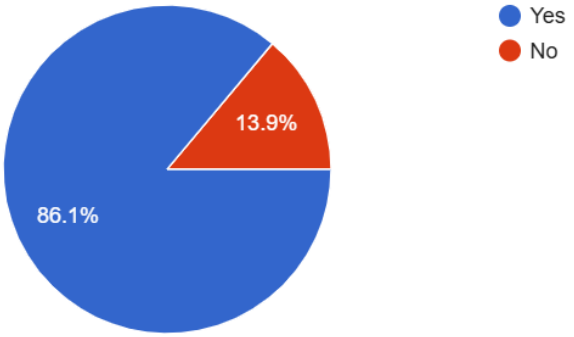

Figure A3: Survey Question 3 Result

A.4 Awareness of Existing Lost & Found Processes

Q4. If yes (according to no.3), did you manage to return the item to its owner?

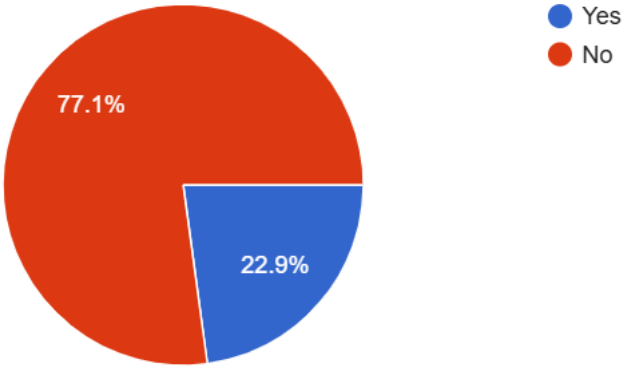

Figure A4: Survey Question 4 Result

A.5 Current Methods Used to Report or Search Items

Q5. What would you most likely do if you lost or found a personal item on campus?

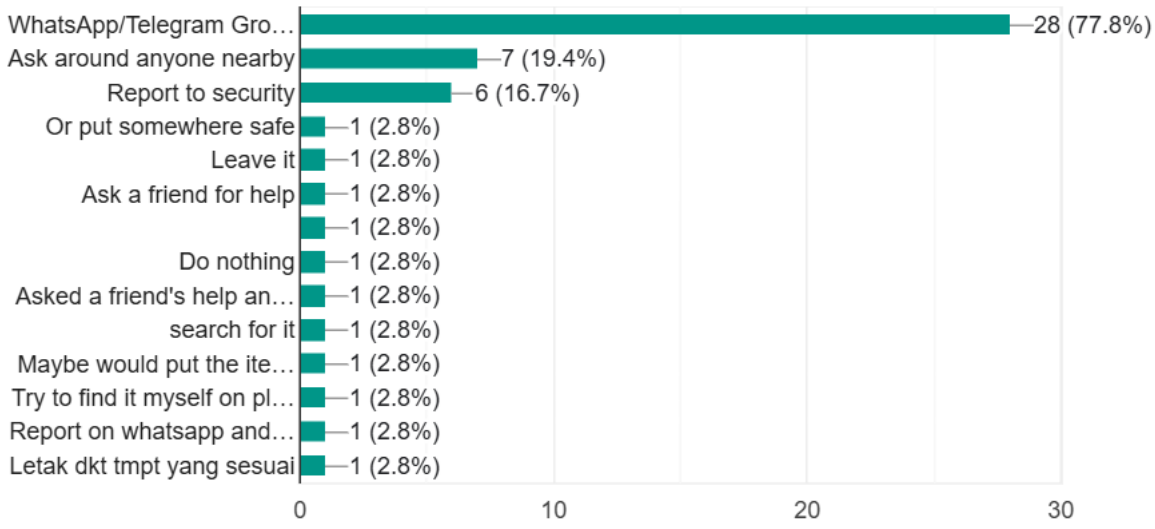

Figure A5: Survey Question 5 Result

A.6 Interest in a Digital Lost and Found System

Q6. If there were an online website for reporting lost and found items, do you think it would be helpful?

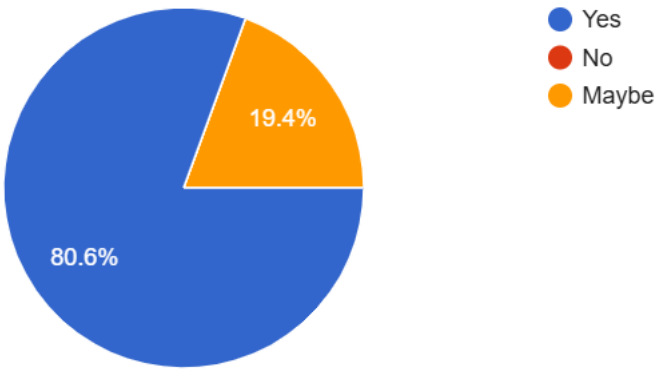

Figure A6: Survey Question 6 Result

A.7 Expected Improvement with LostLink

Q7. Would you use it the online website of lost and found reporting?

