



Department of Computer Science & Engineering

Course Code: CSE479

Course Title: Web Programming

Section: 03

Semester: Fall'25

Group:01

Title: EWU-Lost & Found Management

Course Instructor:

Nahid Hasan

Lecturer

Department of Computer Science and Engineering
East West University

Submitted By:

Name	ID
Azizul Abedin Azmi	2022-1-60-130
Tanzila Afrin	2022-1-60-045
Fayroz Tasnim Rowza	2022-3-60-011

Date of Submission: 14.12.2025

- Live Demo: <https://ewulf.rf.gd/?i=1>
- GitHub Link: <https://github.com/azizulabedinazmi/CSE479-EWU-Lost-And-Found-Management>

Table of Contents

Abstract	3
1. Introduction	3
1.1 Project Description	3
1.2 Functional Overview	3
1.3 List of Functional Requirements	4
1.4 List of Non-Functional Requirements	4
1.5 Stakeholders	4
2. Development Methodology	5
3. Technology and Environment	5
3.1 Technology Stack	5
3.2 Development Environment	6
4. System Overview	6
4.1 High-Level Description	6
4.2 Main User Roles	7
5. System Architecture	7
5.1 Layered Architecture	7
5.2 Navigation Flow	8
6. Functional User Interface Design – Public Side	9
6.1 Landing & Catalog Page (public/index.php)	9
6.2 Item Details & Claim Form (public/item.php)	10
6.3 Submit Found Item Page (public/submit.php)	10
6.4 Contact Page (public/contact.php)	11
6.5 Static Information Page (About / Terms / Privacy)	11
6.6 Error Page – Forbidden (403.php)	12
6.7 Error Page – Not Found (404.php)	12
7. Functional User Interface Design – Admin Side	12
7.1 Admin Login Page (admin/index.php)	12
7.2 Admin Dashboard Overview (admin/dashboard.php)	13
7.3 Items Management Page (admin/items.php)	13
7.4 Claims Management Page (admin/claims.php)	14
7.5 Categories Management Page (admin/categories.php)	14
7.6 Users Management Page (admin/users.php)	14
7.7 Messages Management Page (admin/messages.php)	15
7.8 Pages Management Page (admin/pages.php)	15
7.9 Settings & Maintenance Page (admin/settings.php)	16
8. Instructions for Users	16
8.1 Instructions for Public Users	16
8.2 Instructions for Admin Users	16
9. Security, Error Handling and Maintenance	16
9.1 Security Practices	16
9.2 Error Handling	17
9.3 Maintenance Mode	17
10. Limitations and Future Enhancements	17
10.1 Limitations	17
10.2 Future Enhancements	17
11. Conclusion	17

Abstract

EWU-L&F is a web-based Lost & Found Management System developed primarily for the East West University campus, but adaptable for similar organizations. It provides a central online catalog where users can browse lost and found items, view item details, submit found item reports, and request to claim items that may belong to them. The system also includes an admin panel where authorized staff can manage items, claims, categories, users, static pages, contact messages, and site-wide settings such as maintenance mode.

Built with HTML5, CSS3, JavaScript, PHP (mysqli) and MySQL (XAMPP), EWU-L&F focuses on a clean, responsive interface with theme support, custom 403/404 pages, and basic security practices such as hashed passwords and protected admin routes. This project allowed us to apply software engineering concepts like modular design, separation of concerns, and user-centered design to solve a real-world problem in a simple and reusable way.

1. Introduction

In a busy campus environment, items are lost and found every day, but information is often scattered across noticeboards, social media posts, or office records. This leads to confusion, duplication of effort, and many items not reaching their rightful owners. To address this, we developed EWU-L&F, a web-based lost and found management system that centralizes records and provides a structured claim workflow.

The system allows users to browse a public catalog of items, submit found item reports, and request to claim items with evidence. Admin users can manage items, claims, categories, messages, pages, and settings from a dedicated dashboard. The project demonstrates how modern web technologies can be used to build a practical, secure, and user-friendly application to support everyday campus operations.

1.1 Project Description

- **Web-based lost & found system:** EWU-L&F is a web application that stores all lost and found data in a MySQL database and exposes it via a clean HTML/CSS interface. Public users access the catalog and item details, while admins work through a protected admin panel.
- **Campus-oriented workflow:** The system is designed for use at East West University or similar institutions. A user can submit a found item, another user can search for it, and admins can process claims and update statuses, reflecting the real workflow of returning lost belongings.

1.2 Functional Overview

- **Functionality for public users, Public users can:**
 1. View the landing page and item catalog with category filters.
 2. Open item detail pages and see status badges.
 3. Submit claim requests with evidence for specific items.
 4. Submit found item reports using a form.
 5. Send messages through the contact page.
 6. Read static pages such as About, Terms, and Privacy.
- **Functionality for admin users, Admin users can:**
 1. Log in via the admin login page.
 2. View a dashboard with basic metrics.
 3. Manage items (approve, publish, hide, update status).

4. Review and process claim requests (approve, reject, fulfill).
5. Manage categories, users, static pages, and settings.
6. Read and manage contact messages.
7. Enable or disable maintenance mode with a custom message.

1.3 List of Functional Requirements

Public-side functional features:

1. View landing page with animated hero and item catalog.
2. Filter items by category and see status badges (Published, Claim Pending, Claimed).
3. Open item detail pages to see full information.
4. Submit claim requests with name, contact, evidence text, and optional file.
5. Submit found item reports with title, category, date, location, description, and photo.
6. Send messages through the contact form.
7. View About, Terms, and Privacy pages with dynamic content.
8. View custom 403 and 404 pages when access is forbidden or content is missing.

Admin-side functional features:

1. Log in through the admin login page using a secure password.
2. View an admin dashboard with counts and summary cards.
3. List, search, filter, and sort items in the Items module.
4. Approve pending items and publish them to the public catalog.
5. Change item statuses (pending, published, claim_pending, claimed).
6. Hide or unhide items from the public catalog.
7. View and manage claim requests (approve, reject, fulfill).
8. Manage item categories (add, edit, deactivate).
9. Manage user accounts (create, activate/deactivate admins).
10. View and manage messages received from the contact form.
11. Edit static page contents (home, about, contact, terms, privacy).
12. Configure global settings such as site name, organization name, and service (maintenance) status.

1.4 List of Non-Functional Requirements

1. User-friendly and responsive UI that works across devices.
2. Consistent styling with themes, animations, and readable layouts.
3. Secure admin authentication using password hashing.
4. Protection of admin routes via session and role checks.
5. Use of prepared statements to prevent SQL injection.
6. Basic validation of form inputs on the server side.
7. Safer file upload handling with type and size checks.
8. Clear success and error messages after actions.
9. Simple deployment using XAMPP and MySQL.
10. Maintainable code with modular directory structure.

