FoundIt – Lost & Found Web Application

Problem Statement

Every day, people lose valuable items in public spaces—campuses, workplaces, transit systems—and struggle to recover them. Traditional systems for managing these items are inefficient, slow, and lack transparency. This project, 'FoundIt', aims to bridge that gap through a cloud-based platform where users can report lost and found items, with automated matching powered by AWS Rekognition, and real-time notifications via email or SMS.

Document Information

Project Name: FoundIt Lost & Found Web Application

Document Version: 1.2

Date: August 2, 2025

Table of Contents

Introduction
1.1. Purpose
1.2. Scope
1.3. Definitions, Acronyms, and Abbreviations

Overall Description
2.1. Product Perspective
2.2. User Classes and Characteristics
2.3. Operating Environment
2.4. General Constraints

System Features
3.1. User Portal Features
3.2. Admin Portal Features

Architecture
4.1. User Portal Architecture
4.2. Admin Portal Architecture
4.3. AWS Services Summary
4.4. Detailed Service Descriptions and Connections

Non-Functional Requirements
5.1. Performance
5.2. Security
5.3. Scalability
5.4. Usability
5.5. Reliability

1. Introduction

1.1 Purpose

The purpose of this Software Requirements Specification (SRS) document is to outline the functional and non-functional requirements for the "FoundIt" Lost & Found web application. This document will serve as a guide for the development team, ensuring that the final product meets the needs of its users and stakeholders.

1.2 Scope

The FoundIt application is a cloud-based platform designed to streamline the process of reporting and recovering lost and found items in public spaces such as campuses, workplaces, and transit systems. It provides both a user portal for reporting and tracking items and an admin portal for managing requests, matching items, and notifying users.

1.3 Definitions, Acronyms, and Abbreviations

- **EC2:** Amazon Elastic Compute Cloud
- **S3: Amazon Simple Storage Service**
- **DynamoDB:** Amazon DynamoDB (NoSQL database service)
- **Rekognition:** Amazon Rekognition (image and video analysis service)
- **SNS:** Amazon Simple Notification Service
- **AWS:** Amazon Web Services
- **SRS:** Software Requirements Specification

2. Overall Description

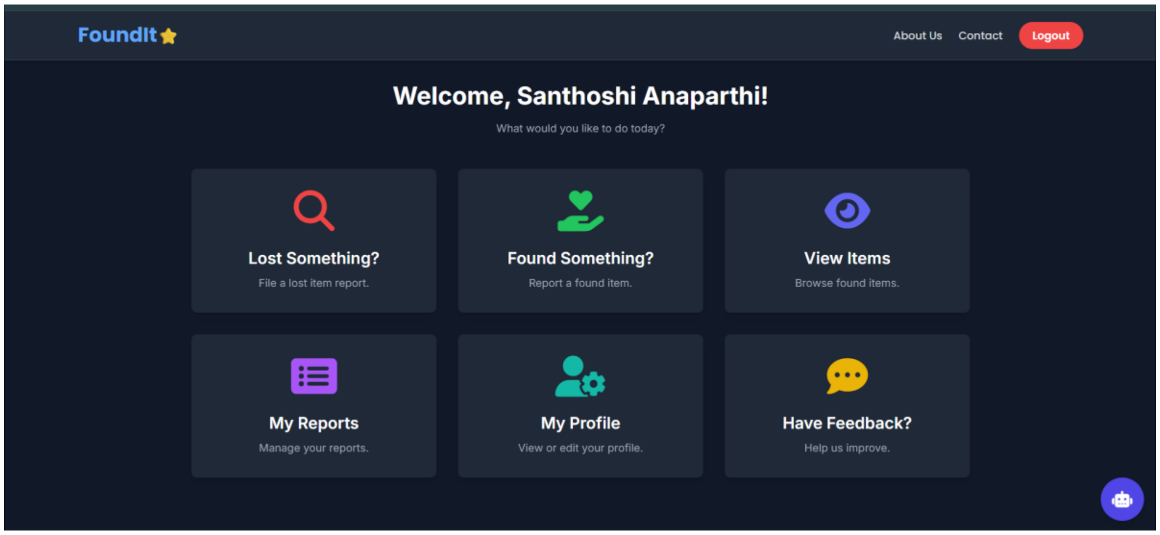
2.1 Product Perspective

FoundIt is a standalone web application built on AWS services. It aims to replace inefficient and slow traditional lost and found systems by providing a transparent and automated solution. The application is divided into two main portals: a User Portal and an Admin Portal.