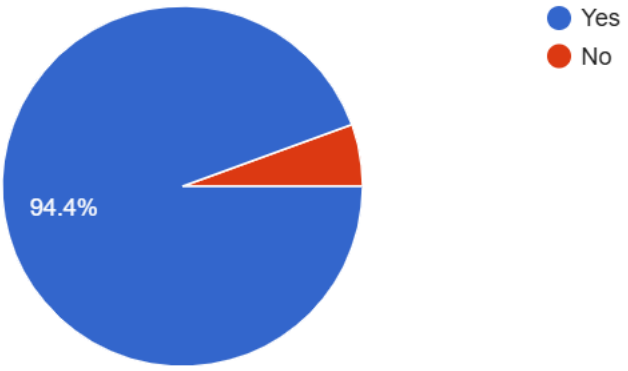

Figure A7: Survey Question 7 Result

APPENDIX B

B.1 Reported Items

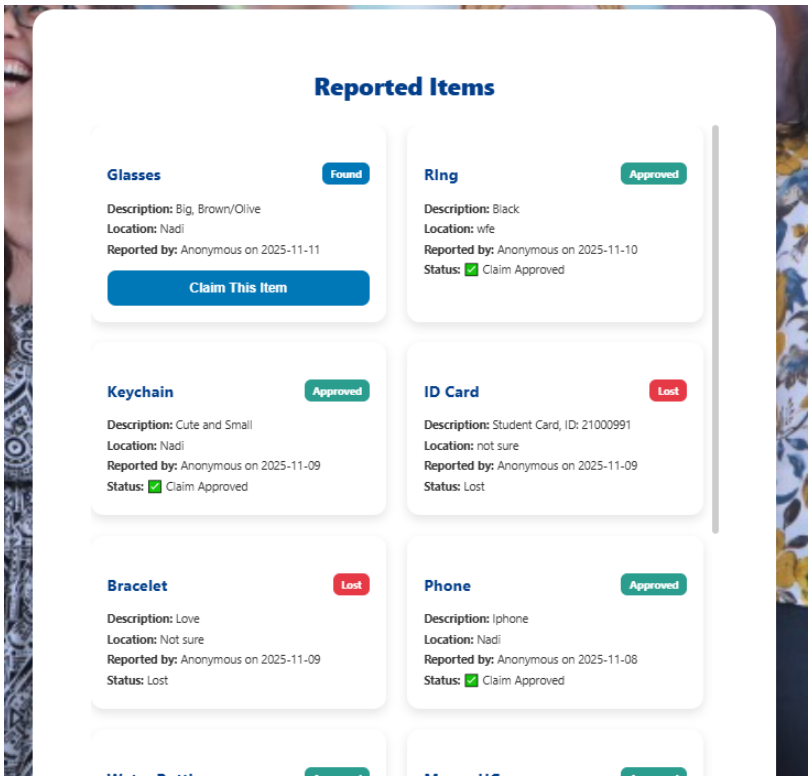

Figure B1: Reported Items

B.2 Claim Item Verification

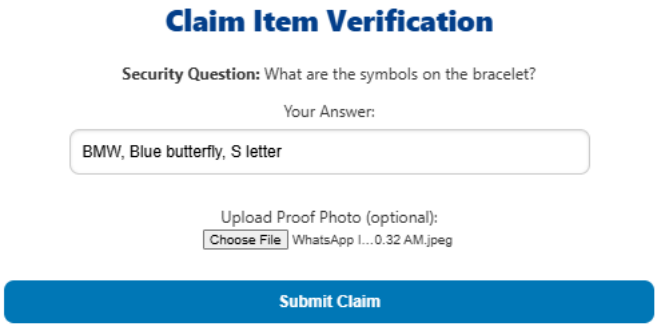

Figure B2: ClaimVerification

B.3 Item change status to “Pending”

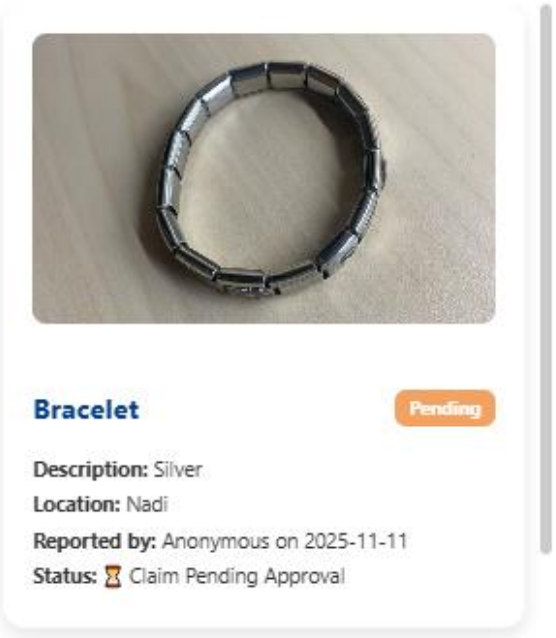

Figure B3: Status changed to “Pending” after submit claim

B.4 Admin: Manage Claims

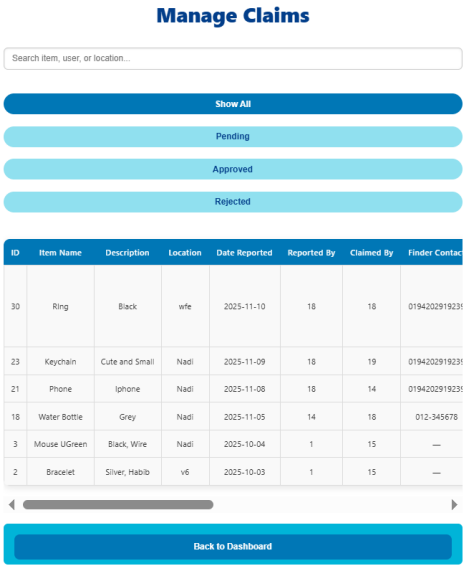

Figure B4: Manage Claims (Admin)

