SwiftFound SwiftFound Lost and Found System

4B

Ibrahim Ulwan bin Zainudin (team leader), 2024413296
Muhammad Faiz bin Anwar, 2024680314
Muhammad Izzat bin Hashim, 2024438392
Afrizal Haffiz bin Kamarul Arifin, 2024220448

SwiftFound Lost and Found System

User Manual

Version 1.0

6/7/2026

Table of Contents

List of Figures

List of Tables

1. Introduction

This User Manual (UM) provides the information necessary for campus residents and admins to effectively use the SwiftFound Platform (SF), operating under identification number v1.0.0 (Release Build 2026-06). The purpose of this document is to outline the operational workflows of the platform, which was developed to provide a localized, real-time ecosystem for campus residents to report lost items and file claims, while equipping admins with management capabilities like handling user reports, dismissing cases, and restricting accounts. The scope of activities that resulted in its development includes building a relational database for user reputation tracking, refining data retrieval modules, and implementing secure role-based access. This manual is intended for campus residents managing active item claims and authorized admins utilizing dashboard management features. It is expected to evolve asynchronously alongside future system optimizations. Security and privacy considerations are strictly enforced across the application; campus resident sessions are protected via secure token configurations, data integrity is maintained through server-side validation loops, and admin interface access is completely gated and protected exclusively through a single secure authorization code instead of individual account credentials.

1.1 Overview

The SwiftFound Platform (SF) is a web-based community tool designed to help campus residents report lost belongings and coordinate item returns through a straightforward, real-time matching system. The platform functions within a shared campus environment, relying on an internal user reputation system that scores helpful community interactions and penalizes fraudulent claims to ensure trust.

The following list outlines the core design elements and operations of the system:

- **Key Features and Major Functions:**
    - **Item Posting and Claiming:** Allows campus residents to post descriptions and photos of found or lost items and submit claim requests to recover their belongings.
    - **Administrative Oversight:** Provides a dedicated panel where an admin can review active community reports, dismiss invalid cases, and restrict users who misuse the platform.
    - **Reputation System:** Tracks and displays reliability scores for campus residents to promote honest interactions during item handovers.
- **System Architecture:** The platform uses a modern, web-based client-server architecture. All application data is processed on a centralized backend server and sent instantly to the client's device, ensuring that item updates and status changes happen in real time.
- **User Access Mode:** Users interact with the platform entirely through a clean, responsive graphical user interface (GUI). It is accessible directly through any standard desktop or mobile web browser, eliminating the need to install external applications or plugins.
- **System Environment and Special Conditions:** *
- **Resident Access:** Campus residents log into the system using individual personalized accounts to track their unique items, claims, and reputation points.

- **Admin Access:** The administrative management panel operates under a special condition where it requires no account creation or traditional registration; access is instead gated entirely behind a single, secure master authorization code.

2. Getting Started

Access to the system depends on your target role. For a campus resident, the process begins on the main landing page, where you click either the Login or Register buttons located in the top-right corner of the navigation header.

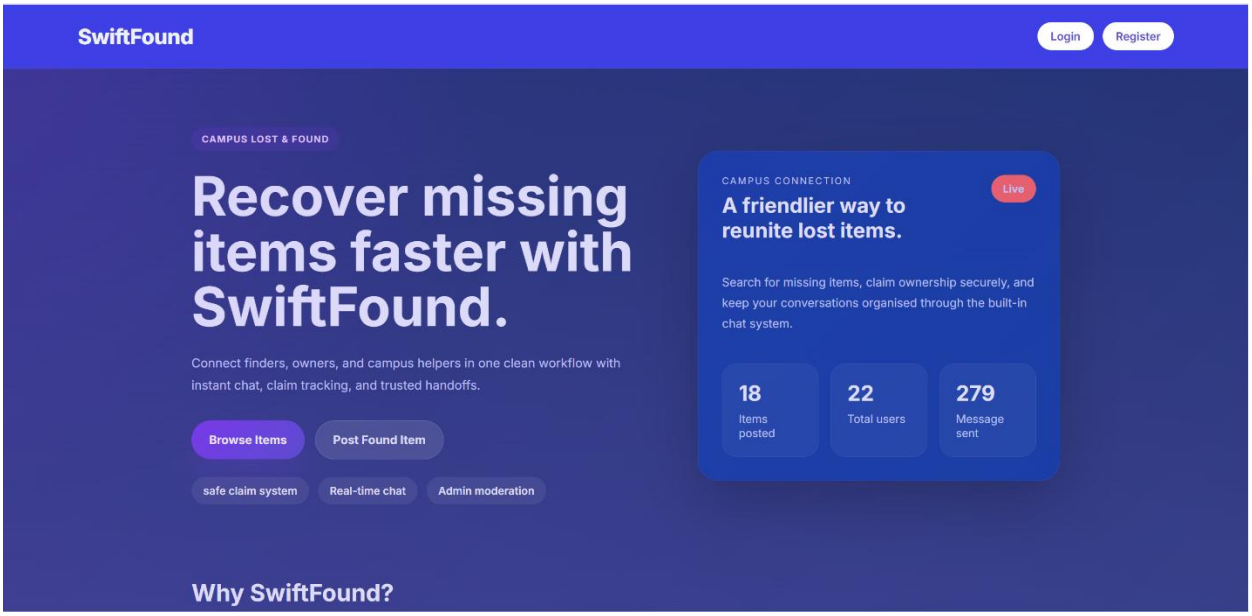

Figure 1: landing page

If you do not have an account yet, clicking Register brings up the account creation window. Enter an available username, set a password, verify it in the confirmation field, and click Sign Up to create your resident profile.

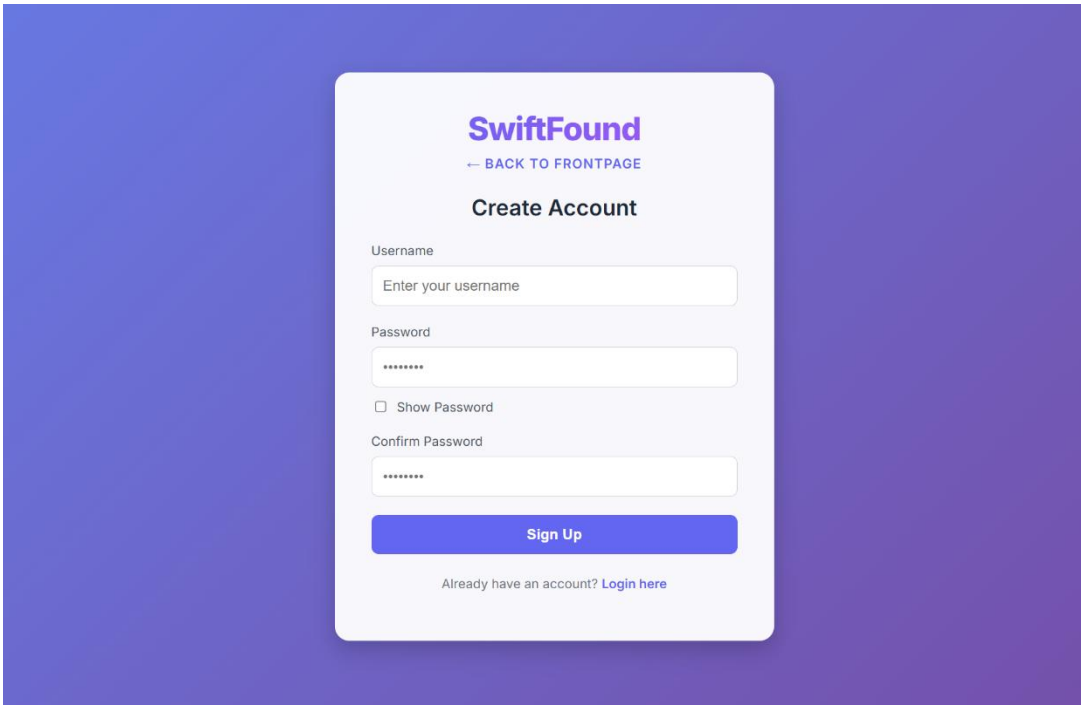

Figure 2: Register account page

If you already have an account, clicking Login from the landing page opens the standard credentials form. Simply fill in your username and password, then click the Log In button to initialize your session.

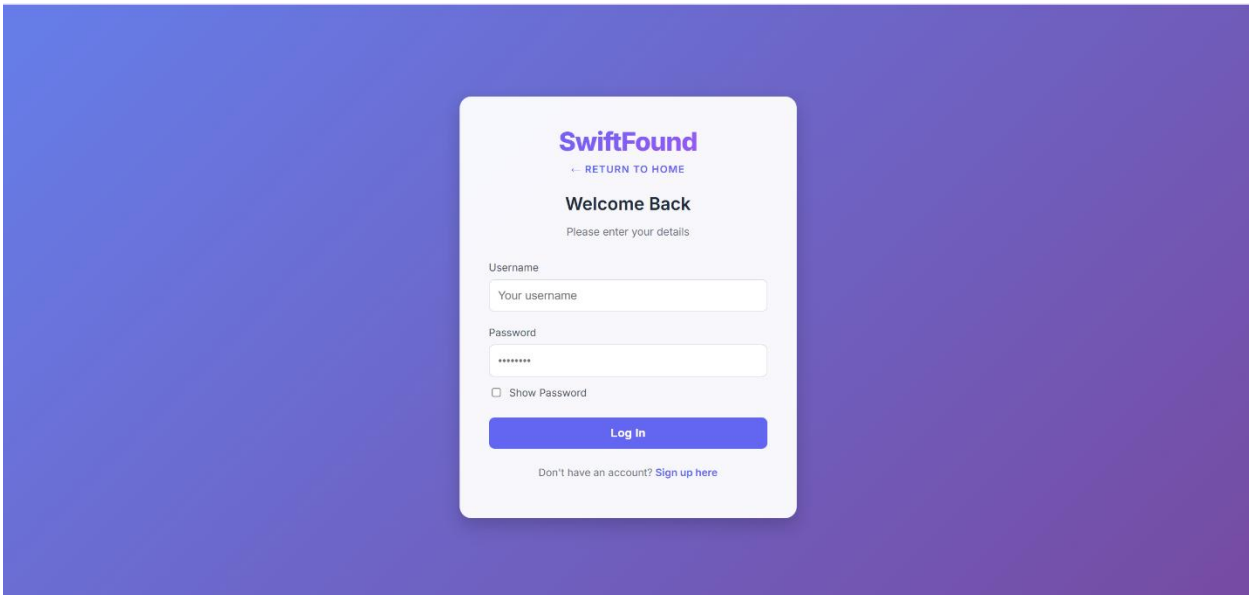

Figure 3: Login page

Upon a successful login, campus residents are directed straight to the Member Dashboard. This home screen serves as your central command area, displaying a summary layout of your activity metrics—including your total posted items, active claims, pending requests, and resolved cases. It also prominently features your live Reputation Progress Bar, showing your current status level and how many points you need to reach the next community tier. From the left-hand navigation menu or the centralized "Quick Actions" panel, you can jump straight to browsing items, posting a found object, checking ongoing claims, or launching the real-time chat interface. To end your session at any time, click the red Logout button at the bottom of the left sidebar.