User Portal

The user portal is a simple, intuitive interface designed for the public. It features a dashboard with clear navigation to report lost items, report found items, view a list of their own reported items, and provide feedback. The process for reporting is guided, allowing users to upload an image and provide a description of the item.

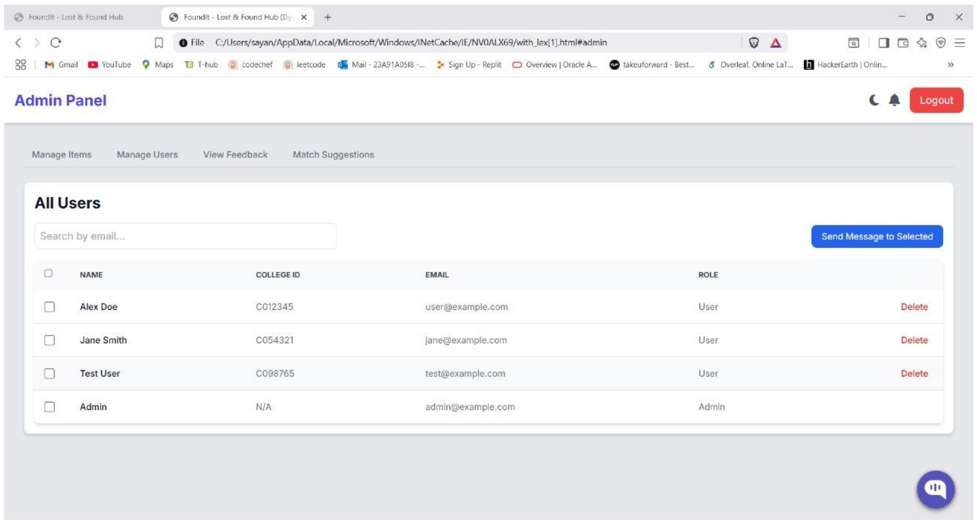
[User Portal Dashboard]



Admin Portal

The admin portal is a secure, backend interface for authorized personnel. It provides tools to manage the entire lost and found process, including viewing all reported requests, manually initiating image matching, and managing user accounts. The portal provides an organized view of all lost and found items, enabling efficient administrative tasks.

[Admin Portal Dashboard]



All Users

	NAME	COLLEGE ID	EMAIL	ROLE	
☐	Alex Doe	C012345	user@example.com	User	Delete
☐	Jane Smith	C054321	jane@example.com	User	Delete
☐	Test User	C098765	test@example.com	User	Delete
☐	Admin	N/A	admin@example.com	Admin	

2.2 User Classes and Characteristics

- **Regular Users:** Individuals who have lost an item or found an item. They will use the User Portal to report items, view their reported items, and provide feedback. They need a simple, intuitive interface for reporting and tracking.
- **Administrators:** Authorized personnel responsible for managing the lost and found process. They will use the Admin Portal to view requests, match lost and found items, manage user accounts, and send notifications. They require tools for efficient management and decision-making.

2.3 Operating Environment

The FoundIt web application will be hosted on Amazon EC2 instances. User and item data will be stored in Amazon DynamoDB, and images will be stored in Amazon S3. Image comparison will utilize Amazon Rekognition, and notifications will be sent via Amazon SNS.

2.4 General Constraints

The application's functionality is dependent on the availability and performance of the integrated AWS services.

3. System Features

3.1 User Portal Features

3.1.1 User Authentication and Authorization

- **Description:** Users can sign up or log in to the web application.
- **Functional Requirements:**

- The system shall allow users to sign up using their name, roll number, email, and password.
- The system shall allow users to log in using their credentials.
- The system shall store user credentials in the DynamoDB Users table.