B.5 Admin: Approval on the Item Verification

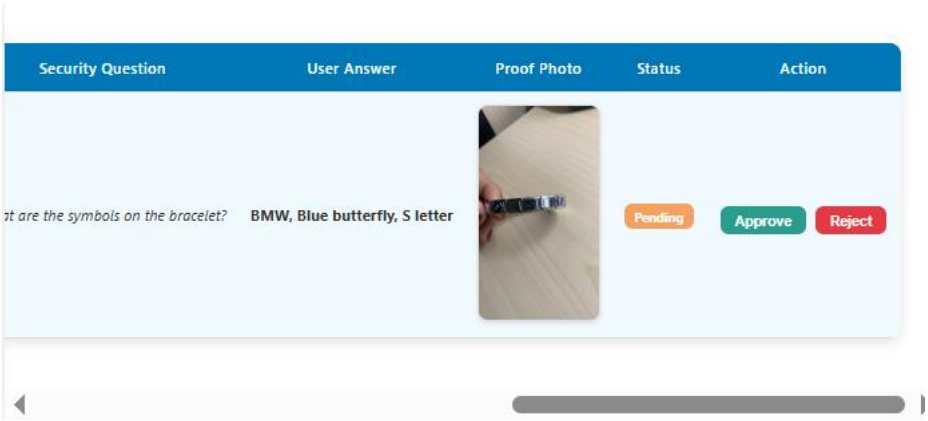

Figure B5: View ending claim (Admin)

B.6 If Approved: This notification appears

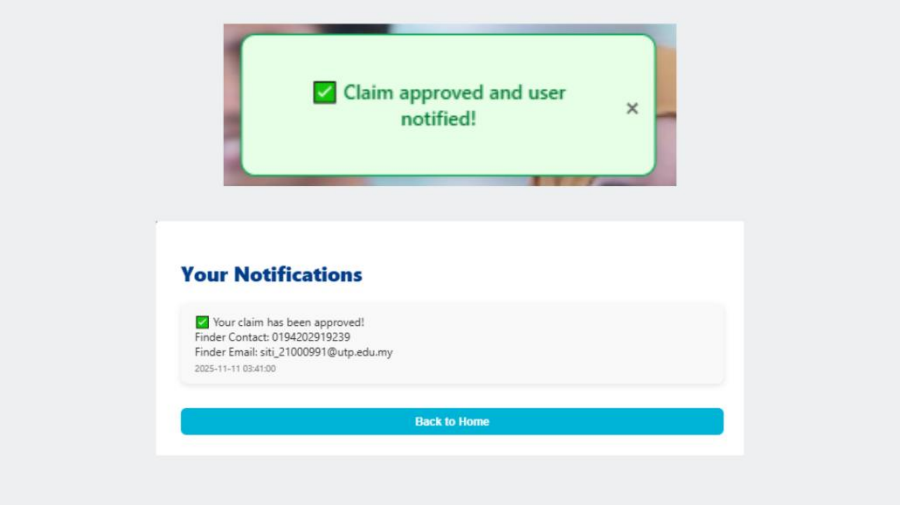

Figure B6: Approved Notifications

B.7 If rejected: This notification will receive

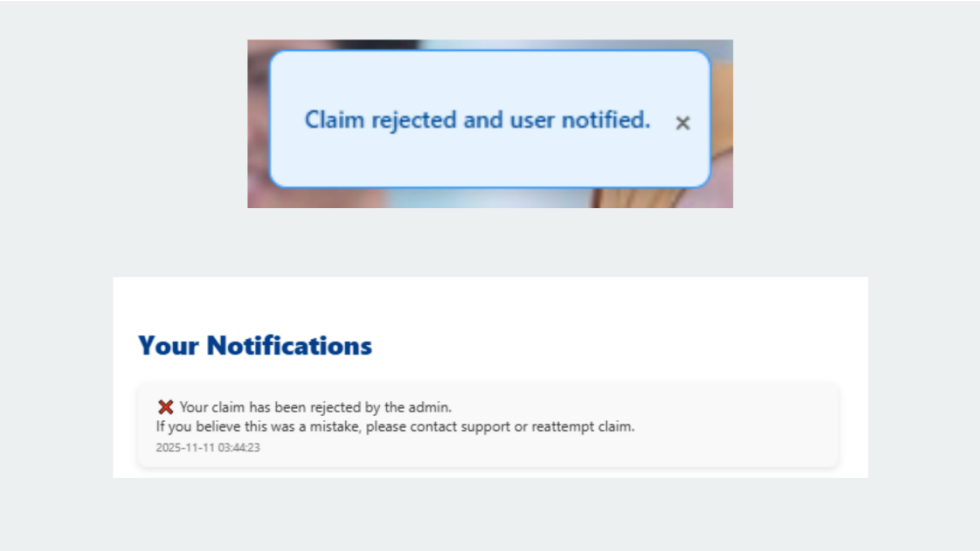

Figure B7: Rejected Notifications

B.8 Admin: Manage Users

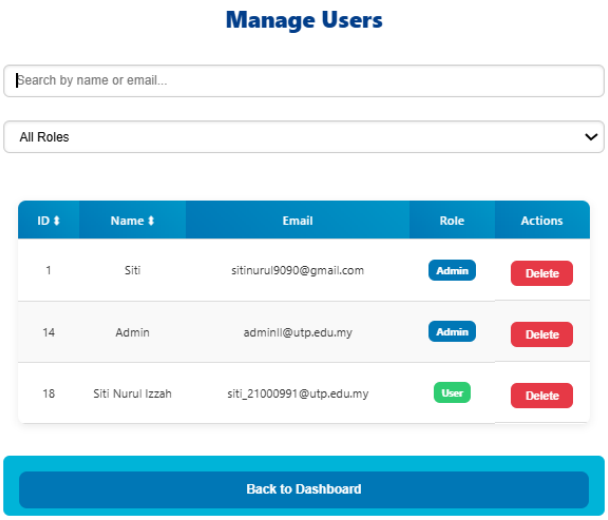

Figure B8: Manage Users (Admin)