1.5 Stakeholders

1. Students and staff who lose or find items on campus.
2. Administrative staff responsible for managing lost and found.
3. Developers and maintainers of the system.
4. UI/UX designers contributing to user experience.
5. Course instructor and evaluators for the project.

2. Development Methodology

For EWU-L&F, we used an iterative, prototype-driven development approach similar to the Prototype Development Model. Instead of designing everything fully up front, we built a working prototype early, tested it with sample data, and then refined features based on feedback and usability.

- **Prototype first approach:** We first created a minimal version with the public catalog, item detail view, and a basic admin login. This prototype helped us verify the core idea and UI structure before adding advanced features like claims, settings, and maintenance mode.
- **Incremental feature addition:** After the initial prototype, we added modules one by one: submit item API, claim API, messages, dynamic pages, and admin tools. Each iteration involved coding, testing on localhost, and adjusting the design for clarity and consistency.
- **User-centered improvements:** Based on feedback, we improved item cards, status badges, and form messages so users understand each step of the process. This iterative refinement led to a smoother overall user experience.

3. Technology and Environment

The EWU-L&F system uses a classic web stack that is easy to set up with XAMPP and familiar to many students. This section explains the technologies used and how the development environment is configured.

3.1 Technology Stack

3.1.1 Front-end

- **HTML5 Templates:** HTML5 is used to structure all pages in both [public/](#) and [admin/](#). It defines sections for navigation, content, and footers, into which dynamic PHP variables and loops are inserted. This keeps the logical structure separate from the styling and behavior.
- **CSS3 Styling ([assets/css/styles.css](#)):** CSS3 styles define the visual appearance, including colors, fonts, spacing, and responsiveness. The single stylesheet includes classes for grids, cards, buttons, status badges, and theme variations (light, dark, and neon). This unified styling ensures a consistent and modern look.
- **JavaScript ([assets/js/app.js](#)):** JavaScript is used to add interactivity such as theme toggling, mobile menu behavior, and closing flash messages. When users interact with UI controls, JS updates the DOM by toggling CSS classes, without needing full page reloads for minor visual changes.

3.1.2 Back-end

- **PHP 7.4+ with mysqli:** PHP acts as the server-side engine that processes forms, communicates with the database, and renders dynamic content. The code uses [mysqli](#) with prepared statements defined in [includes/db.php](#) to protect against SQL injection and handle parameters safely.
- **Apache Web Server (via XAMPP):** Apache serves PHP scripts and static assets. Within XAMPP, it provides an easy way to host the site locally using simple URLs like [http://localhost/public/](#) and [http://localhost/admin/](#). The same structure can be replicated on production servers.
- **MySQL Database ([lf_db](#)):** MySQL stores all persistent data in the [lf_db](#) database, created via phpMyAdmin. The schema is defined in [database/azmi_lf_db.sql](#), which includes tables and initial seed data so the application can start with example content.

- **Helper Tool ([tools/hash_password.php](#)):** A small helper script generates [password_hash\(\)](#) outputs for seed admin accounts. Developers run this script to produce a secure password hash, then insert it into the `users` table so the admin login uses safe credentials.

3.2 Development Environment

- **Local Setup with XAMPP:** For development, Apache and MySQL are started from the XAMPP control panel. The database [lf_db](#) is created in phpMyAdmin and populated by importing the SQL file. The project files are placed in the web root so PHP can execute them.
- **Local URL Structure:** Once configured, the landing page is available at <http://localhost/>, the public interface at <http://localhost/public/>, and the admin panel at <http://localhost/admin/>. This separation makes it clear which parts of the site are public and which are restricted.
- **Deployment to Live Hosting:** For live deployment, the same files and database are moved to a hosting provider that supports PHP and MySQL. Database credentials in [includes/db.php](#) are updated, and the site becomes accessible via a real domain, such as <https://ewulf.rf.gd>, for external users.

4. System Overview

This section presents a high-level view of how the project is organized and what each part does. The codebase is divided into logical folders that separate public pages, admin tools, APIs, shared includes, assets, and uploads.

4.1 High-Level Description

- **Clear folder separation:** The application separates public pages, admin pages, API endpoints, and common includes into different directories. This makes it easier to find specific functionality and maintain or extend the system over time.
- **Common helpers and assets:** Shared logic for database access, configuration, and rendering is placed in [includes/](#), while design resources live in [assets/](#). This avoids duplication and keeps concerns well separated.

4.1.1 [public/](#) – Public Interface

- **Main catalog and detail pages:** [index.php](#) provides the public catalog with category filters and item grid, while [item.php](#) shows the full details for a single item. Together, they form the main user experience for browsing and inspecting items.
- **User submission and contact pages:** [submit.php](#) presents a form for posting found items, and [contact.php](#) provides a form for sending messages. [about.php](#), [terms.php](#), and [privacy.php](#) display information and policies loaded from the database.
- **Custom error pages (403 and 404):** [403.php](#) and [404.php](#) are used when the user tries to access forbidden content or a non-existent page. These files present friendly error messages and links to guide users back to useful parts of the site.

4.1.2 [admin/](#) – Admin Interface

- **Authentication and dashboard:** [index.php](#) is the login page that checks credentials using [password_verify](#), and [dashboard.php](#) displays summary metrics and quick links. After logging in, admins use the sidebar to move between management sections.
- **Management pages:** [items.php](#), [claims.php](#), [categories.php](#), [users.php](#), [messages.php](#), [pages.php](#), [settings.php](#), and [account.php](#) implement the main admin features. Each page focuses on one type of data, offering filters, forms, and actions.

- **Shared layout and route protection:** [includes/header.php](#) and [footer.php](#) define the admin layout and navigation structure. [includes/auth.php](#) checks the session and role to prevent unauthorized access to admin pages.

4.1.3 [api/](#) and Common Includes

- **API endpoints:** [api/submit_item.php](#), [api/submit_claim.php](#), and [api/send_message.php](#) handle form submissions from the public side. They receive data, validate it, interact with the database, and return appropriate responses or redirects.
- **Database and helper functions:** [includes/db.php](#) defines the database connection and wrapper function [query\(\)](#) for prepared statements. [includes/helpers.php](#) holds utility functions, including reading settings, checking maintenance mode, rendering custom error pages, sending flash messages, and notifying admins.