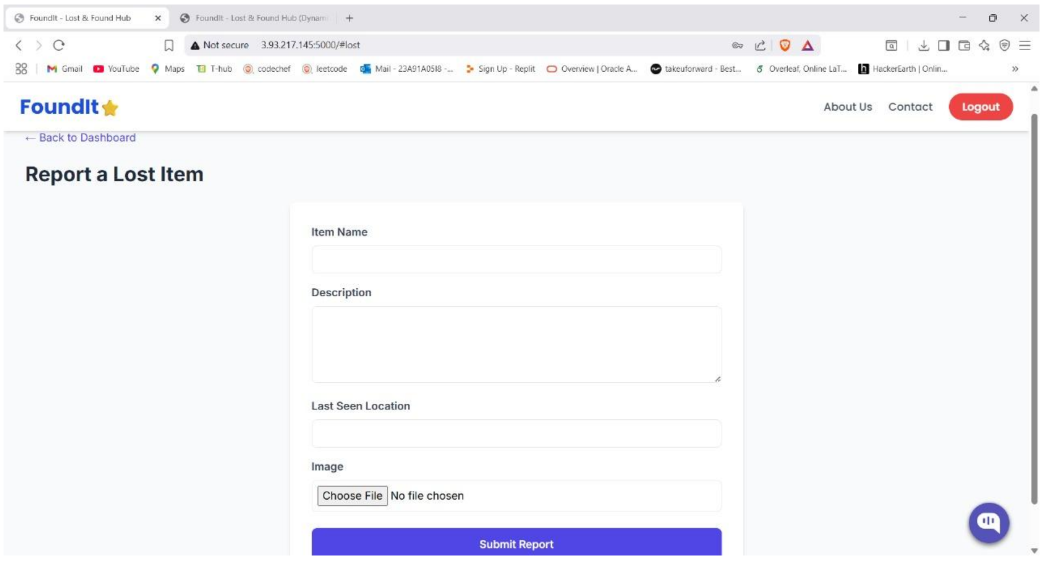
3.1.2 Dashboard View

- **Description:** After successful login, users will be presented with a dashboard.
- **Functional Requirements:**
    - The dashboard shall display options for "My Lost Items," "My Found Items," "Report Lost Item," "Report Found Item," and "Feedback".

3.1.3 Report Lost Item

- **Description:** Users can report a lost item by providing a description and an image.
- **Functional Requirements:**
    - The system shall allow users to submit a description for a lost item.
    - The system shall allow users to upload an image of the lost item.
    - The system shall store the lost item image in the S3 lost-items-images bucket.
    - The system shall store the lost item's metadata in the DynamoDB LostItems table.

[Lost Item Report Form]