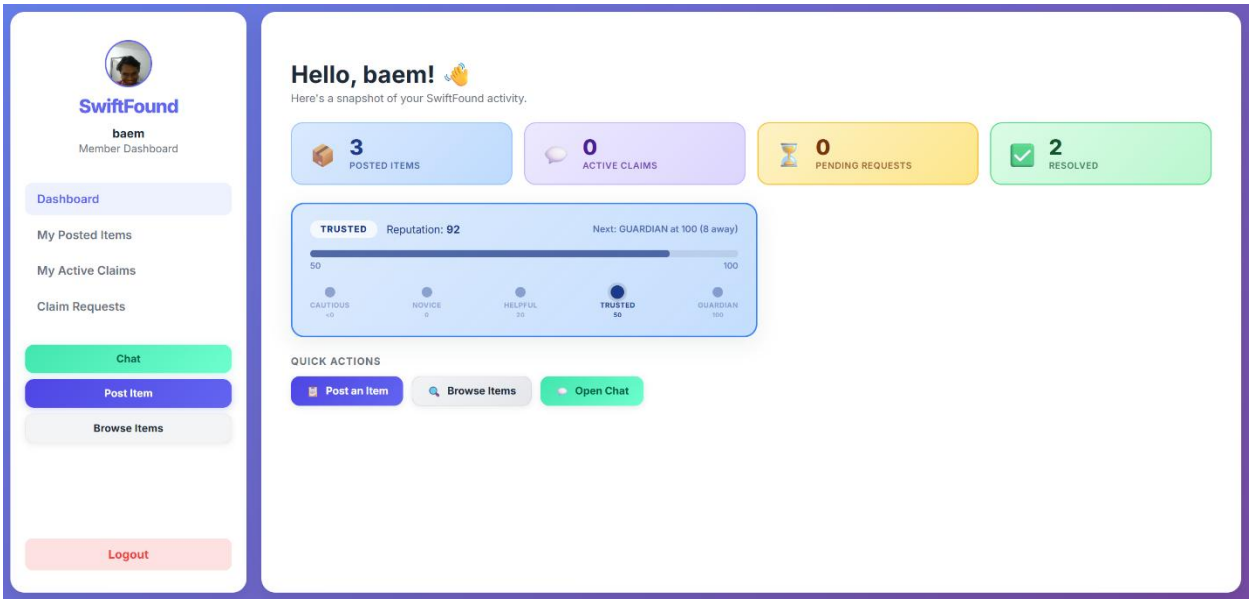

Figure 4: Home page

For an administrator, access bypasses the standard landing page buttons entirely. To initiate an admin session, you must manually type the system's hidden administrative directory path into your browser's address bar. This loads the dedicated Admin Portal security gate. Because administrative features do not use individual user accounts, you simply enter the single master code into the Security Key field and click Authorize Access to open the management tools.

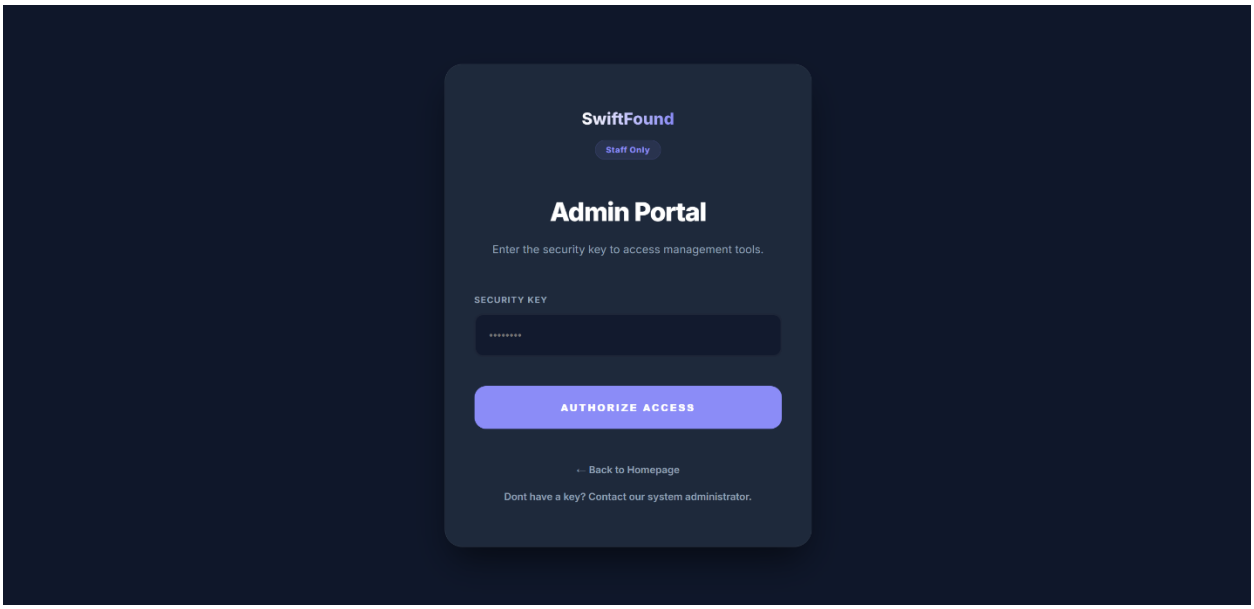

Figure 5: Admin access page

Authorizing access routes you immediately into the Platform Statistics Dashboard within the Admin Panel. This view gives you an instant, high-level operational summary of the system, detailing metrics like total registered users, active lost items, total ongoing claims, and pending moderation reports. Directly beneath the primary stat blocks, the Claim Status Breakdown row lets you observe how many interactions are currently pending, actively chatting, awaiting owner confirmation, or marked as resolved, rejected, or canceled. The left sidebar layout allows you to toggle easily between this live overview grid, the detailed user report management queue, and the full user account registry. To terminate access and secure the interface, click the Logout button positioned at the bottom left of the panel.

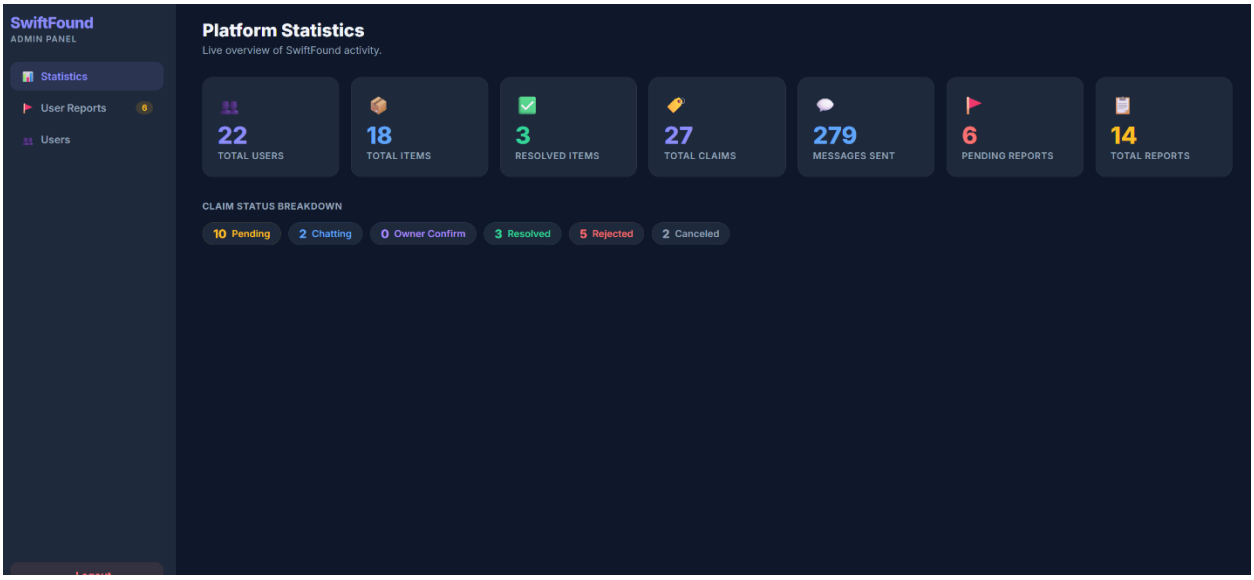

Figure 6: Admin page

2.1 Cautions & Warnings

To maintain platform integrity and ensure a safe environment for recovering lost property, both campus residents and administrators must adhere to the following operational guardrails, system prohibitions, and access penalties.

- **Fraudulent Claim Prohibitions:** Campus residents are strictly prohibited from submitting false ownership claims over items they did not lose. The system tracks all interactions, and attempting to claim an item using intentionally misleading descriptions or falsified proof will result in an immediate penalty to your community reliability score.
- **The Reputation Penalty System:** Your account status is directly tied to your behavior. Helpful handovers and verified, successful returns will advance your progress bar toward higher, premium community tiers. Conversely, picking up items without owner confirmation, generating spam posts, or receiving valid user reports will cause your score to drop into the negative **CAUTIOUS** tier.
- **Account Restriction Penalties:** If a campus resident repeatedly violates platform rules or behaves maliciously in the real-time chat interface, an administrator will flag the account. Once an account is updated to **RESTRICTED** status, a warning banner will permanently lock the profile layout, instantly revoking your permission to submit new claims or interact with active listings.
- **Unauthorized Administrative Access:** The administrative dashboard is strictly reserved for authorized platform overseers. Because access is gated entirely by a single, shared master security key instead of individual accounts, sharing this code with unauthorized personnel or attempting to brute-force the hidden admin portal URL is treated as a severe security violation and will result in disciplinary action under university IT and campus residency regulations.
- **Data Content & Image Permissions:** All images and descriptive text uploaded to the platform when logging a found item must respect student privacy and campus codes of

conduct. Users retain ownership of their posts, but by publishing an item to the public registry, they grant the platform temporary permission to display that content across the graphical user interface until the item is successfully marked as resolved or removed.

2.2 Set-up Considerations

Because SwiftFound is a fully web-based application, you do not need to install any software or perform a complex equipment setup. It runs entirely through your web browser over a standard internet connection.

Network & Software Requirements

- **Internet Connection:** Your device must be connected to an active network (such as campus Wi-Fi or mobile data) so that item posts, claim statuses, and chat messages update instantly in real time.
- **Web Browser:** The platform works on any modern web browser (like Chrome, Safari, Edge, or Firefox). There are zero external downloads or plug-ins required.

Hardware Inputs & Outputs

- **Input:** You only need a standard touchscreen or keyboard/mouse to type descriptions and enter login credentials or the admin security key. A device camera is helpful for campus residents to take and upload photos of found items.
- **Output:** Any standard mobile phone, tablet, or laptop screen will cleanly display the platform's graphical dashboard and interactive elements.

2.3 User Access Considerations

The system strictly divides its access controls into two distinct user tiers to separate general public actions from administrative management utilities. Each group operates on a completely different accessibility model with specific permissions and system restrictions.

- **Campus Residents**
    - **Access Model:** This group accesses the platform via personalized accounts created through the standard username and password registration page.
    - **Permissions:** Campus residents are authorized to manage their personal profiles, view active lost-and-found listings, post found items, submit claim requests for missing property, and use the real-time chat feature to coordinate returns.
    - **Restrictions:** Residents are strictly locked out of all administrative panels and cannot view global system statistics. Furthermore, if a resident's account is flagged for violating community guidelines, their account status changes to **RESTRICTED**. This places a permanent restriction on their accessibility, disabling their ability to submit claims or interact with active listings entirely.
- **Administrators (Admins)**
    - **Access Model:** This tier does not use individual accounts or traditional user registration. Accessibility is granted under the special condition that the operator

knows the hidden administrative URL path and enters a single, shared master security key at the portal gate.

- **Permissions:** Admins have full access to the Platform Statistics Dashboard, the global user registry, and the report moderation queues. They have the authority to review flagged interactions, permanently dismiss community reports, and toggle the restriction flags on malicious user accounts.
- **Restrictions:** Admins operate strictly on an ecosystem management level and do not participate in the core item-claiming or posting workflows meant for residents. Because access relies on a single master code rather than unique user profiles, session tracking is uniform across the administrative tier.

2.4 Accessing the System

Registering an Account (Campus Residents)

1. Open your web browser and navigate to the SwiftFound main landing page.
2. Click the **Register** button located in the top-right corner of the navigation header.
3. In the **Create Account** window, type a unique username into the *Username* field.
4. Enter a secure password into the *Password* field and re-type it exactly into the *Confirm Password* field.
5. Click the blue **Sign Up** button to finalize your profile and initialize your database record.

Logging On to the System (Campus Residents)

1. From the main landing page, click the **Login** button in the top-right corner of the header.
2. Input your registered username and password into the respective form fields.
3. Click the blue **Log In** button to authorize your session and open your personal Member Dashboard.

Logging On to the System (Administrators)

1. Manually type the hidden administrative directory path into your browser's address bar to bypass the public landing page.
2. Once the dark-themed **Admin Portal** security gate loads, locate the *Security Key* field.
3. Type the shared master operational code into the input field.
4. Click the **Authorize Access** button to initialize the Platform Statistics Panel.

Changing or Resetting a Password

- **No Self-Service Recovery:** Because this version of the application does not feature automated email recovery links or self-service password reset forms, users cannot change their passwords directly through the interface.
- **Support Action:** If you forget your account password or require a credential update, you must contact the platform administrator directly. The administrator can manually update your password record on the backend server to restore access to your account dashboard.

2.5 System Organization & Navigation

Member Dashboard (Campus Residents)

The resident environment is built around a unified sidebar menu that remains anchored to the left side of the screen, allowing campus residents to jump between their active items and messaging channels with a single click.