4.1.4 Other Folders

- **Assets ([assets](#)):** The [css](#), [js](#), and [img](#) subfolders contain the styling, behavior scripts, and images used across the application. This keeps visual resources separate from core logic.
- **Tools ([tools](#)):** The [tools/hash_password.php](#) script simplifies the creation of secure hashed passwords for admin users. This is mainly a development helper.
- **Uploads ([uploads](#)):** The [uploads/](#) folder stores user-uploaded files such as item photos and claim evidence documents. Filenames and paths are recorded in the database and displayed on relevant pages.

4.2 Main User Roles

4.2.1 Public User

- **Unauthenticated access:** Public users do not need to create accounts or log in. They can access the landing page, browse items, and read public information directly.
- **Participation via forms:** These users can submit found items, claim items, and send contact messages through forms. Each action interacts with the backend APIs and stores data in the database for admin processing.

4.2.2 Admin User

- **Authenticated control:** Admin users must log in through the admin interface, where their credentials are verified. Once authenticated, they gain access to management tools and sensitive operations.
- **Full management capabilities:** Admins can control almost every aspect of the system: they approve or reject items and claims, manage categories and users, read messages, edit pages, and adjust site settings, including turning maintenance mode on or off.

5. System Architecture

The architecture of EWU-L&F follows a simple layered model commonly used in web applications. It separates presentation, application logic, and data storage to keep the system organized and easier to maintain.

5.1 Layered Architecture

5.1.1 Presentation Layer

- **Views and templates:** The presentation layer consists of HTML/PHP views in [public/](#) and [admin/](#), plus CSS and JavaScript assets. These files handle the structure and style of pages but call backend logic to fetch and display data.

- **Reusable components:** Shared components like navigation bars, headers, and footers are included in multiple pages using PHP includes. This keeps appearance consistent and reduces duplication when updating layouts.

5.1.2 Application Layer

- **System logic and workflows:** The application layer lives mainly in the PHP scripts that process requests ([items.php](#), [claims.php](#), API files). It contains rules for status transitions, validation, and what should happen when a user submits forms or clicks admin actions.
- **Session and access control:** This layer also manages sessions and guards, especially through [admin/index.php](#) and [includes/auth.php](#). It ensures that only logged-in admins can reach sensitive pages and that different flows respect the current state of items and claims.

5.1.3 Data Layer

- **Database abstraction:** The data layer is built around the MySQL database and the wrapper defined in [includes/db.php](#). The [query\(\)](#) function uses prepared statements and parameter binding, allowing all CRUD operations to be performed safely.
- **Persistence of core entities:** All important entities users, items, categories, claims, messages, pages, and settings are stored in relational tables. This provides a consistent and reliable foundation for the application's features.

5.2 Navigation Flow

5.2.1 Public Navigation

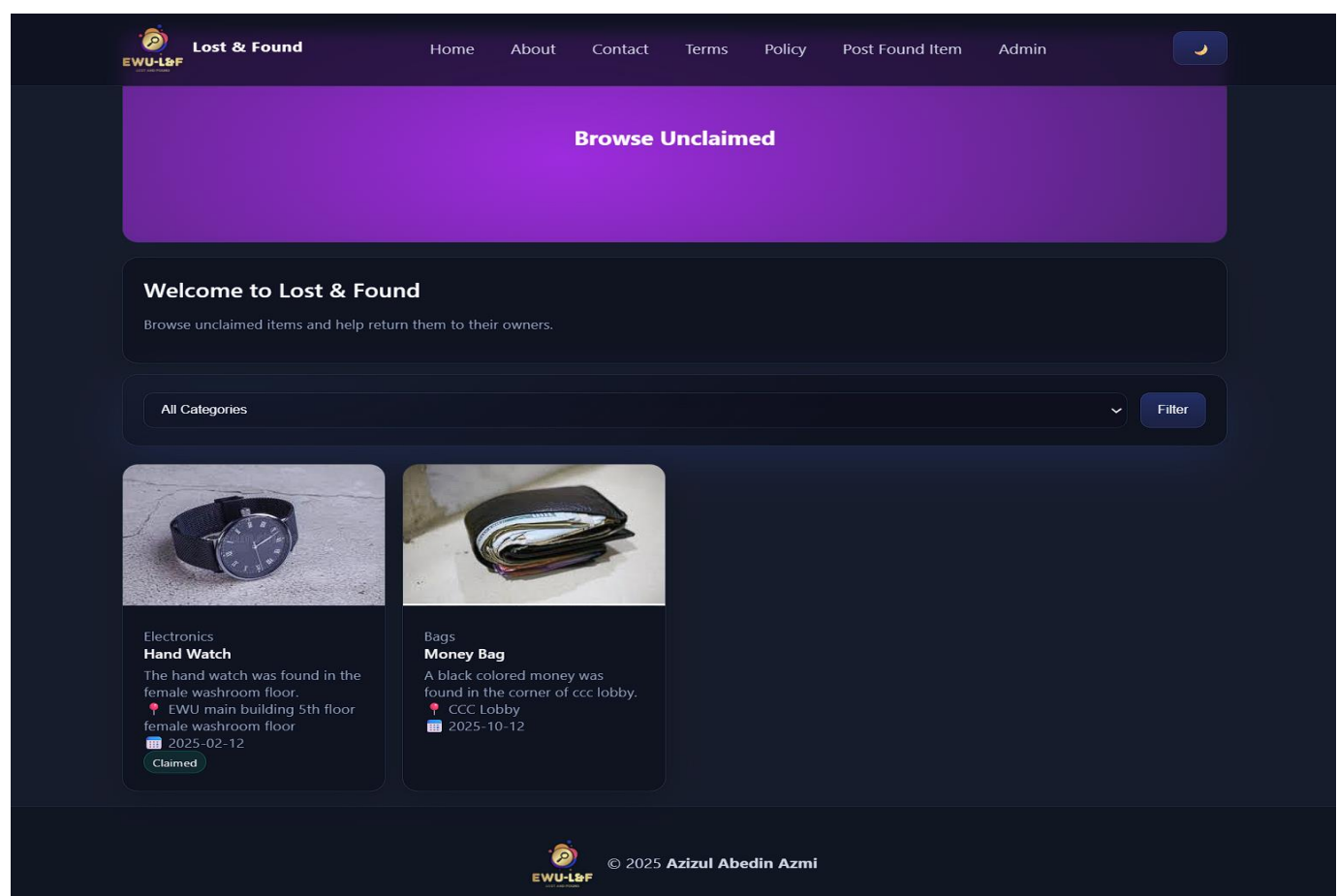
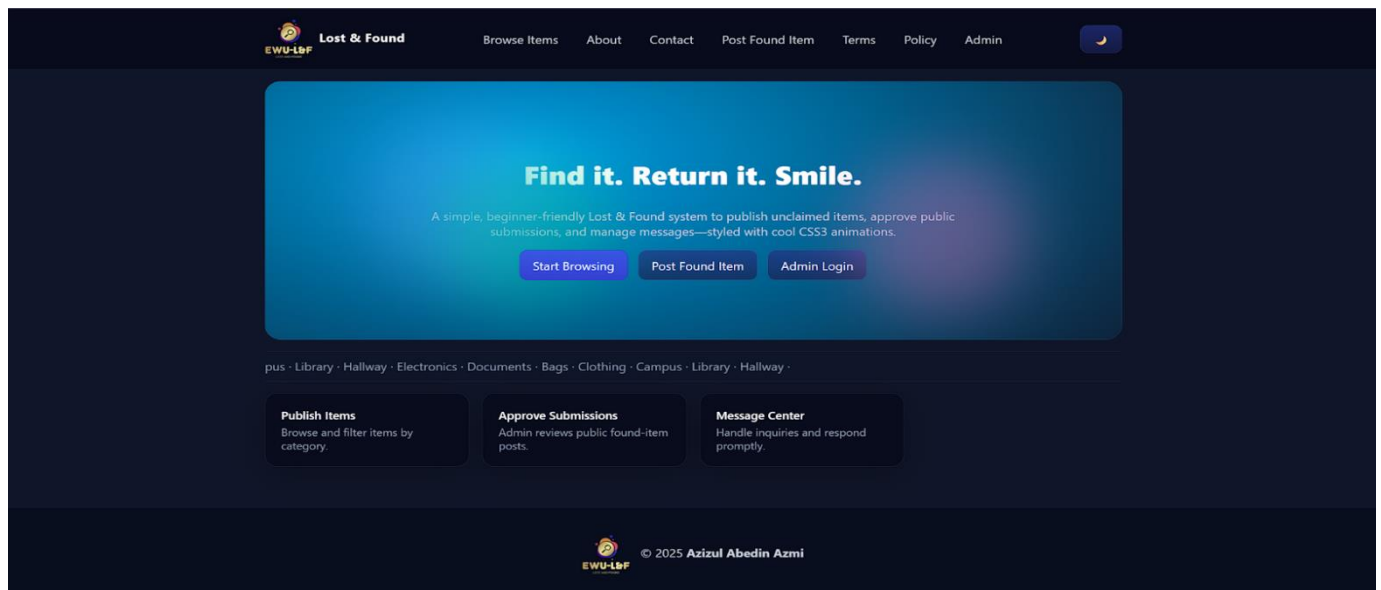
- **From catalog to actions:** A public visitor starts at [index.php](#), where they see the catalog and filters. From there, they navigate to [item.php](#) pages for details, and can choose to submit a claim, a found item, or a contact message via the appropriate forms.
- **Access to information pages:** At any time, the user can open About, Terms, or Privacy pages through the navigation menu. These pages explain how the system works, which builds confidence and clarity.