FoundIt - Lost & Found Hub | FoundIt - Lost & Found Hub (Dynam

Not secure 3.93.217.145:5000/#lost

FoundIt — About Us — Contact — Logout

← Back to Dashboard

Report a Lost Item

Item Name

Description

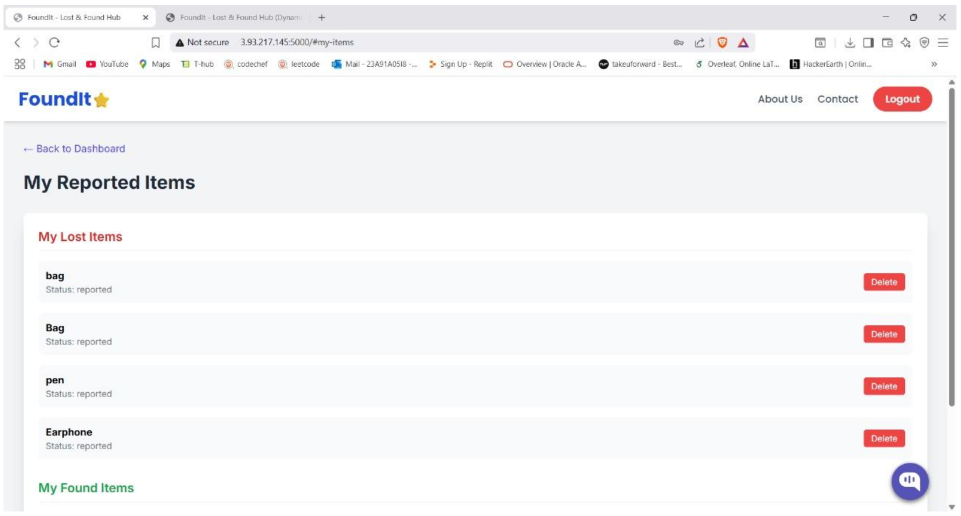
Last Seen Location

Image

Choose File No file chosen

Submit Report

[user uploading a lost item]


FoundIt - Lost & Found Hub
Not secure 3.93.217.145:5000/#my-items
FoundIt
About Us
Contact
Logout
← Back to Dashboard

My Reported Items

My Lost Items

bag
Status: reported
Delete

Bag
Status: reported
Delete

pen
Status: reported
Delete

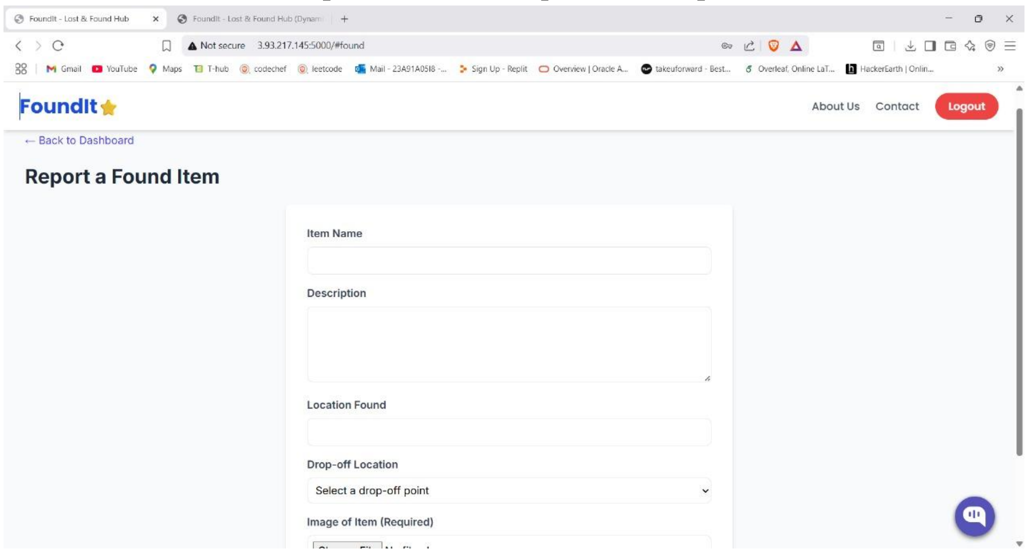
Earphone
Status: reported
Delete

My Found Items

3.1.4 Report Found Item

- **Description:** Users can report a found item by providing a description and an image.
- **Functional Requirements:**
  - The system shall allow users to submit a description for a found item.
  - The system shall allow users to upload an image of the found item.
  - The system shall store the found item image in the S3 found-items-images bucket.
  - The system shall store the found item's metadata in the DynamoDB FoundItems table.

[Found Item Report Form]