B.9 Admin: Manage Reported Items

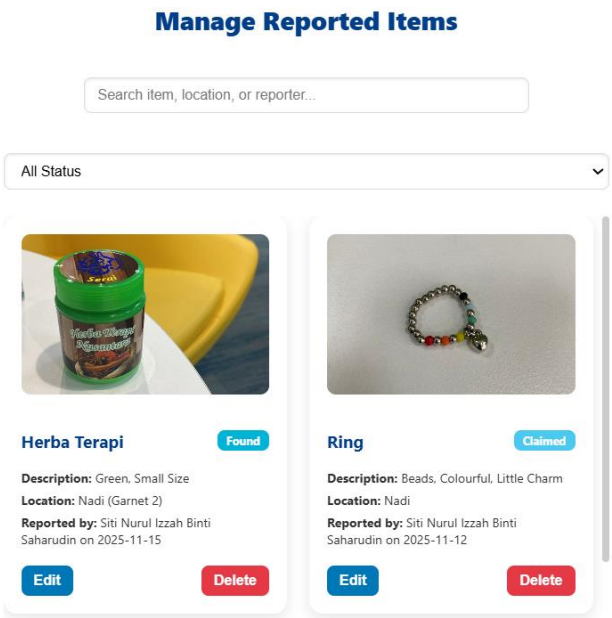

Figure B9: Manage Reported Items (Admin)

B. 10 Admin: View Activity Log

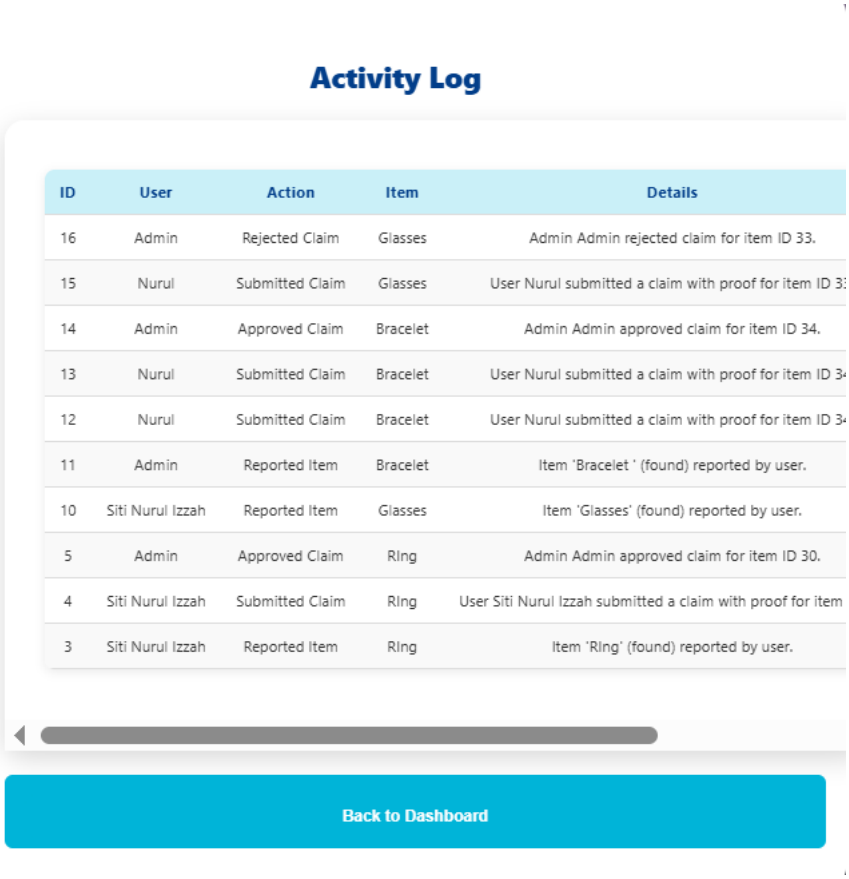

Figure B10: Activity Log (Admin)

APPENDIX C:

C.1 Age

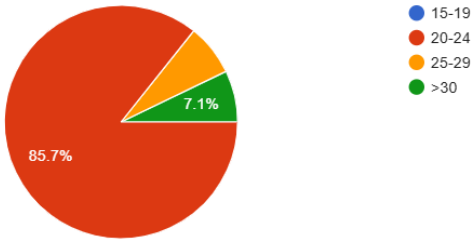

Figure C1: Age

C.2 Role

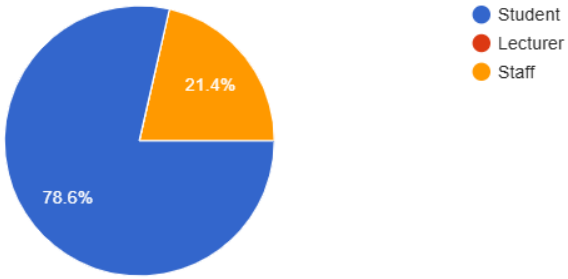

Figure C2: Role

C.3 Have you ever lost/found items in UTP?

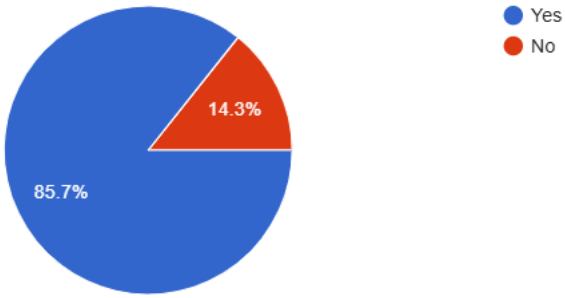

Figure C3: Experienced lost/found items in UTP

C.4 The system interface looked simple and easy to understand.