5.2.2 Admin Navigation


- **Login and dashboard:** Admins first access [admin/index.php](#) to log in. After successful authentication, they land on [dashboard.php](#), which gives an overview of system metrics.
- **Module-based navigation:** From the dashboard and sidebar, admins can jump to modules like Items, Claims, Categories, Users, Messages, Pages, Settings, and Account. This organized navigation supports efficient daily operations.

6. Functional User Interface Design – Public Side


6.1 Landing & Catalog Page (<public/index.php>)




6.2 Item Details & Claim Form (<public/item.php>)

**Lost & Found**

[Home](#) [Browse](#) [Contact](#)





Money Bag

Category: Bags
📍 CCC Lobby
📅 2025-10-12

A black colored money was found in the corner of ccc lobby.

Request to Claim

If this is yours, submit strong evidence (purchase receipt, unique identifiers, or photos). Admin will review and respond.

Your Name

Contact

Evidence / Proof

Describe evidence that proves ownership


Attach File (optional)

Choose File


No file chosen

Submit Claim Request

6.3 Submit Found Item Page (<public/submit.php>)

**Lost & Found**

[Home](#) [About](#) [Contact](#) [Post Found Item](#) [Admin](#)



Thank you! Your submission is pending review.

Post a Found Item

Title

Category

Date Found

e.g., Electronics

mm / dd / yyyy

Location Found

Description

Photo

Choose File

No file chosen

Your Name

Your Contact

Submit (Pending Approval)

6.4 Contact Page (<public/contact.php>)

The screenshot shows the 'Contact Us' page of the 'Lost & Found' system. The page has a dark blue header with the logo and navigation links: Home, About, Contact, Post Found Item, and Admin. A moon icon in the top right corner indicates a dark theme. The main content area is a light blue rounded rectangle containing a contact form. The form has four fields: 'Your Name' (filled with 'Fayroz Tasnim Rowza'), 'Email' (filled with 'fayroztasnim@gmail.com'), 'Subject' (filled with 'Contact with the admin for controversial issue'), and 'Message' (filled with 'Regarding a recent lost post, I want to tell that wasn't a true lost post.'). A 'Send' button is at the bottom of the form.

Lost & Found Home About Contact Post Found Item Admin

Contact Us

Your Name: Fayroz Tasnim Rowza Email: fayroztasnim@gmail.com

Subject: Contact with the admin for controversial issue

Message: Regarding a recent lost post, I want to tell that wasn't a true lost post.

Send

6.5 Static Information Page (About / Terms / Privacy)

The screenshot shows the 'About Us' page of the 'Lost & Found' system. The header is identical to the contact page, but the navigation links are Home, About, Contact, Post Found Item, and Admin. The main content area is a light blue rounded rectangle with the title 'About Us' and the text 'We simplify lost and found management and connect people with their belongings.'

Lost & Found Home About Contact Post Found Item Admin

About Us

We simplify lost and found management and connect people with their belongings.

The screenshot shows the 'Terms & Conditions' page of the 'Lost & Found' system. The header is identical to the contact page, but the navigation links are Home, About, Contact, Terms, Policy, Post Found Item, and Admin. The main content area is a light blue rounded rectangle with the title 'Terms & Conditions' and the text 'By using this system, you agree to submit accurate information and understand that administrators may approve or remove content at their discretion.'

Lost & Found Home About Contact Terms Policy Post Found Item Admin

Terms & Conditions

By using this system, you agree to submit accurate information and understand that administrators may approve or remove content at their discretion.