← Back to Dashboard

Report a Found Item

Item Name

Description

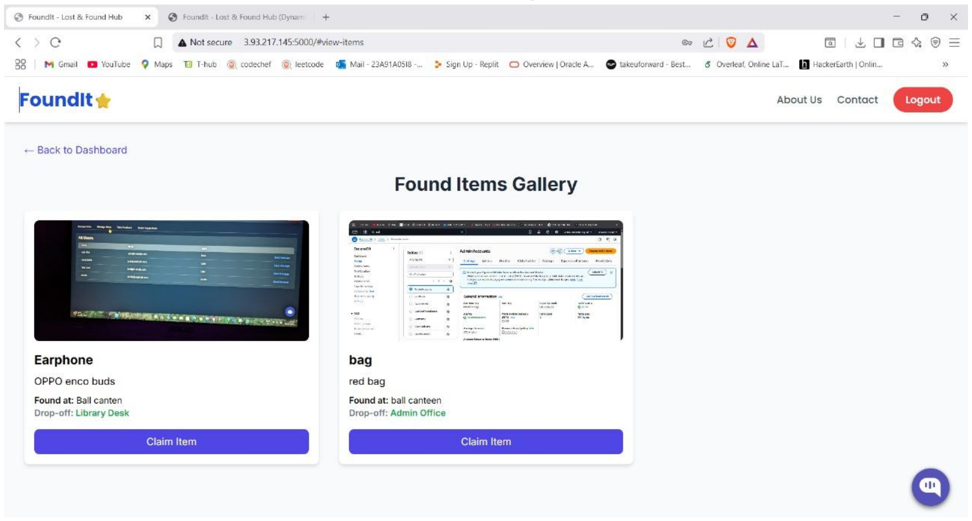
Location Found

Drop-off Location

Select a drop-off point

Image of Item (Required)

[user uploading a found]


FoundIt - Lost & Found Hub
Not secure 3.93.217.145:5000/#view-items
FoundIt
About Us
Contact
Logout

← Back to Dashboard

Found Items Gallery

Earphone
OPPO enco buds
Found at: Ball canten
Drop-off: Library Desk
Claim Item

bag
red bag
Found at: ball canteen
Drop-off: Admin Office
Claim Item

3.1.5 View My Items

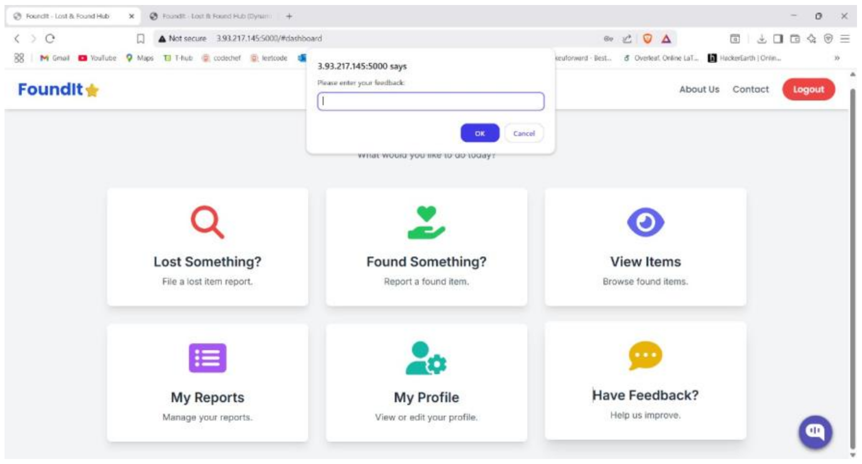
- **Description:** Users can view a list of items they have reported as lost or found.
- **Functional Requirements:**
  - The system shall query DynamoDB by user ID to list their reported lost items.
  - The system shall query DynamoDB by user ID to list their reported found items.

3.1.6 Chatbot/Feedback

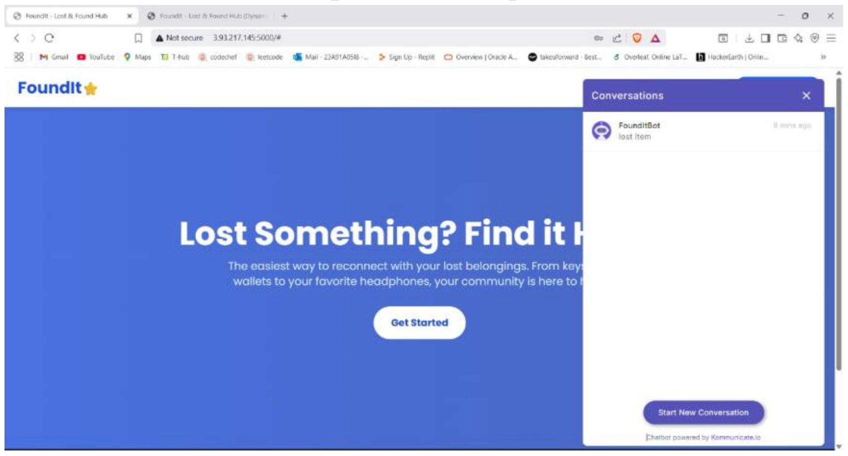
- **Description:** Users can submit feedback or interact with a chatbot.