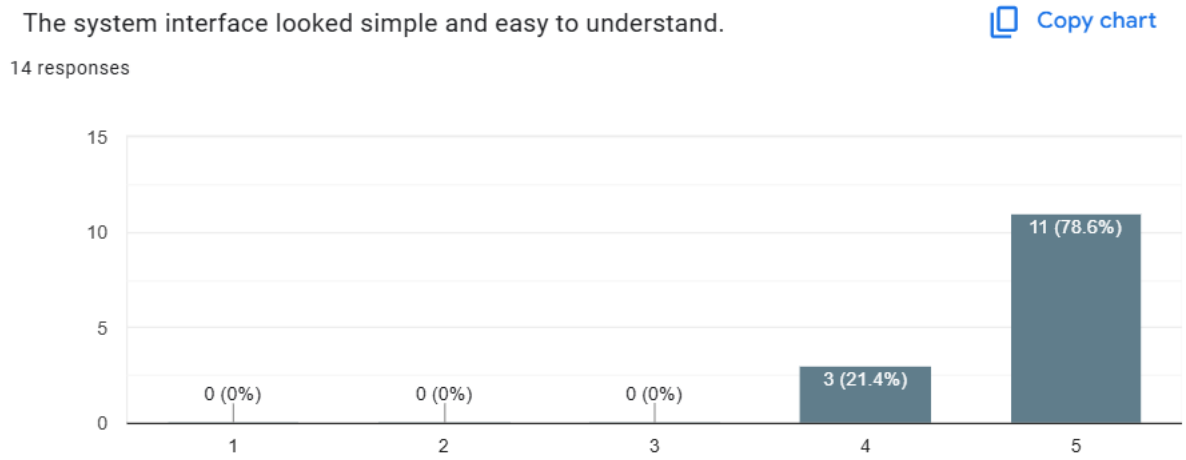

Figure C4: Simplicity of the Interface

C.5 The navigation flow was clear and logical.

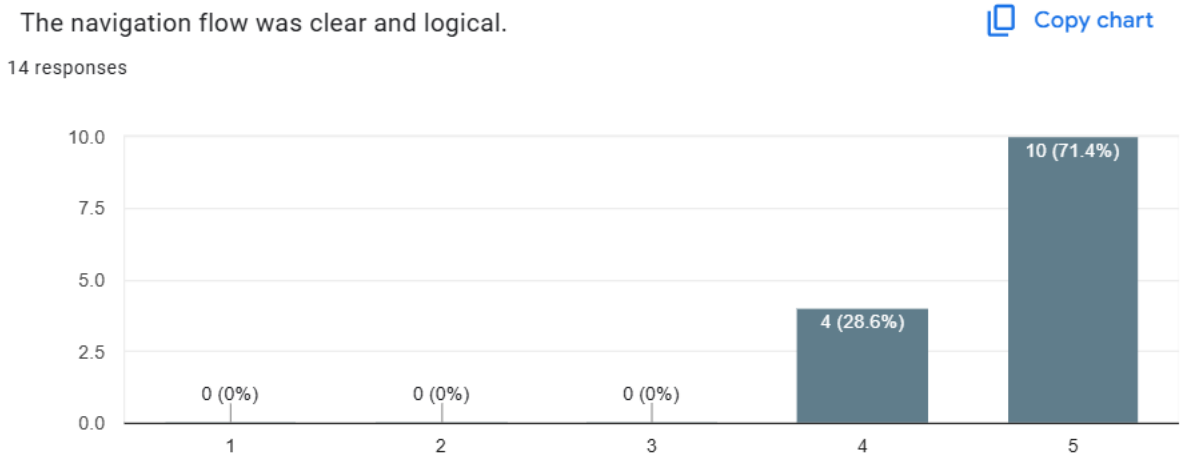

Figure C5: Navigation

C.6 The report and search features seemed easy to use.

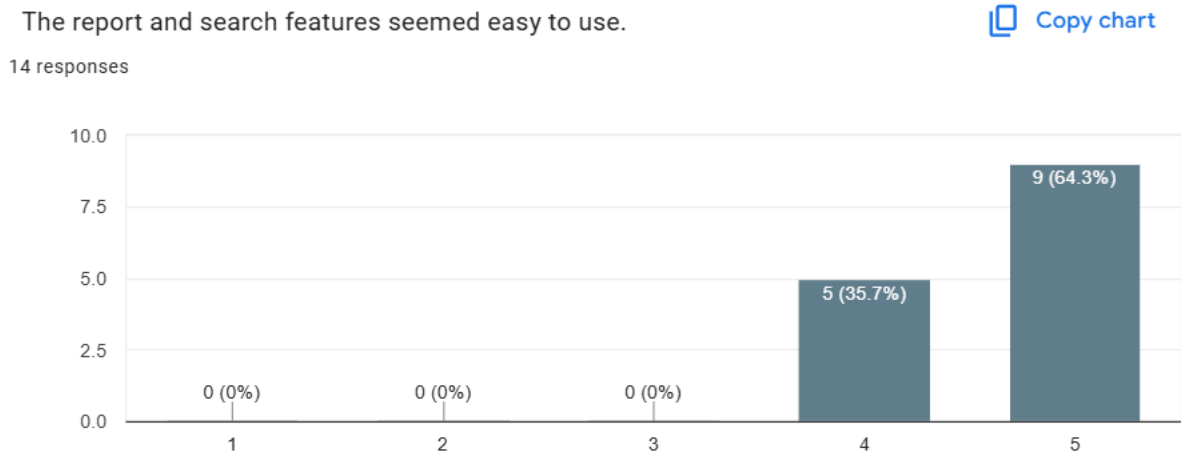

Figure C6: Report and Search Features

C.7 The claim verification (security question) increased my trust.