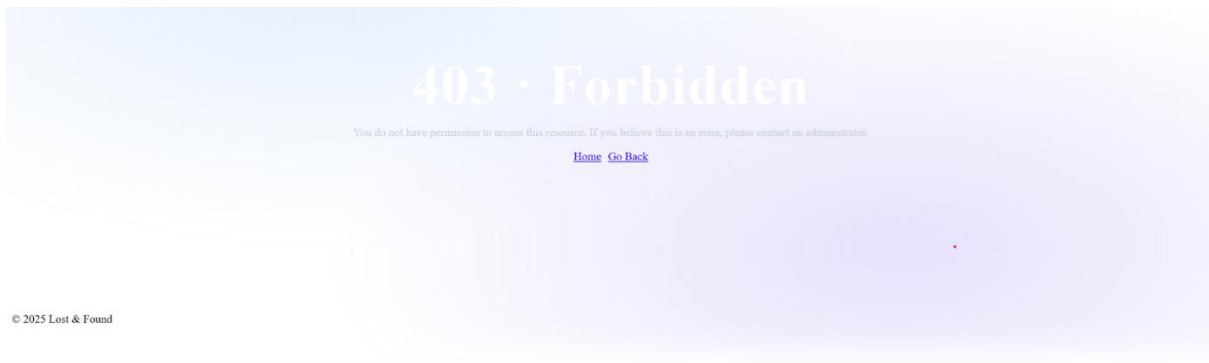
The screenshot shows the 'Privacy Policy' page of the 'Lost & Found' system. The header is identical to the contact page, but the navigation links are Home, About, Contact, Terms, Policy, Post Found Item, and Admin. The main content area is a light blue rounded rectangle with the title 'Privacy Policy' and the text 'We store minimal data to operate the system and do not share it with third parties. You may request deletion of your messages.'

Lost & Found Home About Contact Terms Policy Post Found Item Admin

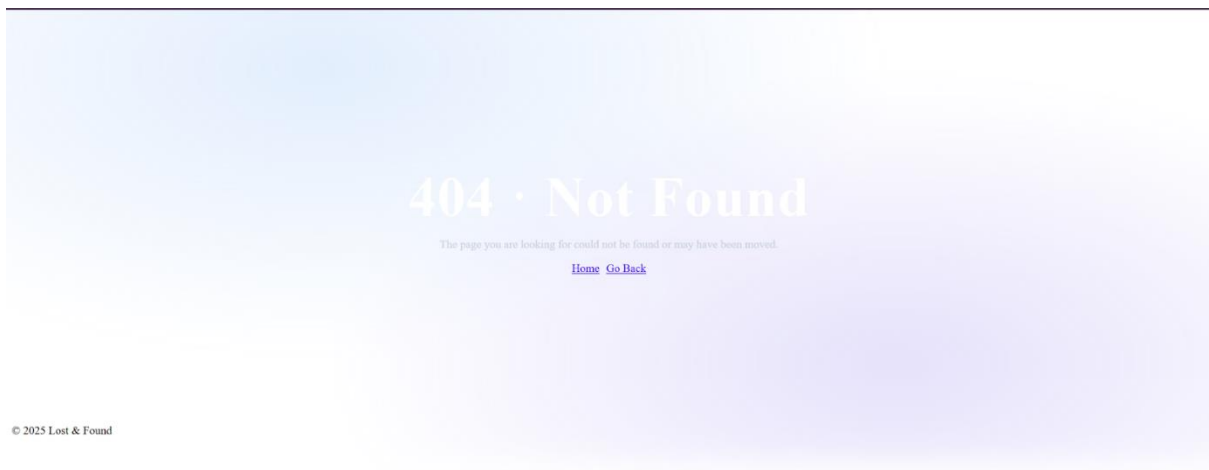
Privacy Policy

We store minimal data to operate the system and do not share it with third parties. You may request deletion of your messages.