Dashboard Home Screen

- **Navigation Path:** Click **Dashboard** at the top of the left navigation menu.
- **Purpose:** Serves as the resident's default landing hub. It displays a real-time summary grid of items posted, active claims, pending requests, and resolved cases, alongside the live visual community reputation meter.

My Posted Items

- **Navigation Path:** Click **My Posted Items** in the left navigation menu.
- **Purpose:** Opens a dedicated repository displaying all found property listings logged by the resident, allowing them to track who has requested to claim those items.

My Active Claims

- **Navigation Path:** Click **My Active Claims** in the left navigation menu.
- **Purpose:** Stores and monitors the real-time fulfilment status of claim requests the resident has submitted for their own lost property.

Claim Requests

- **Navigation Path:** Click **Claim Requests** in the left navigation menu.
- **Purpose:** Houses the incoming claim applications submitted by other campus residents who are attempting to verify ownership of items you have posted.

Communication and Actions

- **Navigation Path (Chat):** Click the green **Chat** button in the left sidebar menu, or click the **Open Chat** action button inside a specific item card.
- **Navigation Path (Post Item):** Click the blue **Post Item** button in the left sidebar menu.
- **Purpose:** The green chat button opens the primary inbox module where real-time text threads are hosted to coordinate physical item handovers. The blue post button initializes the data input form to register a newly discovered item onto the campus network.

Admin Panel (Administrators)

Authorized operators who enter the master security key are greeted by a specialized, dark-themed management interface. Its navigation is isolated strictly to backend maintenance functions.

Platform Statistics

- **Navigation Path:** Click **Statistics** at the top of the Admin Panel left sidebar.
- **Purpose:** Displays the master system metrics dashboard, showcasing total user registration counts, global item inventories, outstanding claims, and live status breakdowns (such as pending, chatting, or resolved cases).

User Reports Queue

- **Navigation Path:** Click **User Reports** in the Admin Panel left sidebar (displays a numeric indicator highlighting active cases).
- **Purpose:** Routes the admin directly to the moderation queue where flagged community disputes, user complaints, and malicious claim attempts are reviewed. From here, admins can permanently dismiss reports or apply account restrictions.

Full User Registry

- **Navigation Path:** Click **Users** in the Admin Panel left sidebar.
- **Purpose:** Opens the global user management data grid, giving the administrator the ability to look up individual resident records, audit community reputation scores, and manually toggle account states between active and restricted.

2.6 Exiting the System

To secure your personal data, prevent unauthorized access to active sessions, and close down management tools properly, follow these exit procedures:

- **Exiting as a Campus Resident**
    1. Locate the left-hand navigation sidebar from any screen within your dashboard.
    2. Scroll to the bottom of the menu and locate the red **Logout** button.
    3. Click the **Logout** button to instantly destroy your local browser session token and wipe active memory parameters.
    4. The application will automatically redirect you back to the public front landing page, confirming that your profile credentials are no longer active on the device.
- **Exiting as an Administrator**
    1. Locate the dark-themed navigation sidebar on the left side of the screen.
    2. Look at the very bottom corner of the panel for the **Logout** button.
    3. Click the **Logout** button to lock the administrative environment.
    4. Clicking this button immediately revokes the master security key authorization cache for that browser session and redirects the screen away from the backend tools.

Security Reminder: Because the Admin Panel relies on a shared master security code rather than individual accounts, always click the explicit **Logout** button before walking away from a device instead of just closing the browser tab. This ensures the management session is fully deactivated and cannot be accessed through your browser's back-history.

3. Using the System

The following sub-sections provide detailed, step-by-step instructions on how to use the various functions or features of the SwiftFound lost and found system.

3.1 Post Found Item Feature

This function allows campus residents who have discovered lost property on campus to log the item details into the system database, making it visible to the community for recovery.

3.1.1 User Input

To register a newly discovered item, the campus resident navigates to the **Post Item** form and inputs the following mandatory parameters:

1. **Title:** A short name of what was found (e.g., "Watch", "car keys").
2. **Category:** A drop-down selection to classify the object (e.g., Electronics, Jewelry, Other).
3. **Description:** Specific visual text details explaining the item's state.
4. **Location:** The specific campus zone where the object was found (e.g., "volleyball court", "susur gajah").
5. **Upload Image:** A system file picker attachment where the user uploads a clear photo captured via their device camera.
6. **Secret Question:** A verification question crafted by the finder (e.g., "What color was it?") used to filter out dishonest claim attempts before a conversation starts.

Clicking the **Submit Form** button packages these values into a multipart payload and transmits them to the server layer.

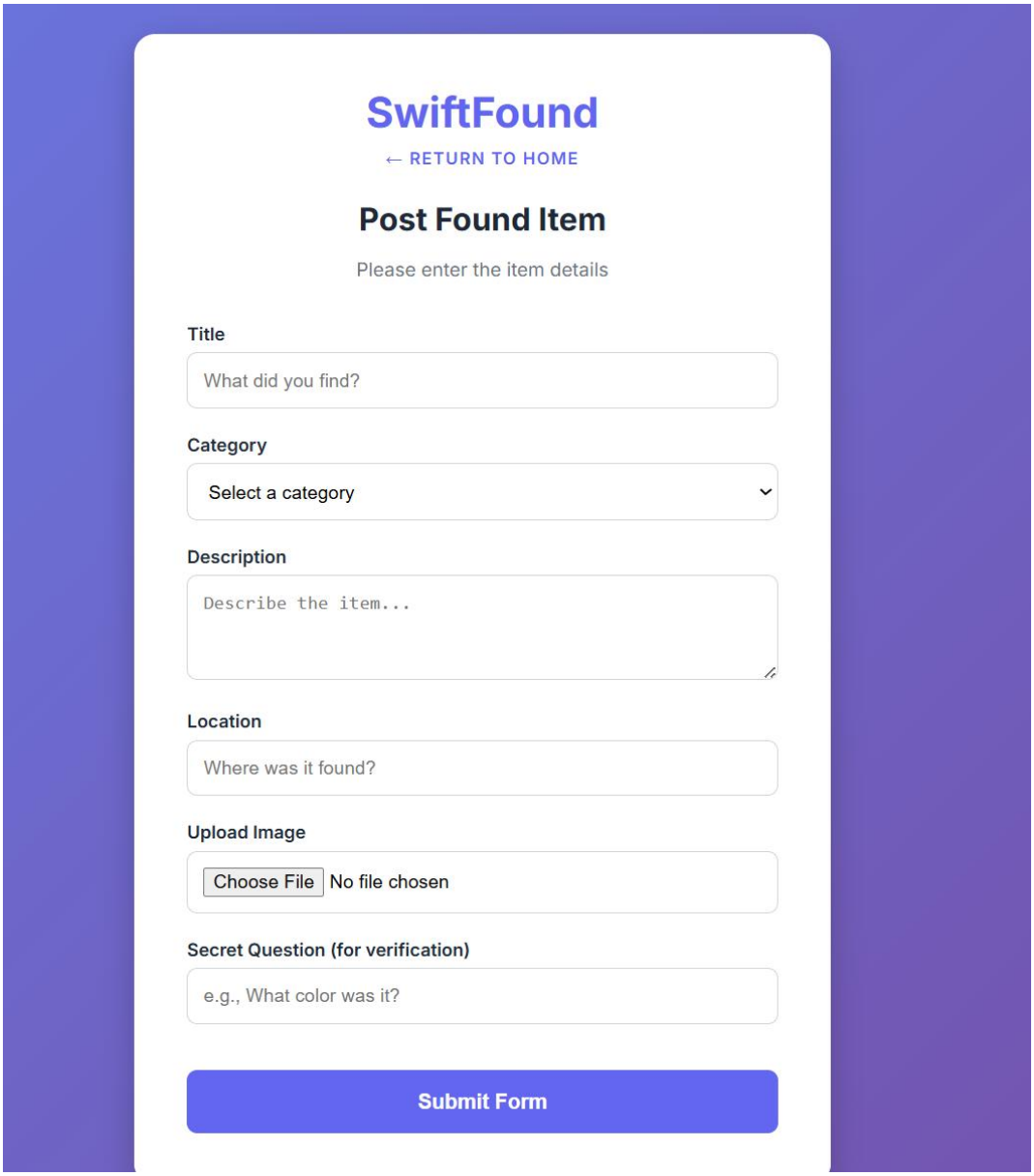

Figure 7: Post item form

3.2 Browse Item

This core module handles the discovery of lost property, utilizing client-side script filtration running over a comprehensive structural server data fetch.

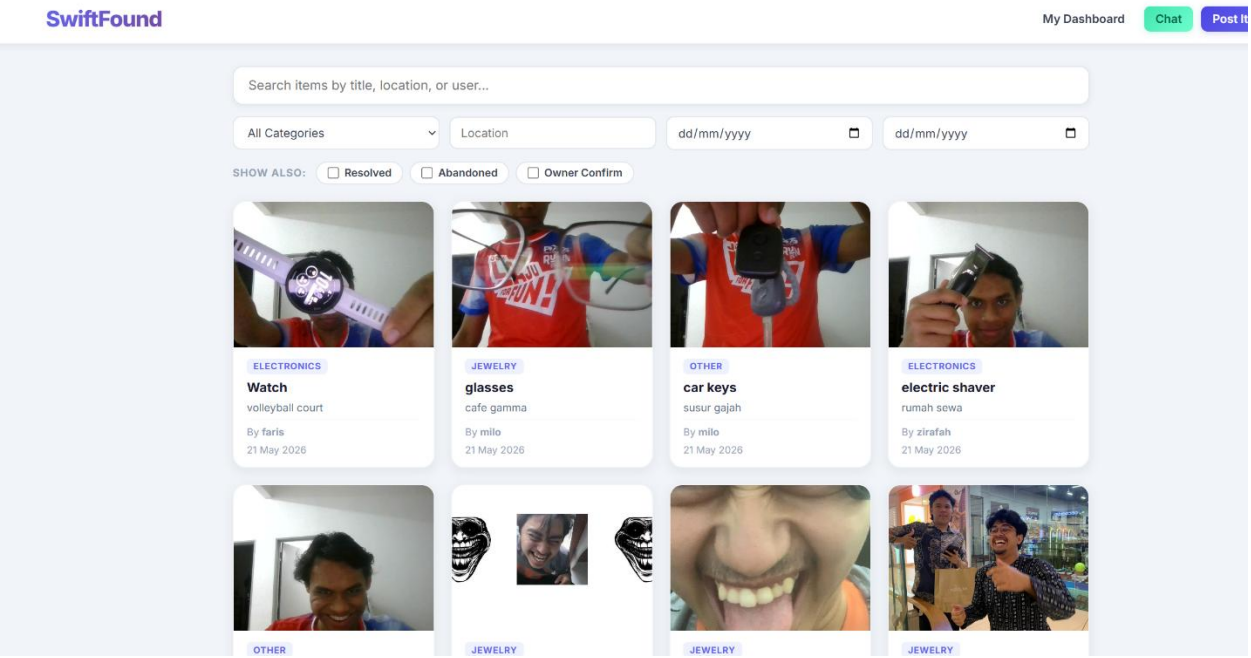

Figure 8: Browse item page

3.2.1 Data Retrieval Queries

When a user opens the browsing portal, the application hits the database using a relational data join query to pair active items with their respective poster reputation details.

SQL query: SELECT item.*, user.user_id, user.username, user.reputation FROM item, user WHERE item.user_id = user.user_id AND item.status != 'REMOVED'

3.2.2 Client-Side Search and Filters

Once the unified payload is returned to the client browser, users can refine their view using the top interface bar inputs. All search and filtering mechanics are processed instantly at the client's JavaScript layer without needing to resend query requests to the database:

- **Search Field Input:** Type text matching a *Title*, *Location*, or posting *Username*.
- **Dropdown Selection:** Filter items by structural categories (e.g., Electronics, Jewelry).
- **Date Pickers:** Restrict the grid to specific date ranges (dd/mm/yyyy).
- **Status Checkboxes:** Toggle visibility parameters to look up Resolved, Abandoned, or Owner Confirm items.