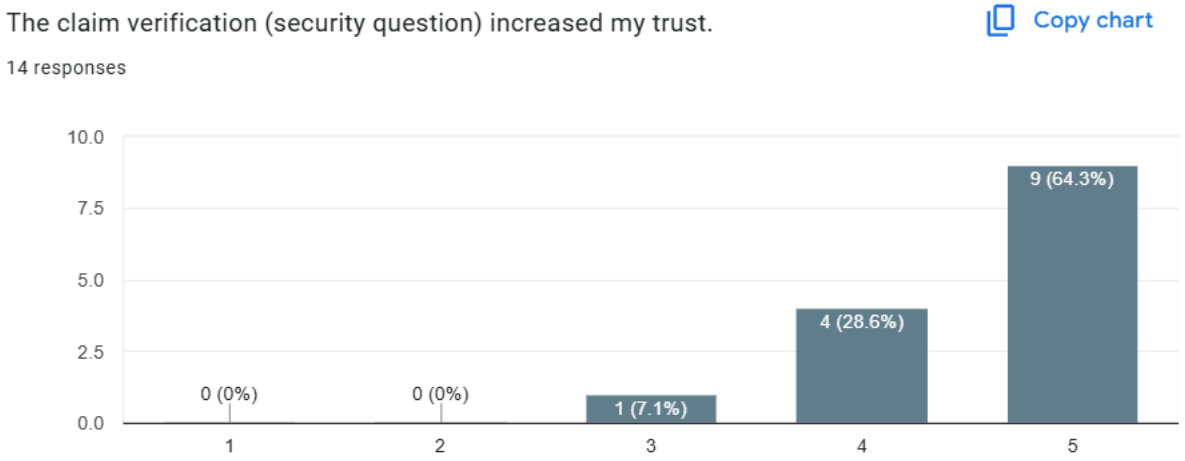

Figure C7: Security Question

C.8 The system appeared responsive and fast during the demo.

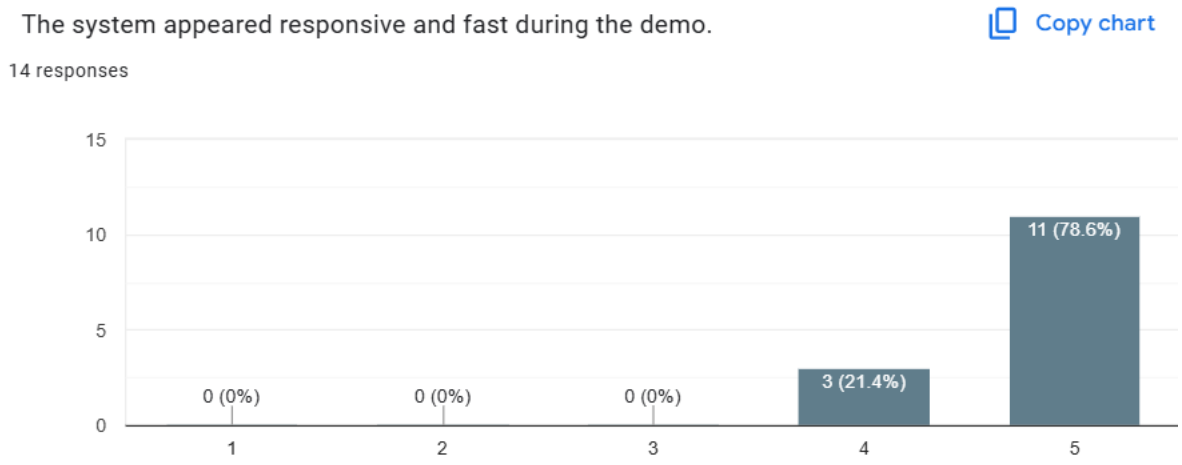

Figure C8: Responsive and Fast

C.9 The admin interface looked manageable and clear.

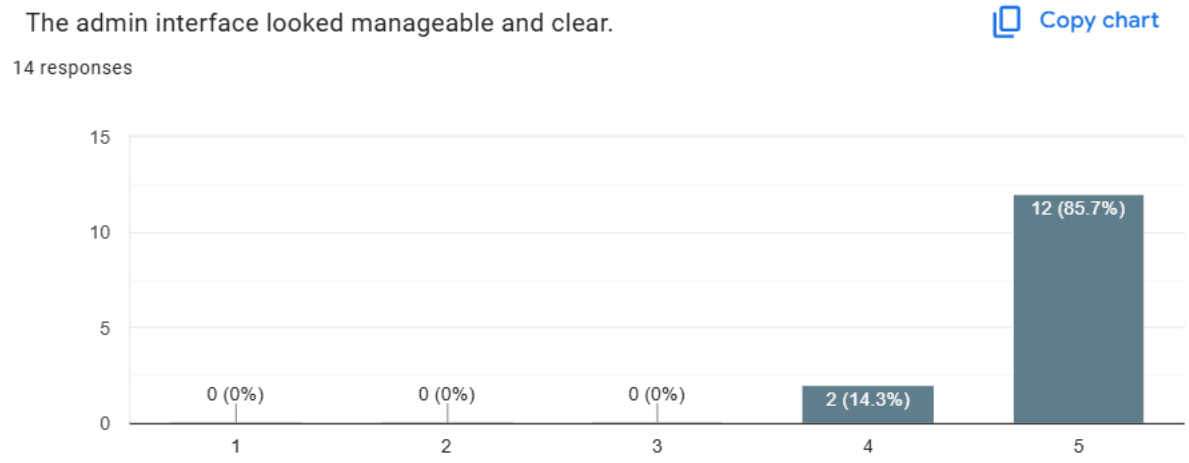

Figure C9: Admin Interface

C.10 How fast does the system load?