6.6 Error Page – Forbidden ([403.php](#))



6.7 Error Page – Not Found ([404.php](#))

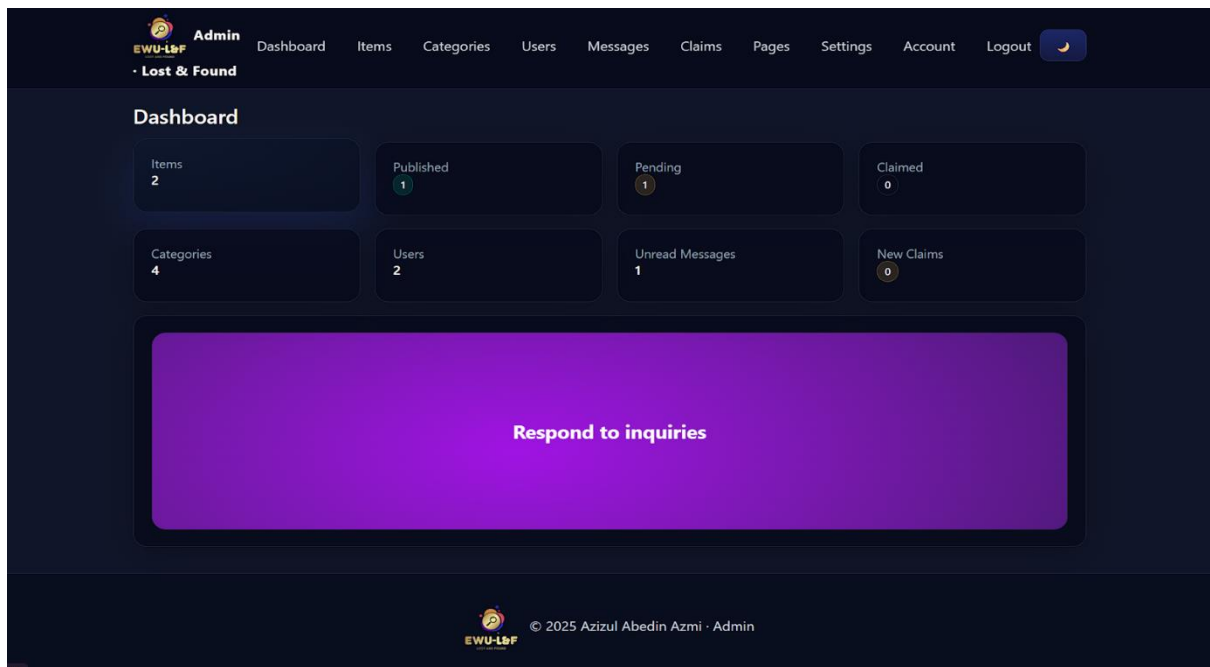


7. Functional User Interface Design – Admin Side

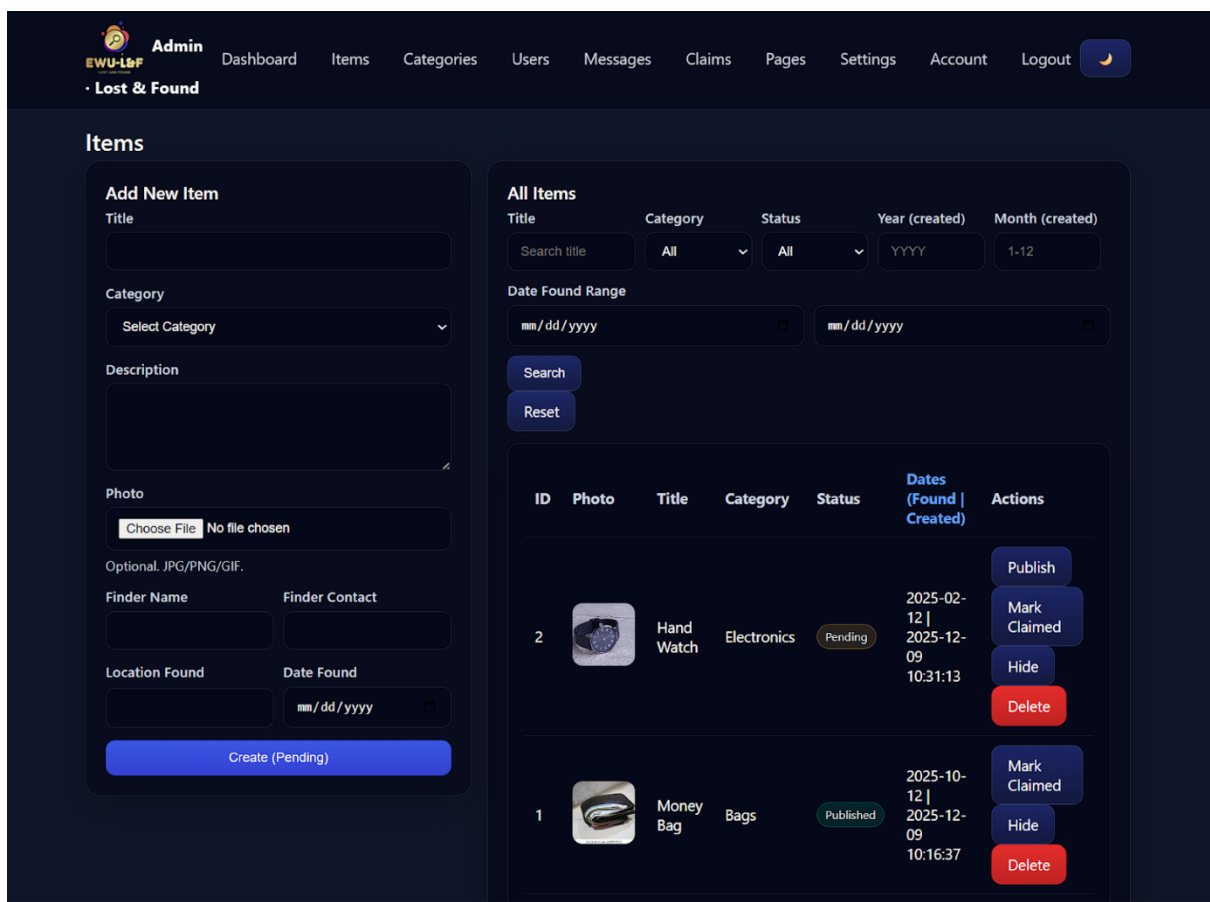
7.1 Admin Login Page ([admin/index.php](#))

The screenshot shows the Admin Login page with a dark blue background. At the top, there is a green rounded rectangle with the text "Admin Portal" in white. Below this, there is a dark blue rounded rectangle containing the "Admin Login" form. The form has two input fields: "Username" and "Password". The "Username" field has a red asterisk next to it, indicating it is required. Below the "Password" field, there is a blue "Login" button. At the bottom of the form, there is a tip: "Tip: Use a hashed password in the database (generated via PHP password_hash()). You can temporarily use admin/admin123 for legacy plaintext." At the bottom of the page, there is a logo for "EWU-L&F" and the text "© 2025 Azizul Abedin Azmi - Admin".

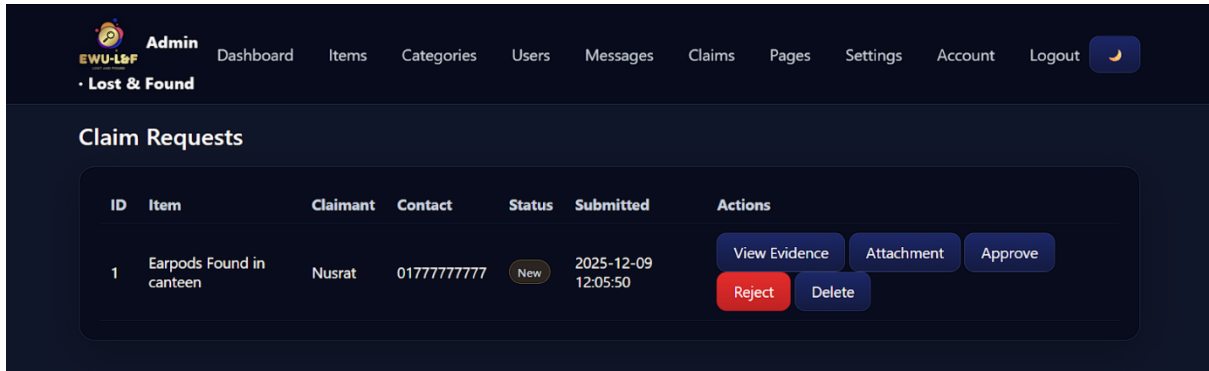
7.2 Admin Dashboard Overview (<admin/dashboard.php>)