3.3 Item Detail View States

Clicking on any card inside the browsing feed loads the dedicated item details screen. The platform checks the active session credentials and changes the interface layout automatically across four distinct viewing states, each presenting specific interactive options and action buttons. The 4 states are USER_VIEW, USER_POSTED, USER_CLAIMED, ADMIN_REVIEW

3.3.1 Standard Finder Listing (USER_VIEW State)

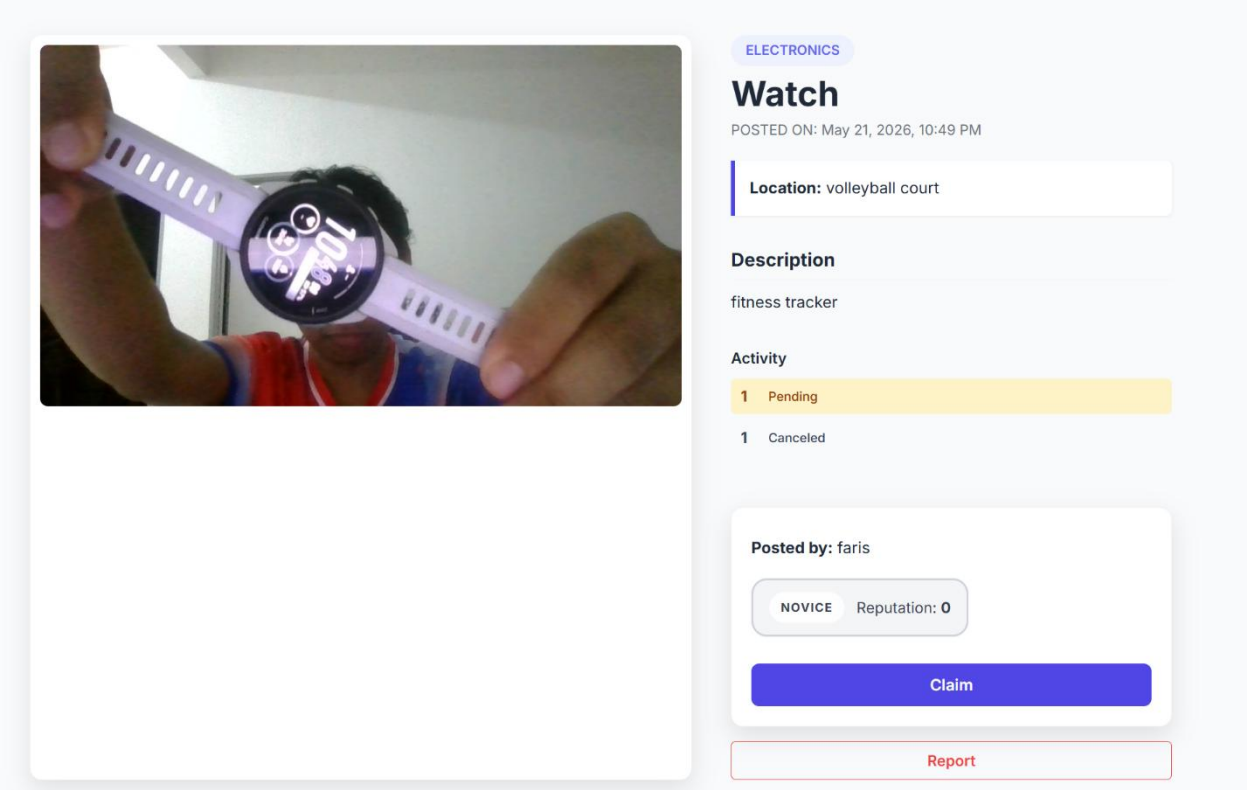

Figure 9: Item detail page for state = USER_VIEW

This state triggers when a campus resident views an item posted by another user that they have not yet interacted with. It is designed to allow users to verify if the found property belongs to them.

- **Interface Components:** Displays the full picture of the object, category, date/time posted, description, and the finder's username alongside their current community reputation badge.
- **Available Buttons & Functions:**
    - **Claim Button:** Submits a verification request to initiate the recovery process.
    - **Report Button:** Flags the item listing for administrative review if the image or description is inappropriate or suspected to be a scam.

System Access Check: If the backend database checks reveal that your account flag is set to is_restricted = 1, the system prevents this action by disabling the **Claim** button and revealing the soft red .restricted-msg banner reading: *"Your account are restricted and not allowed to claim items"*.

3.3.2 Active Claim Item View (USER_CLAIMED State)

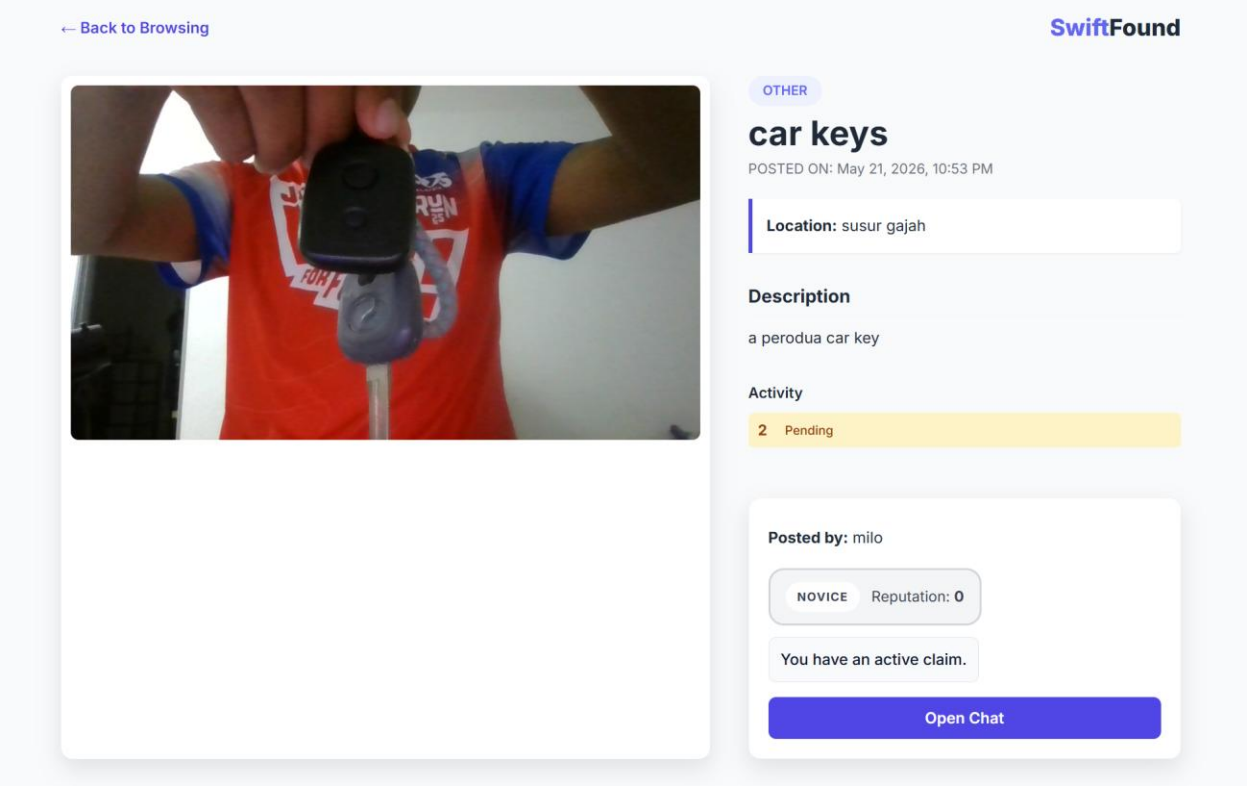

Figure 10: Item detail page for state = USER_CLAIMED

This state loads when a resident views an item detail page for a lost object where they have already submitted an active claim request.

- **Interface Components:** Displays the standard item details alongside a yellow activity status badge highlighting the active claim tier (e.g., 2 Pending).
- **Available Buttons & Functions:**
  - **Open Chat Button:** Launches the direct, real-time messaging window. This connects the claimer with the finder to answer the required secret verification question and coordinate a physical meeting spot on campus.

3.3.3 Item Owner Controls (USER_POSTED State)

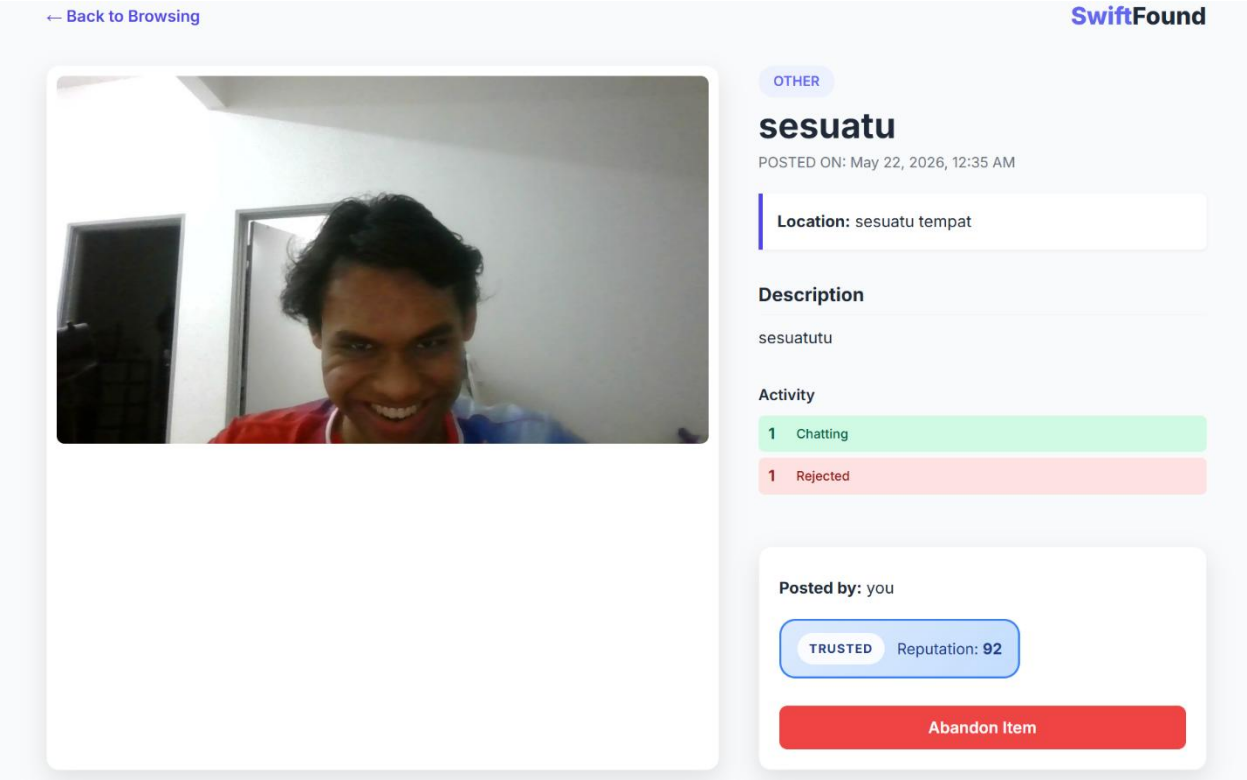

Figure 11: Item detail page for state = USER_POSTED

This layout triggers when a campus resident views an item detail page for an object that they personally discovered, logged, and uploaded to the platform.

- **Interface Components:** Displays the posted item data alongside a status grid tracking incoming community interest threads (e.g., a green indicator showing 1 Chatting or a soft red badge showing 1 Rejected).
- **Available Buttons & Functions:**
    - **Abandon Item Button:** A red action button that cancels all ongoing claim inquiries for that item, closes active chat channels, and safely pulls the listing out of the active community pool.