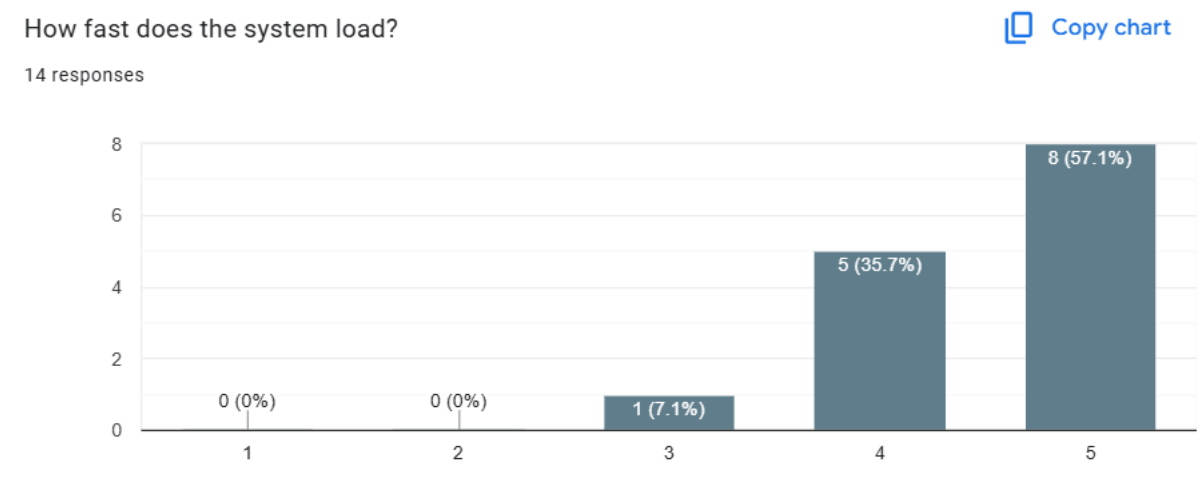

Figure C10: Speed of System Load

C.11 Overall, LostLink seems user-friendly.

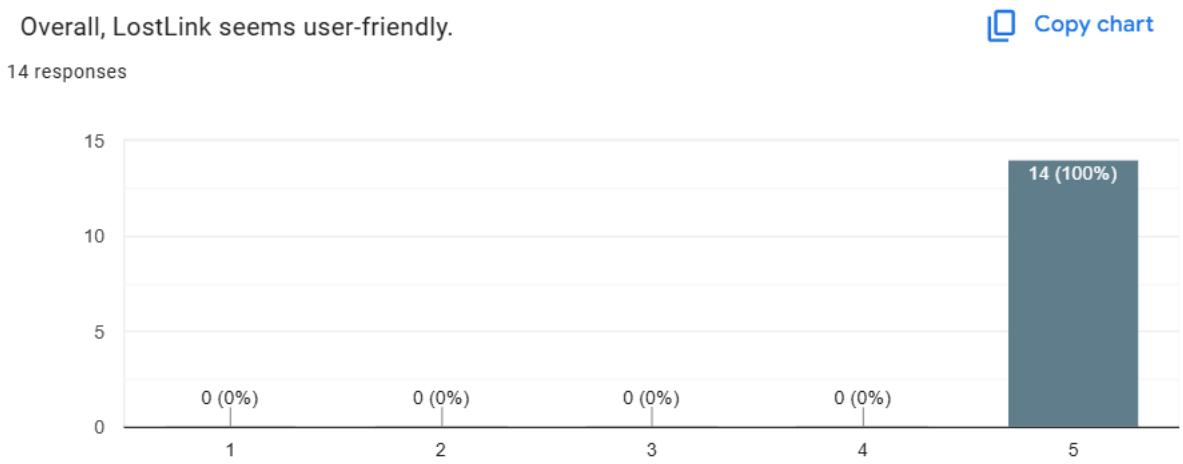

Figure C11: User-friendly

C.12 I would personally use a system like this in UTP.

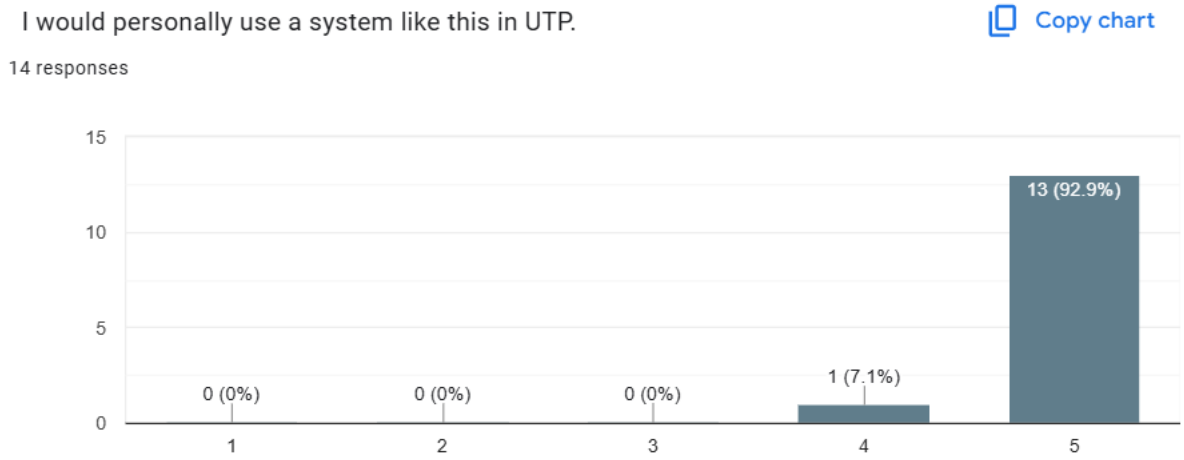

Figure C12: Use LostLink in UTP

C.13 What did you like most about the system?