7.3 Items Management Page (<admin/items.php>)



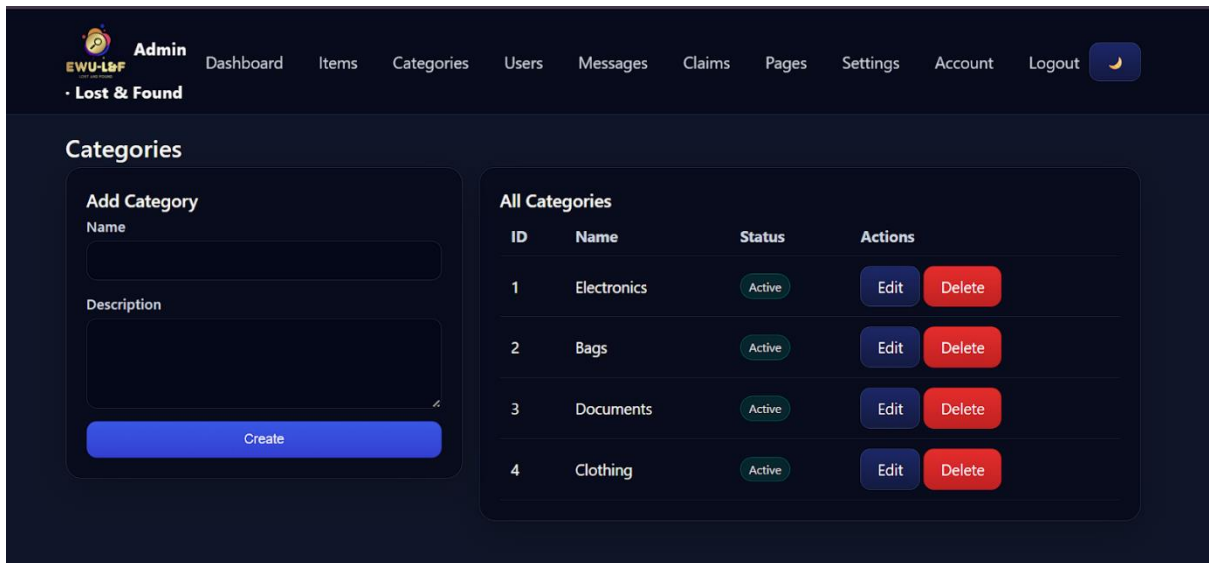
7.4 Claims Management Page ([admin/claims.php](#))



The screenshot shows the 'Claims Management Page' in the admin dashboard. The top navigation bar includes 'Admin', 'Dashboard', 'Items', 'Categories', 'Users', 'Messages', 'Claims', 'Pages', 'Settings', 'Account', 'Logout', and a moon icon. The 'Claims' menu item is highlighted. Below the navigation bar, the page title is 'Claim Requests'. A table displays a single claim request with the following data:

ID	Item	Claimant	Contact	Status	Submitted	Actions
1	Earpods Found in canteen	Nusrat	01777777777	New	2025-12-09 12:05:50	View Evidence Attachment Approve Reject Delete

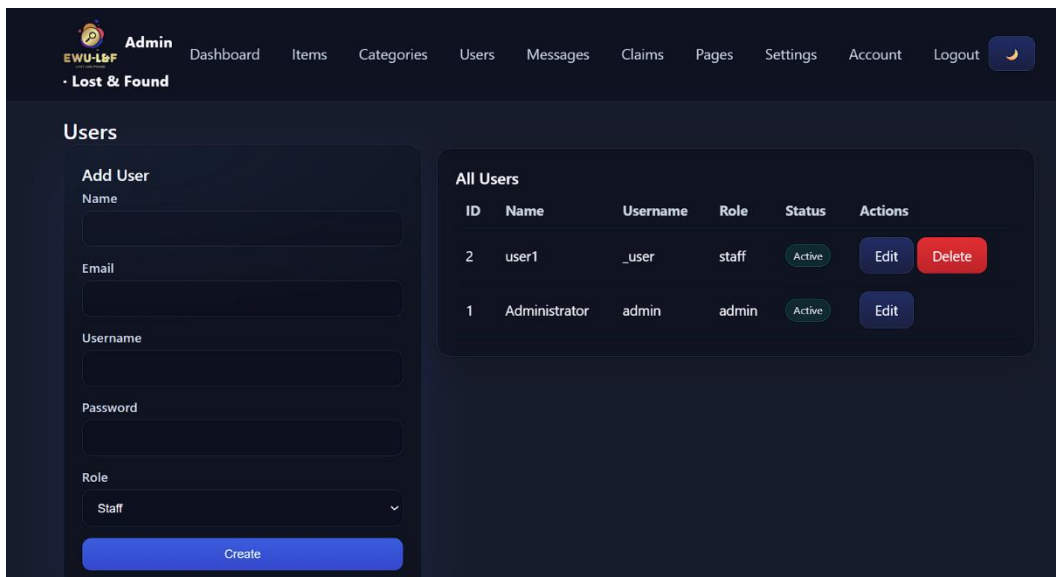
7.5 Categories Management Page ([admin/categories.php](#))



The screenshot shows the 'Categories Management Page' in the admin dashboard. The top navigation bar is identical to the previous page. The 'Categories' menu item is highlighted. Below the navigation bar, the page title is 'Categories'. On the left, there is a form to 'Add Category' with fields for 'Name' and 'Description', and a 'Create' button. On the right, a table displays all categories:

ID	Name	Status	Actions
1	Electronics	Active	Edit Delete
2	Bags	Active	Edit Delete
3	Documents	Active	Edit Delete
4	Clothing	Active	Edit Delete

7.6 Users Management Page ([admin/users.php](#))



The screenshot shows the 'Users Management Page' in the admin dashboard. The top navigation bar is identical to the previous pages. The 'Users' menu item is highlighted. Below the navigation bar, the page title is 'Users'. On the left, there is a form to 'Add User' with fields for 'Name', 'Email', 'Username', 'Password', and a 'Role' dropdown menu (currently set to 'Staff'), and a 'Create' button. On the right, a table displays all users:

ID	Name	Username	Role	Status	Actions
2	user1	_user	staff	Active	Edit Delete
1	Administrator	admin	admin	Active	Edit