3.3.4 Moderation Review Overlay (ADMIN_REVIEW State)

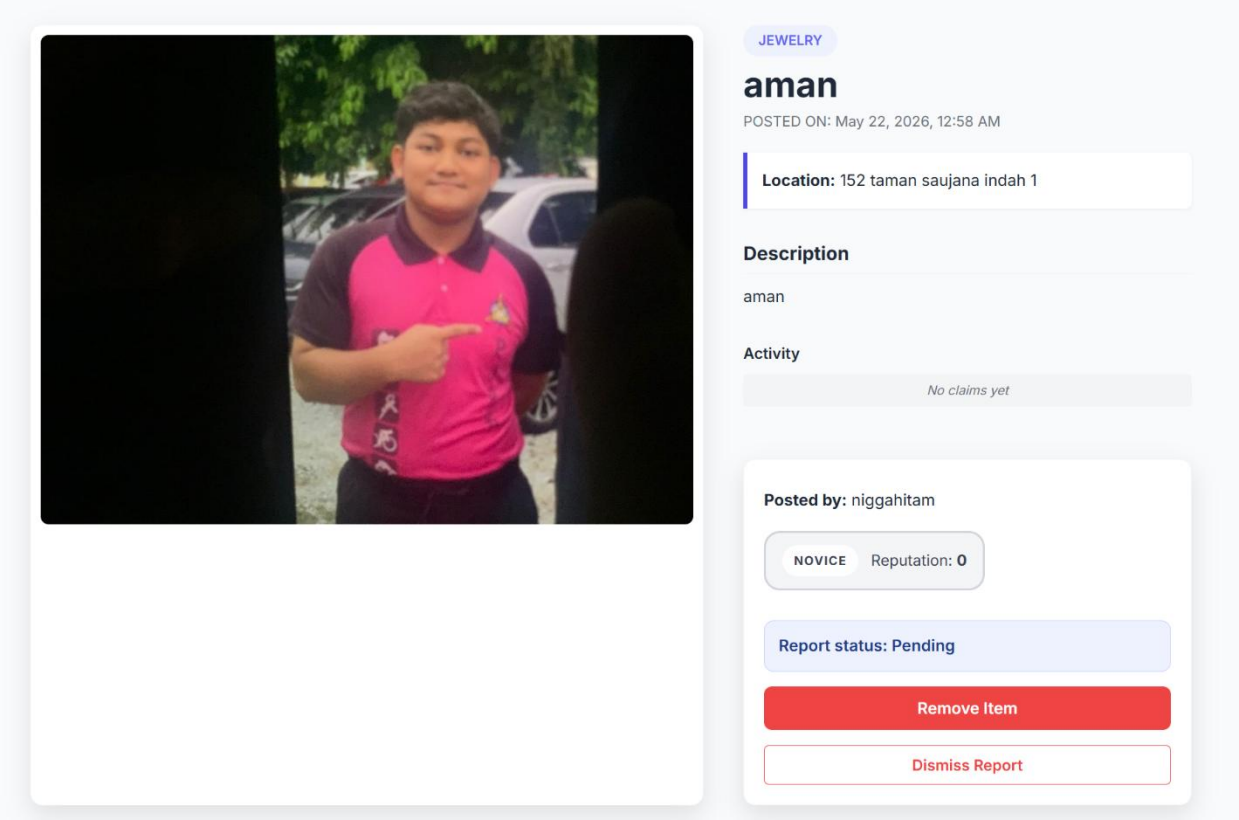

Figure 12: Item detail page for state = ADMIN_REVIEW

This specialized view overrides the standard layout when an administrator follows an infraction flag from the user reports queue to inspect a disputed listing.

- **Interface Components:** Loads the dark-themed administrative theme, featuring a permanent status block that displays Report status: Pending directly above the control panel.
- **Available Buttons & Functions:**
  - **Remove Item Button:** A red management button that instantly wipes the listing from the public browse registry by shifting its database state to REMOVED.
  - **Dismiss Report Button:** A bordered button that clears the active case flag from the moderation queue and archives the incident file without changing any user parameters.

3.4 Claim Messaging and Flow

Once a campus resident initializes a claim request for a found object, the application establishes a secure conversation thread bound to that specific item claim ID. The entire workflow—from initial answer review to final physical verification—is managed dynamically across eight distinct

lifecycle status filters: PENDING, CHATTING, OWNER_CONFIRM, PENDING_RESOLUTION, RESOLVED, REJECTED, CANCELED, and ABANDONED.

Claim Initial Intake Phase (PENDING State)

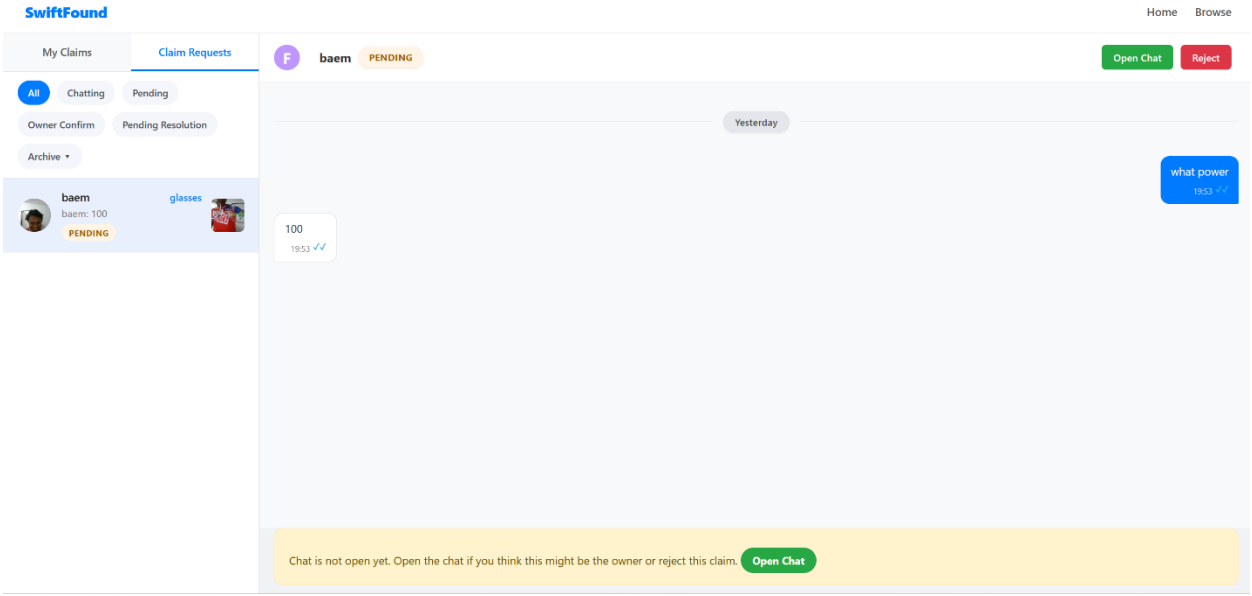

Figure 13: Chat page for a pending claim (poster side)

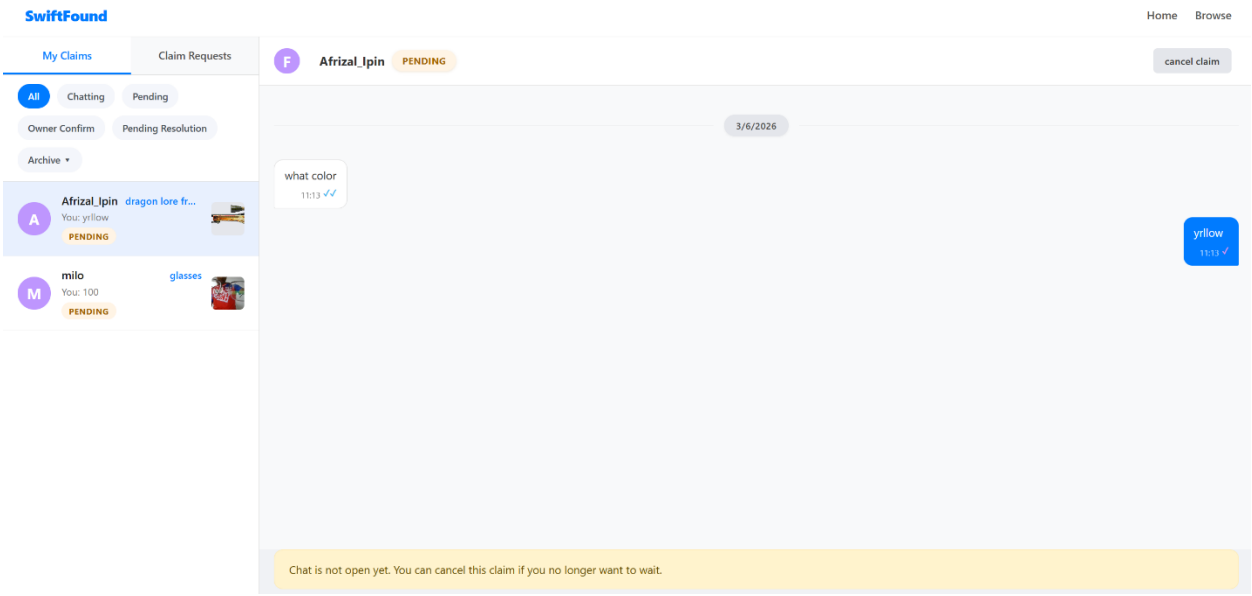

Figure 14: Chat page for a pending claim (claimer side)

The conversation thread begins in the **PENDING** state. At this stage, text message boxes are restricted, and the chat engine evaluates the initial verification responses.

- **Poster View:** The item creator reviews incoming claims under the **Claim Requests** tab. The conversation space is locked by a bottom banner reading: *"Chat is not open yet. Open the chat if you think this might be the owner or reject this claim."*

- **Available Buttons:**
  - **Open Chat:** Moves the single claim into an active discussion loop.
  - **Reject:** Instantly sets the claim status to REJECTED, closing the thread.

- **Claimer View:** The individual looking for their lost item monitors the progress under the **My Claims** tab. The chat input is hidden, and a warning banner states: *"Chat is not open yet. You can cancel this claim if you no longer want to wait."*
  - **Available Buttons:**
    - **Cancel Claim:** Sets the status to CANCELED, removing the user from the finder's queue.

3.4.1 Verification & Match Phase (CHATTING State)

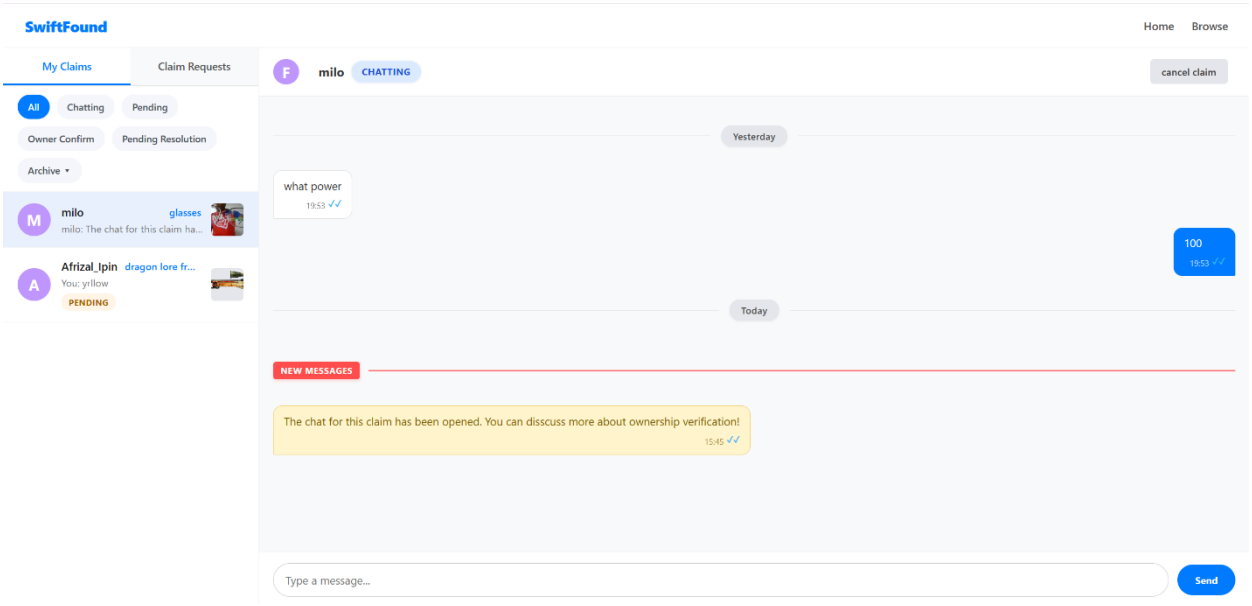

Figure 15: Chat page for a chatting claim (claimer side)

When the item finder selects **Open Chat**, the workspace transitions immediately to the **CHATTING** state.

Multi-Claim Concurrency Rule: The system allows an item finder to simultaneously open chat streams with multiple independent claimers for a single item. However, the moment a definitive match is verified and confirmed, an automated server-side trigger executes, instantly changing all other competing active claims for that item ID to REJECTED.