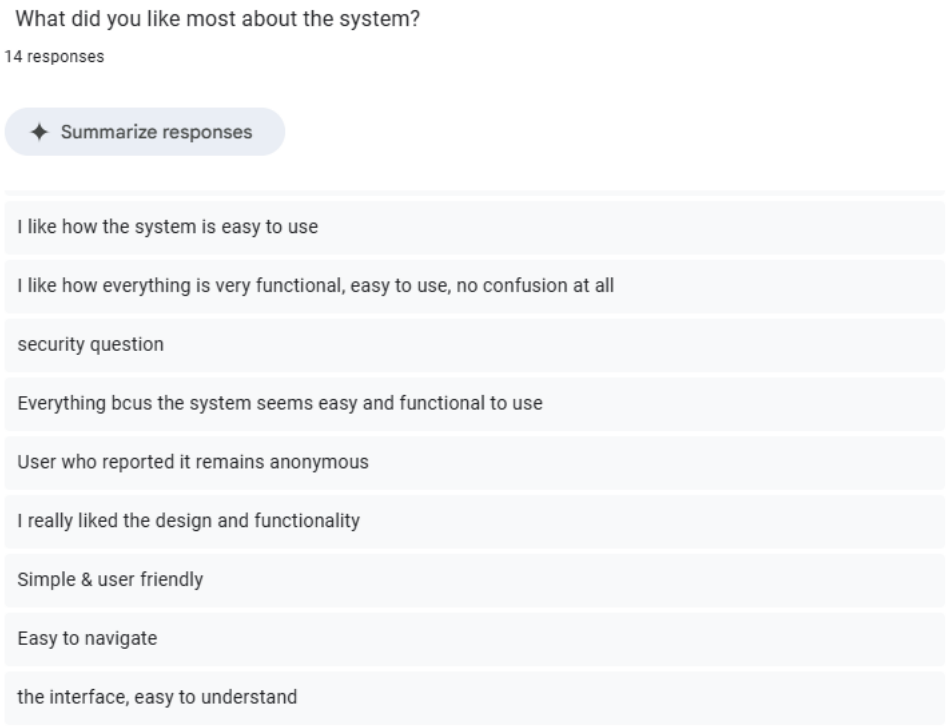

Figure C13: User's Positive Impressions of the System

C.14 Which part looked confusing or could be improved?

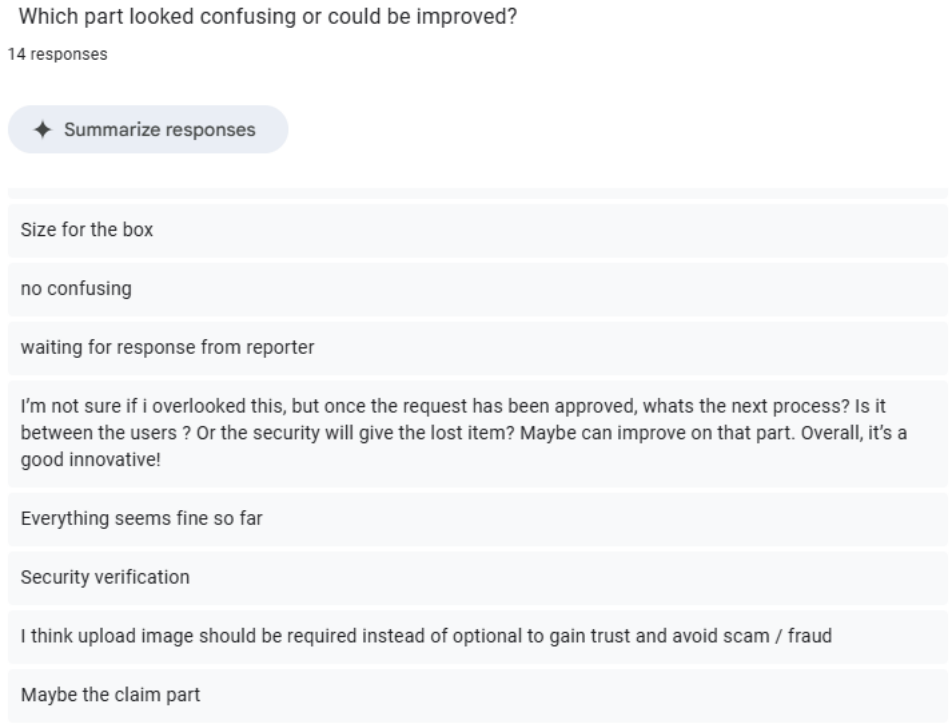

Figure C14: Improvements that can be made

C.15 Any feature you think should be added?

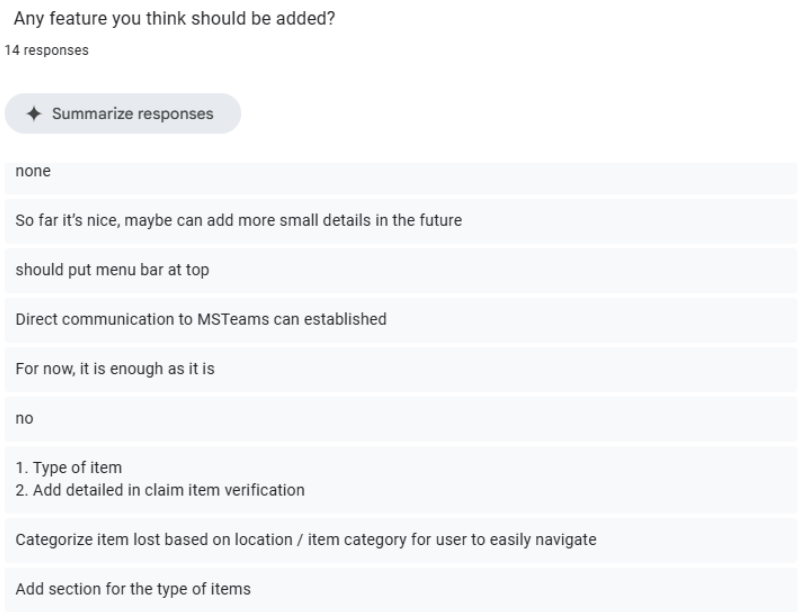

Figure C15: Features that can be added

C.16 General Comments

Figure C16: General Comments