7.7 Messages Management Page (<admin/messages.php>)

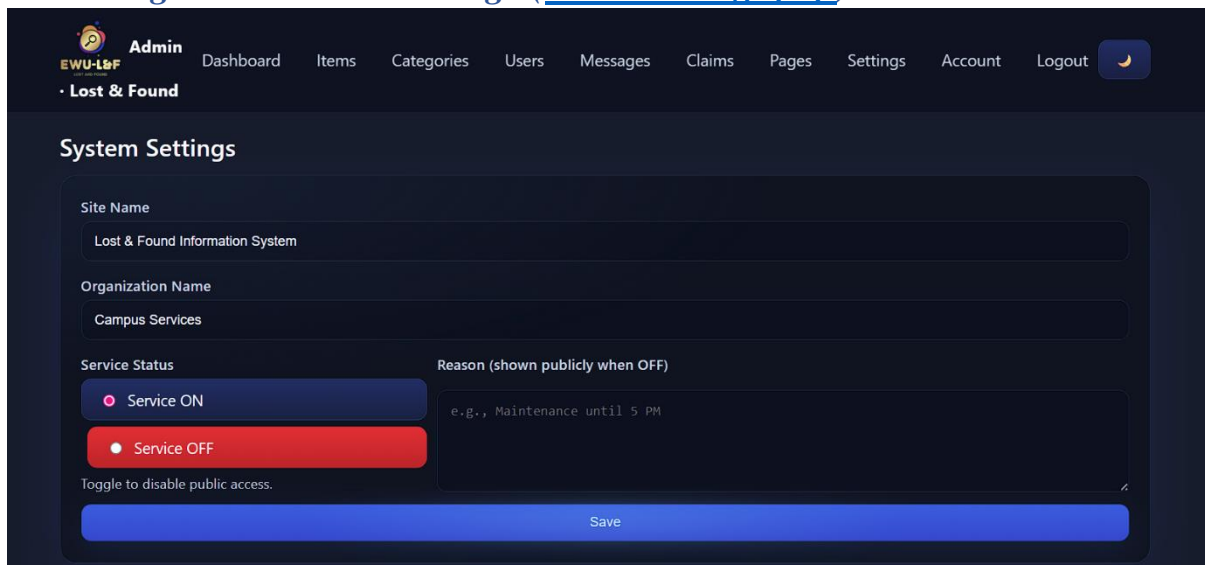
The screenshot shows the 'Messages / Inquiries' section of the Admin interface. The top navigation bar includes 'Admin', 'Dashboard', 'Items', 'Categories', 'Users', 'Messages', 'Claims', 'Pages', 'Settings', 'Account', 'Logout', and a dark mode toggle. Below the navigation bar, the page title is 'Messages / Inquiries'. The main content area displays a table of messages with columns: ID, From, Email, Subject, Status, Received, and Actions.

ID	From	Email	Subject	Status	Received	Actions
2	Fayroz Tasnim Rowza	fayroztasnim@gmail.com	Contact with the admin for controversial issue	New	2025-12-09 10:34:25	Mark Read Delete
1	Fayroz Tasnim Rowza	fayroztasnim@gmail.com	checking	Read	2025-12-04 03:31:12	Mark Unread Delete

7.8 Pages Management Page (<admin/pages.php>)

The screenshot shows the 'Page Management' section of the Admin interface. The top navigation bar is identical to the previous page. Below the navigation bar, the page title is 'Page Management'. On the left, there is a sidebar with buttons for 'Home', 'About', 'Contact', 'Terms', and 'Privacy'. The main content area is titled 'Edit: HOME' and contains a form for editing the home page. The form has a 'Title' field with the value 'Welcome to Lost & Found' and a 'Content (HTML allowed)' field with the value '<p>Browse unclaimed items and help return them to their owners.</p>'. A 'Save' button is at the bottom of the form.

7.9 Settings & Maintenance Page (<admin/settings.php>)



The screenshot shows the 'System Settings' page in the Admin panel. The top navigation bar includes links for Dashboard, Items, Categories, Users, Messages, Claims, Pages, Settings, Account, and Logout. The 'Settings' link is active. Below the navigation bar, the 'System Settings' section contains several form fields: 'Site Name' (Lost & Found Information System), 'Organization Name' (Campus Services), 'Service Status' (Service ON selected), and 'Reason (shown publicly when OFF)' (e.g., Maintenance until 5 PM). A 'Save' button is at the bottom.

8. Instructions for Users

8.1 Instructions for Public Users

- **Browse the catalog:** Open the website and start at the main landing page. Scroll through the list of items or use the category filter to narrow down the results.
- **View item details and claim:** Click on an item card to open its detail page. If you think the item belongs to you and it is still claimable, fill in the claim form with your name, contact, and evidence, then submit.
- **Submit a found item:** If you have found an item, go to the “Submit Item” page and fill in all details including location, date, and a description. Optionally upload a photo, then submit the form so admins can review and publish it.
- **Contact the admin:** If you have general questions or need help, visit the contact page and send a message with your name, email, subject, and details. You will see a confirmation message when it is sent.
- **Read the policies:** Check the About, Terms, and Privacy pages to understand how the system works, what rules apply, and how your information is handled.

8.2 Instructions for Admin Users

- **Log in to the admin panel:** Visit the admin URL and enter your username and password. If credentials are correct, you will be redirected to the dashboard. Here a temporary username and password has been used which are respectively admin, admin123.
- **Review items and claims:** Use the Items and Claims sections to see new submissions and requests. Approve items to make them public, and process claims by approving, rejecting, or marking them as fulfilled.
- **Manage system data:** Use Categories, Users, Messages, and Pages modules to keep the system organized and content up to date. Update settings to change the site name or enable maintenance mode when necessary.

9. Security, Error Handling and Maintenance

9.1 Security Practices

1. Admin passwords stored with [password_hash\(\)](#) and checked with [password_verify\(\)](#).
2. Admin routes protected using session checks in [auth.php](#).

3. Prepared statements in [db.php](#) avoid SQL injection.
4. Basic server-side validation for forms (required fields, basic formats).
5. File uploads stored under [uploads/](#) and checked for allowed types and size (recommended).

9.2 Error Handling

1. Custom 403 and 404 pages, rendered by helper functions, improve user experience.
2. Common error responses and redirects are handled in [helpers.php](#).
3. User-friendly status messages are shown after important actions.

9.3 Maintenance Mode

1. Controlled through [settings](#) table and [admin/settings.php](#).
2. When service is “OFF”, public pages show a maintenance message instead of normal content.
3. Useful for deployments, updates, or database maintenance.

10. Limitations and Future Enhancements

10.1 Limitations

1. Only web version; no native Android/iOS app.
2. Public users do not have full personal accounts; claims are stored with name and contact only.
3. No automatic AI-based matching algorithm between lost and found items; admin must review manually.
4. Basic security is implemented, but no 2-factor authentication or full audit logs.
5. No built-in analytics or reporting dashboards.

10.2 Future Enhancements

1. Add email or SMS notifications for new items and claim status changes.
2. Build a basic registration system for public users to track their own claims.
3. Implement advanced filters and smart suggestions between similar items.
4. Improve file upload security and move uploads outside the web root in production.
5. Add reporting features (export to CSV, PDF) for admin.
6. Develop mobile apps using the same API.

11. Conclusion

The EWU Lost & Found Management System is a complete, practical solution for managing lost and found items at a campus like EWU or can be used for small organizations. It combines a modern responsive interface with a clear item catalog, claim process, and admin tools. The system centralizes information, reduces confusion, and helps return items to their owners in an organized way.

The project uses common and simple web technologies (PHP, MySQL, HTML, CSS, JS), making it easy to understand, maintain, and extend. With the planned improvements, EWU-L&F can grow into a more powerful platform, but even in its current form, it already offers a strong foundation for real-world use.