- **Poster View:** A yellow system alert system block declares: *"The chat for this claim has been opened. You can discuss more about ownership verification!"* The bottom text entry bar unlocks.
  - **Available Buttons:**
    - **Confirm Owner:** Clicked when the finder is completely satisfied that the chat partner is the real owner. This advances the claim state.
    - **Reject:** Closes the thread if conversation reveals the claimer is guessing or fraudulent.

- **Claimer View:** Unlocks the standard message text area and send function to chat directly with the finder to set up a meeting venue.
    - **Available Buttons:**
        - **Cancel Claim:** Abandons the active chat thread if the user realizes the item isn't theirs.

3.4.2 Handover Confirmation Phase (OWNER_CONFIRM State)

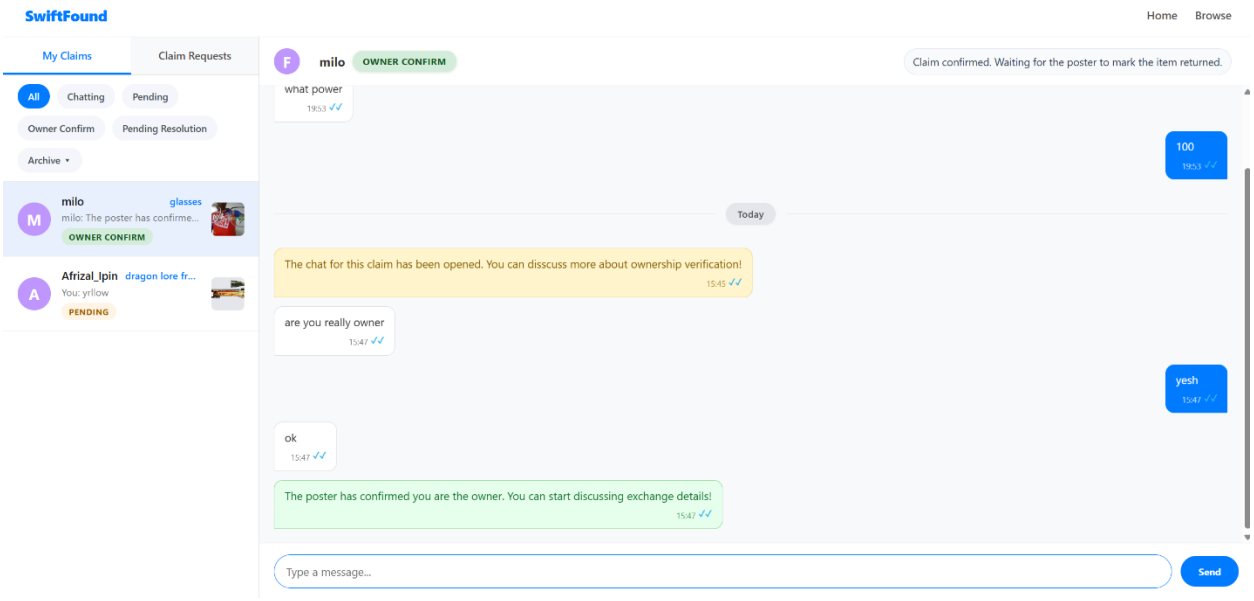

Figure 16: Chat page for a owner confirm claim (claimer side)

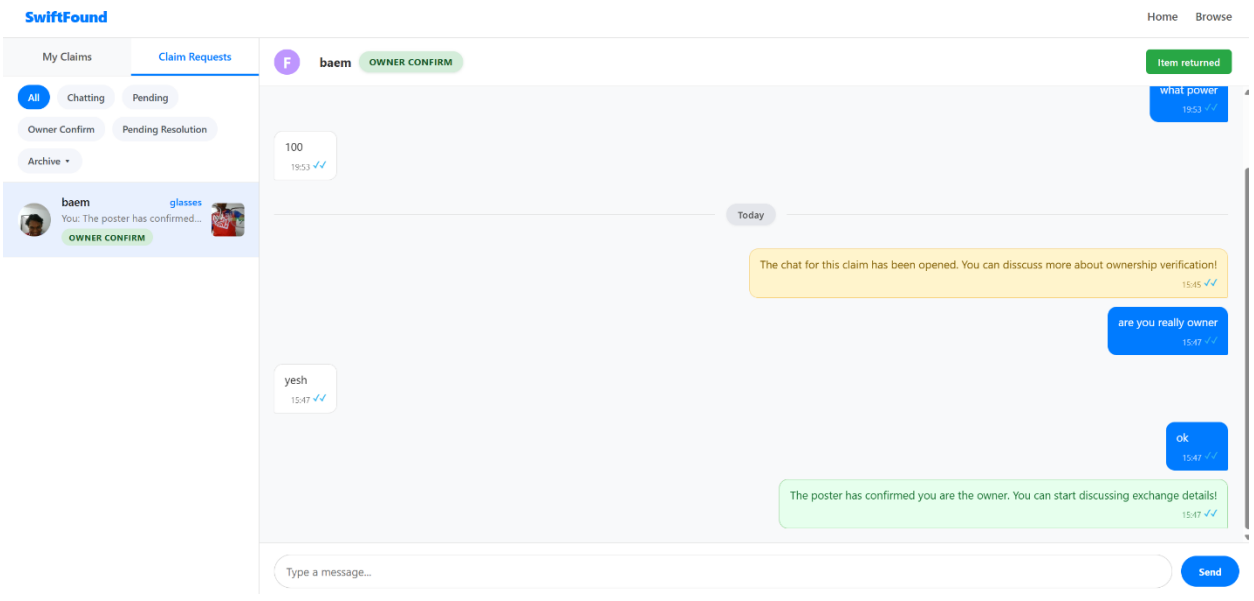

Figure 17: Chat page for a owner confirm claim (poster side)

Once the finder selects *Confirm Owner*, the system locks down competing threads and moves this specific claim into the **OWNER_CONFIRM** state to prepare for the physical item handover.

- **Poster View:** The interface shifts to direct exchange coordination. A green information box confirms: *"The poster has confirmed you are the owner. You can start discussing exchange details!"*
    - **Available Buttons:**
        - **Item Returned:** Clicked by the finder *only* after they physically handover the property to the claimer on campus. Selecting this pushes the workflow into the verification gate.
- **Claimer View:** The top banner displays: *"Claim confirmed. Waiting for the poster to mark the item returned."* The user uses the open chat box to coordinate meeting the finder in person.

3.4.3 Resolution Verification Gate (PENDING_RESOLUTION State)

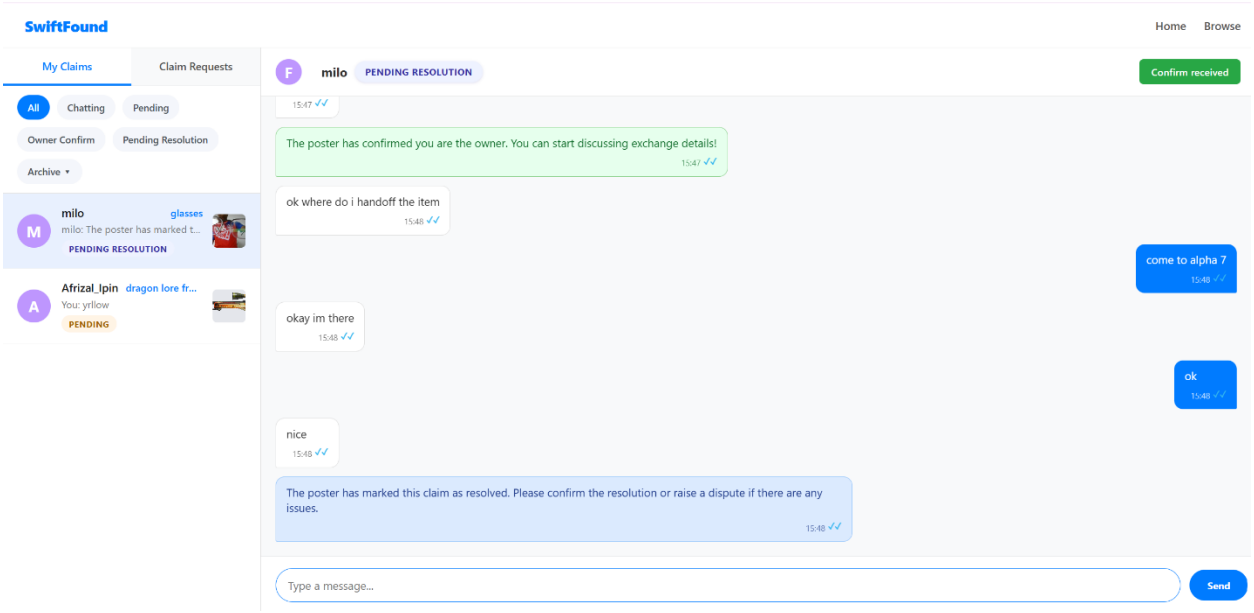

Figure 18: Chat page for a owner confirm claim, after the poster marked item returned (claimer side)

The moment the finder clicks *Item Returned*, the platform routes the claim state to **PENDING_RESOLUTION**. This gate prevents dishonest finders from claiming successful handovers without the owner's verification.

- **Poster View:** The message container shows an updated system prompt block: *"The poster has marked this claim as resolved. Please confirm the resolution or raise a dispute if there are any issues."* The top header tracking block locks with the notice: *"Pending receipt confirmation from the claimer."*
- **Claimer View:** The platform provides a green terminal action interface directly over the conversation screen.
    - **Available Buttons:**

- **Confirm Received:** Clicked by the property owner to formally sign off on the recovery. This increments both users' community reputation parameters and shifts the claim permanently into the **RESOLVED** archive folder, freezing the chat interface from further inputs.

3.4.4 Lifecycle Archival Phase (RESOLVED State)

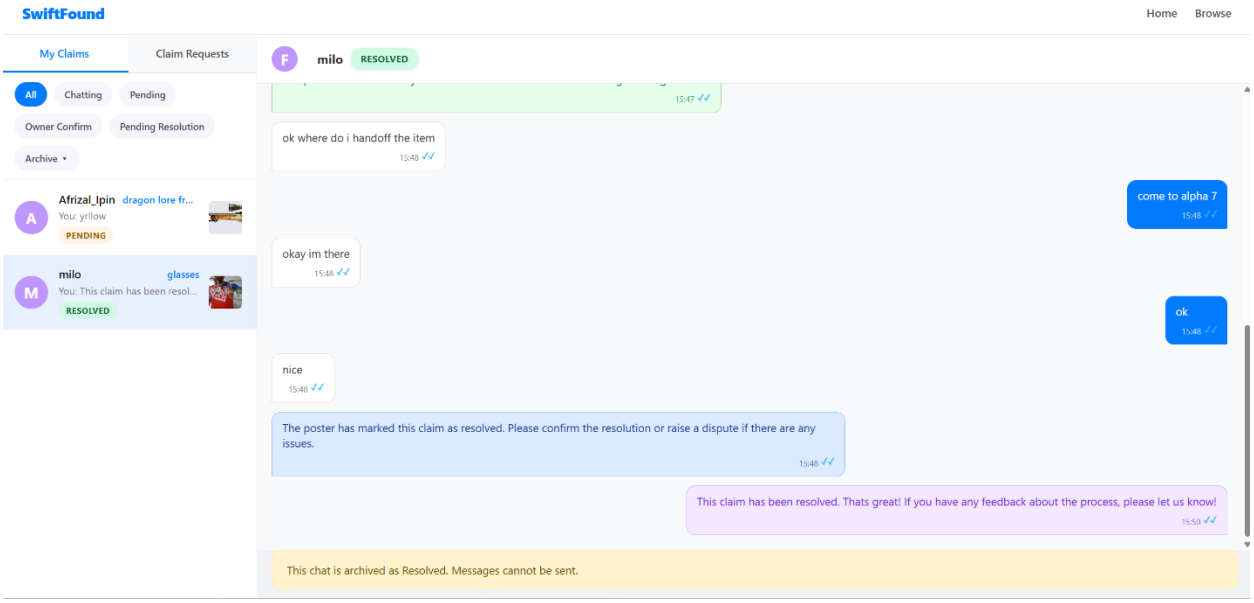

Figure 19: Chat page for a resolved claim, after the claimer confirmed resolution (claimer side)

When the claimer clicks *Confirm Received*, the system processes the final lifecycle update, moving the record into the **RESOLVED** state.

- **System-Produced Output:** The chat input is permanently removed. A purple system banner outputs: *"This claim has been resolved. That's great! If you have any feedback about the process, please let us know!"* A bottom yellow status layout notice secures the room: *"This chat is archived as Resolved. Messages cannot be sent."* Both users' dashboard counters increment their **Resolved** stat block by +1.

3.4.5 Archived Messages (CANCELED State)

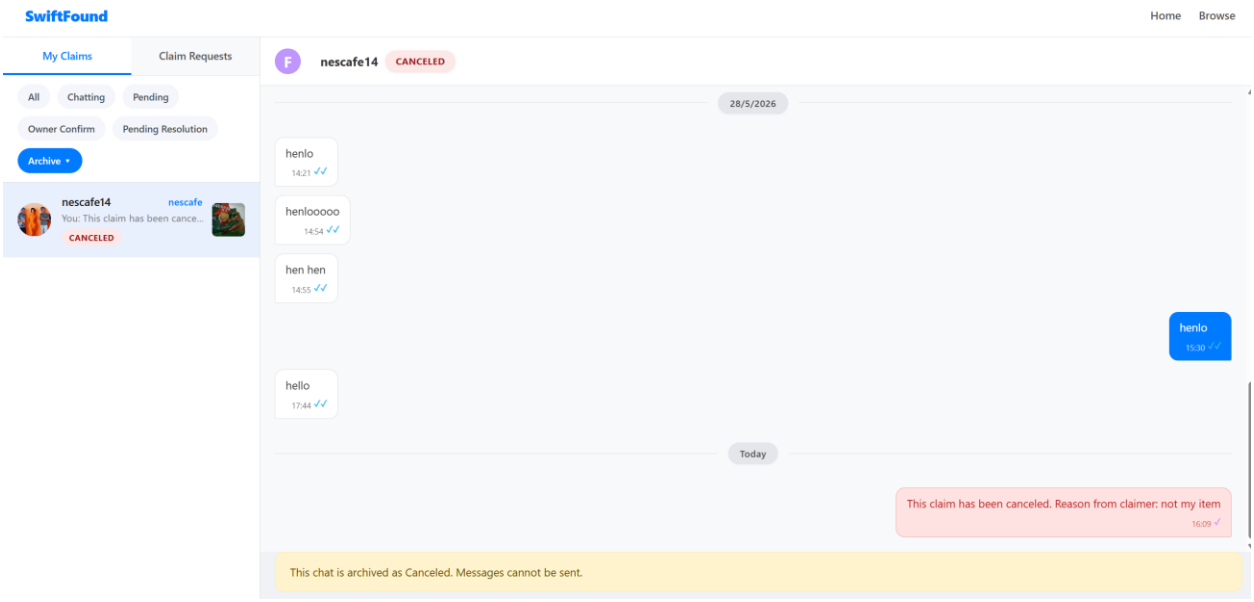

Figure 20: Chat page for archived CANCELED claim

This layout triggers when a claimer decides to pull their own recovery application out of a finder's queue.

- **Trigger Condition:** The individual tracking their lost item clicks **Cancel Claim** because they realized the item description does not match their missing property, or they have already recovered it elsewhere.
- **System-Produced Output:** The window appends a red message notification block stating the cancellation reason (e.g., *"This claim has been canceled. Reason from claimer: not my item"*). The input bar is replaced with a yellow alert block: *"This chat is archived as Canceled. Messages cannot be sent."*

3.4.6 Archived Messages (REJECTED State)

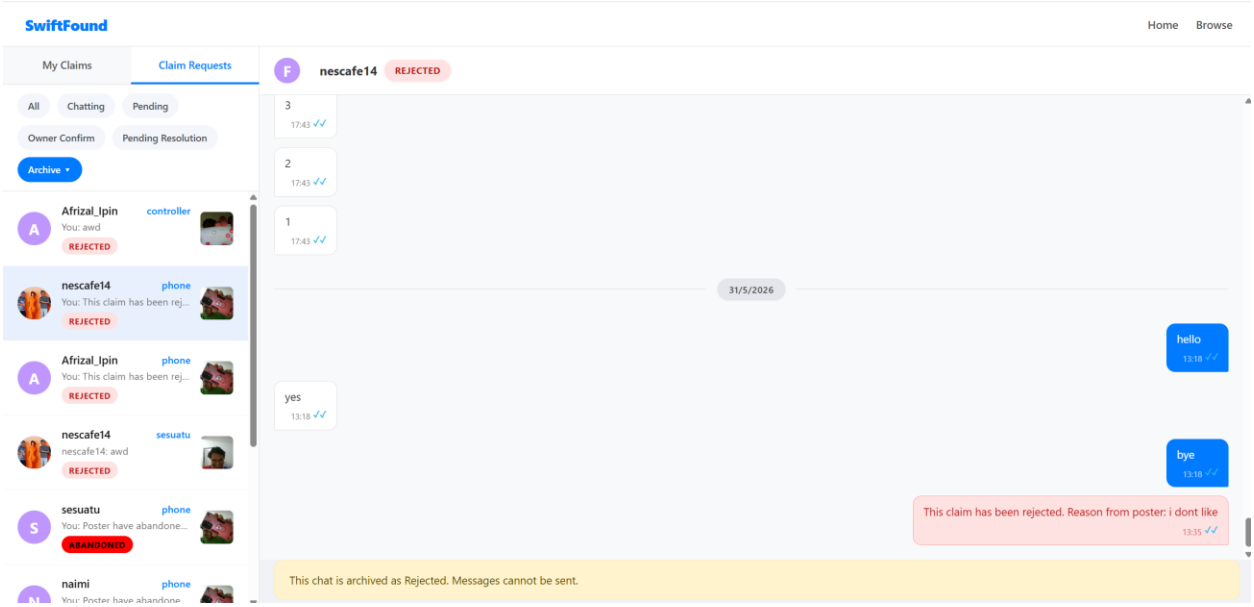

Figure 21: Chat page for archived REJECTED claim

This state occurs when an item finder reviews an open claim thread and determines that the claimer is not the rightful owner of the property.

- **Trigger Condition:** The item finder explicitly clicks the **Reject** button during either the initial evaluation (PENDING state) or an active conversation (CHATTING state).
- **System-Produced Output:** The system freezes the messaging module and outputs a red status card into the chat timeline containing a structured explanation from the finder (e.g., *"This claim has been rejected. Reason from poster: you not owner >:("*). A permanent bottom banner locks the view: *"This chat is archived as Rejected. Messages cannot be sent."*

3.4.7 Archived Messages (ABANDONED State)

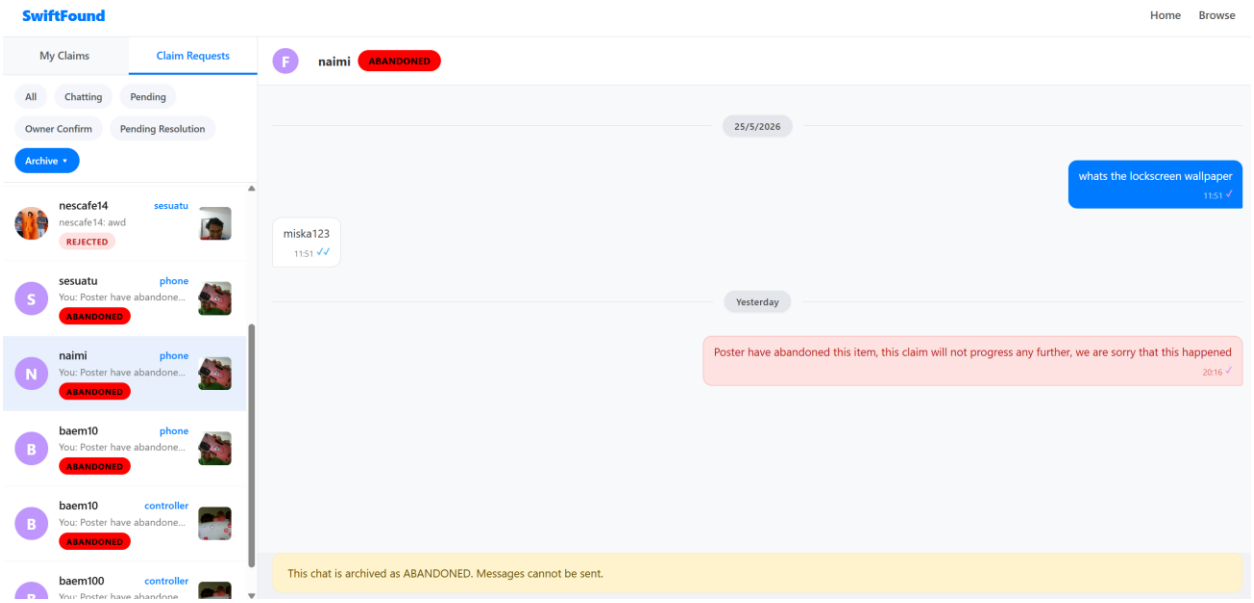

Figure 22: Chat page for archived ABANDONED claim

This terminal configuration applies globally when a finder decides they can no longer store or maintain a found item listing on the campus network.

- **Trigger Condition:** The item creator navigates to their main item control panel and clicks **Abandon Item**.
- **System-Produced Output:** The application immediately closes all active, pending, or chatting threads tied to that specific item asset ID. Each affected claimer receives an identical red notification block stating: *"Poster have abandoned this item, this claim will not progress any further, we are sorry that this happened"*. The screen updates globally with the permanent archival badge: *"This chat is archived as ABANDONED. Messages cannot be sent."*

3.5 Admin Item Reports Moderation and User Restriction

3.5.1 The User Reports

When an administrator clicks on **User Reports** in the left sidebar, the system executes a backend retrieval to gather all community-submitted incident logs.

1. Interface Information & Fields

The queue displays detailed transaction cards for every logged complaint. Each report block breaks down into the following structured data fields:

- **Reporter:** The username of the resident who initialized the infraction flag (e.g., baem101).

- **Target:** Multi-link badges specifying the reported **Item** (e.g., glasses) and the **Reported User** who posted it (e.g., milo).
- **Reason:** A high-level category tag highlighting the nature of the rule violation (e.g., scam).
- **Details:** Explicit descriptive text typed by the reporter outlining the problem (e.g., "picture wierd").
- **Admin Note:** A persistent text tracking field showing current auditing milestones (defaults to "not reviewed yet").
- **Dates:** A precise timestamp tracking exactly when the incident entry hit the database network.

2. Execution Procedures

To process an open incident from the queue, the admin performs the following initial steps:

1. Navigate to the **User Reports** panel.
2. Filter the queue by selecting the **Pending** top layout tab.
3. Locate the target infraction card and review the reporter's typed text details.
4. Click the **Review Item** button at the bottom of the card. This action dynamically routes the administrator to the corresponding item detail view in its dedicated moderation state.

3.5.2 Moderation Actions (ADMIN_REVIEW State)

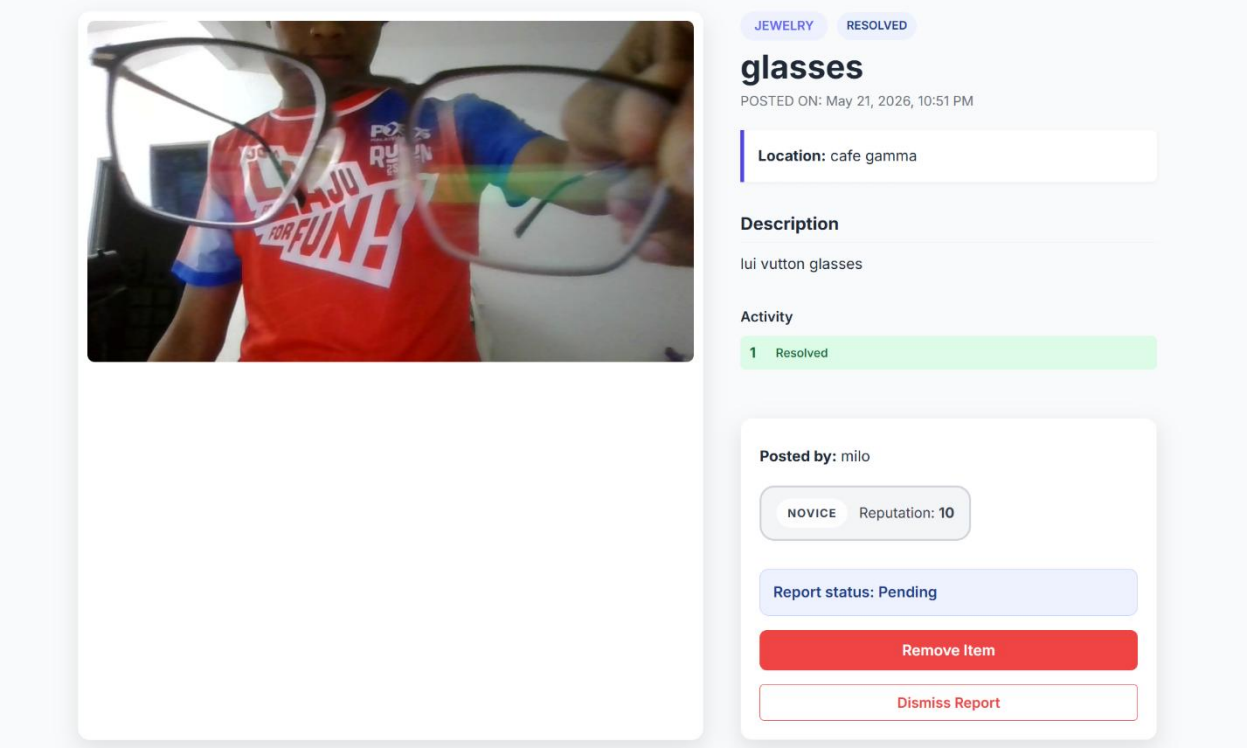

Figure 23: Item detail page for viewing status = ADMIN_REVIEW

Upon clicking *Review Item*, the platform shifts the item detail screen into the **ADMIN_REVIEW** state. This layout provides an isolated verification interface containing a permanent blue indicator banner reading: Report status: Pending.

From this view, the administrator has two distinct options to resolve the active incident:

1. Dismissing the Incident (Dismiss Report Button)

If the administrator audits the listing picture, title, and description and determines that the post is legitimate and does not break campus guidelines:

- **Action:** Click the white **Dismiss Report** button.
- **System-Produced Output:** The platform updates the database record state to DISMISSED. This action removes the card from the active moderation queue, clears the alert banner from the item, and leaves both the post and the creator's data completely unchanged.

2. Accepting the Infraction (Remove Item Button)

If the listing is verified to be a fraudulent scam, inappropriate image upload, or intentional system abuse:

- **Action:** Click the red **Remove Item** button.
- **System-Produced Output:** The server automatically fires a sequence of multi-table database operations:
    1. It sets the target listing's status field to REMOVED, permanently hiding it from the public community browsing feed.
    2. It flags the incident report status as ACCEPTED within the moderation logs.
    3. It applies a penalty directly to the creator's account record, deducting points from their community reliability metrics and updating their visual **Reputation Progress Bar** accordingly.

3.5.3 Admin User Registry & Restriction Management

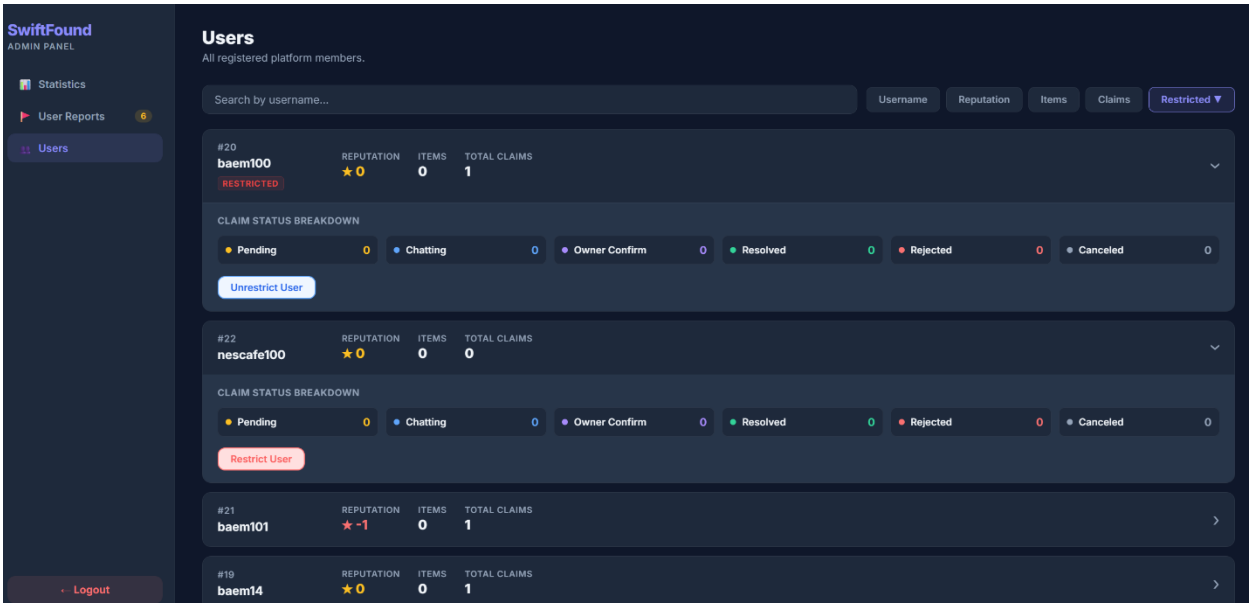

Figure 24: Admin User Registry

1. The Users Registry Interface

Clicking on **Users** in the Admin Panel left sidebar fetches the master system member directory. The top management bar allows operators to search specific community members instantly using text filters or toggle the view state via dedicated sorting buttons: **Username**, **Reputation**, **Items**, **Claims**, and account state categories like **Restricted**.

Each user block displays a clean, high-level dataset:

- **Identification Parameters:** Displays the unique database registry number (e.g., #20, #22) and account username.
- **Core Metrics Blocks:** Tracks live user telemetry data, including their active community **Reputation** score, total **Items** posted, and **Total Claims** initiated.
- **Claim Status Breakdown Dropdown:** Expanding a user's record grid pulls a real-time analytics breakdown showing exactly how many of their individual claims are currently sitting in Pending, Chatting, Owner Confirm, Resolved, Rejected, or Canceled operational phases.

2. Executing an Account Restriction (Restrict User Button)

When a campus resident repeatedly triggers community incident logs, spams fake items, or displays malicious behaviour within chat sessions, the administrator can apply a severe platform restriction.

- **Procedure:** Locate the malicious user's block card within the registry grid and click the red **Restrict User** button.
- **System-Produced Output:** The server instantly toggles the target row parameters on the backend database. This action alters their status, appending a red **RESTRICTED** warning tag right below their username.
- **Resulting Accessibility Constraints:** The moment this state override applies, the user's active session is flagged. The platform automatically blocks them from initiating

any new item claims, freezes their ability to list found property, and displays a permanent soft red warning banner across their personal graphical interface layout.

3. Revoking an Account Restriction (Unrestrict User Button)

If a restricted resident submits a successful appeal or resolves their community standing with a campus administrator, their standard account privileges can be reinstated.

- **Procedure:** Filter the directory using the **Restricted** category button to isolate penalized accounts. Select the target profile and click the white **Unrestrict User** button.
- **System-Produced Output:** The system instantly clears the restrictive flag, wiping the red *RESTRICTED* badge from their profile block.
- **Resulting Accessibility Reinstatement:** The resident's platform permissions are completely restored in real time. The warning banner disappears from their dashboard view, and they are immediately cleared to use the standard lost-and-found browse registry, claim items, and interact with community chat threads normally.

4. Troubleshooting & Support

This section outlines the platform recovery paths, error states, and systemic guardrails designed to guide users when an operation fails or requires authentication validation.

4.1 Error Messages

When an unauthenticated session attempts to bypass the landing screen and access core application actions, the system enforces strict routing loops to protect data integrity. The table below details these scenarios and the required user actions:

Table 1: Possible errors user might encounter

Error Condition / Action Triggered	Likely Cause	System-Produced Output & Corrective Action
Unauthenticated Item Posting	A user attempts to navigate directly to the post-item form URL or clicks a posting link without an active session token.	The system completely terminates the form initialization loop. It forces an immediate redirect back to the **Login Page**, requiring valid resident credentials before rendering the form fields.
Unauthenticated Item Claiming	An anonymous visitor attempts to click the **Claim** button on the public browsing feed or inputs an explicit claiming endpoint URL string.	The application interceptor blocks the request. The browser immediately redirects the user to the **Login Page** to verify community identity before processing claim queues.
Account Restricted Warning Block	A flagged resident clicks the **Claim** button on a live item card while their account status is locked.	The claim execution routine fails. The page renders a soft red layout banner stating: Your account are restricted and not allowed to claim items. No further actions can be taken unless an administrator removes the flag.
Invalid Master Key Authentication	An operator inputs an incorrect string or an expired code inside the Admin Portal *Security Key* field.	The interface rejects the payload, blocks access to the management grids, and retains the portal lock state. The operator must re-enter the correct, active master code.

4.2 Special Considerations

To ensure data consistency and avoid common troubleshooting confusion, keep the following environmental conditions in mind:

- **Session Token Cache:** If you log out or if your network connection drops unexpectedly, your local browser token clears out. If you are suddenly kicked back to the login screen while drafting a lost-item post, verify your internet connection and log back in.
- **Concurrent Chat Termination:** If you are actively talking to a finder about an item, and that item suddenly disappears from your *My Claims* tab or moves instantly to **REJECTED**, it means the finder has already selected *Confirm Owner* on a competing claim. The system automatically handles this data cleanup behind the scenes.

4.3 Support

Because this application version does not feature automated email recovery portals or self-service password reset tools, all technical issues, credential losses, or account restriction disputes must be handled directly through the campus system administrator.

Incidents & Escalation Procedures

1. **Password Failures:** If you lose access to your resident username, do not attempt to register a new account. Contact support to perform a manual credential update on the server database.
2. **Reporting Abuse:** If an interaction inside a chat window becomes toxic or a scam post bypasses initial filters, use the built-in **Report** button to route the issue straight to the Admin Panel report queue.

Support Points of Contact

The table below displays the technical contact grid for the platform management ecosystem:

Table 2: Support Points of Contact

Contact	Organization	Phone	Email	Role	Responsibility
Ibrahim Ulwan	SwiftFound	0194460641	Baraimz14@gmail.com	Development Team Leader	Lead the development team

Appendix A: Record of Changes

Table 3: Record of Changes

Version Number	Date	Author/Owner	Description of Change
0.1	05/12/2026	Ibrahim Ulwan	Developed foundational document structures, layout requirements, and Section 1 system overview definitions.
0.5	05/12/2026	Ibrahim Ulwan	Appended detailed Section 3 graphics, user interface walkthrough sequences, and database SQL query descriptions for core workflows.
1.0	06/07/2026	Ibrahim Ulwan	Finalized Section 4 Troubleshooting matrix, support matrices, chat lifecycle archival specifications, and finalized compliance checks.

Appendix B: Glossary

Table 4: Glossary

Term	Acronym	Definition
ABANDONED	N/A	A terminal archive state triggered when a finder deletes or stops hosting a found item, instantly closing all associated user chat lines.
ADMIN_REVIEW	N/A	The specialized interface viewing state that activates when an operator uses the master security key to audit a flagged item report.
CANCELED	N/A	An archival chat status applied when a property owner manually retracts their own active recovery application from a queue.
CHATTING	N/A	The active communication state where text messaging inputs unlock to let users negotiate ownership parameters and meeting venues.
OWNER_CONFIRM	N/A	A verification milestone triggered when a finder authorizes a claimant as the true owner, locking down all other competing threads for that item.
PENDING	N/A	The default entry status assigned to newly uploaded item listings and incoming claim requests awaiting initial evaluation.
PENDING_RESOLUTION	N/A	The final verification gate triggered after a finder hands over an object, requiring the receiver to click and confirm receipt.
REJECTED	N/A	An archival status assigned when a claim is denied by a finder or systematically discarded due to another claimant being confirmed.
RESOLVED	N/A	The permanent, successful archive state achieved when an owner confirms safe receipt of their lost property, incrementing user reputation scores.
SwiftFound Platform	SF	The overarching web application software ecosystem built to facilitate real-time lost-and-found matching on campus.
User Manual	UM	This tracking and operational document created to outline navigation streams, code execution strings, and interface states.