- **Functional Requirements:**
  - The system shall save user feedback messages to the DynamoDB Feedback table.
  - (Optional) The system may integrate with AWS Lex for chatbot functionality.

[user feedback]



[user chatbot]


FoundIt
Conversations
FoundItBot
lost item
Lost Something? Find it
The easiest way to reconnect with your lost belongings. From keys
wallets to your favorite headphones, your community is here to
Get Started
Start New Conversation
Chatbot powered by Kommunicate.io

3.2 Admin Portal Features

3.2.1 Admin Login

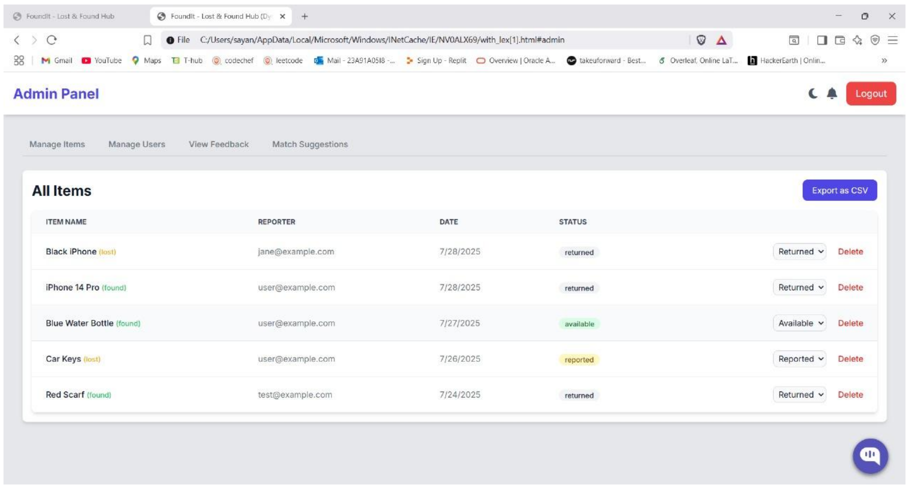
- **Description:** Administrators can log in to the admin portal using custom credentials.
- **Functional Requirements:**
  - The system shall authenticate admin users based on custom credentials.

  - The system shall check the admin type in the DynamoDB Users table for authentication.

3.2.2 Admin Dashboard

- **Description:** After successful admin login, the dashboard will display administrative options.
- **Functional Requirements:**
  - The admin dashboard shall display "View Lost Requests," "View Found Items," "Match Requests," "Send Notifications," and "Manage Users".

[Admin Dashboard]



All Items

ITEM NAME	REPORTER	DATE	STATUS		
Black iPhone (lost)	jane@example.com	7/28/2025	returned	Returned	Delete
iPhone 14 Pro (found)	user@example.com	7/28/2025	returned	Returned	Delete
Blue Water Bottle (found)	user@example.com	7/27/2025	available	Available	Delete
Car Keys (lost)	user@example.com	7/26/2025	reported	Reported	Delete
Red Scarf (found)	test@example.com	7/24/2025	returned	Returned	Delete

3.2.3 Match Requests

- **Description:** Administrators can manually select a lost and a found item to attempt a match using image comparison.
- **Functional Requirements:**
  - The system shall allow the admin to select one lost item and one found item.
  - The system shall use Amazon Rekognition to compare the images of the selected lost and found items.
  - If the confidence score from Rekognition meets or exceeds a predefined threshold, the system shall mark the items as matched in DynamoDB.
  - Upon a successful match, the system shall trigger an SNS notification to the user who reported the lost item.

3.2.4 Manage Users

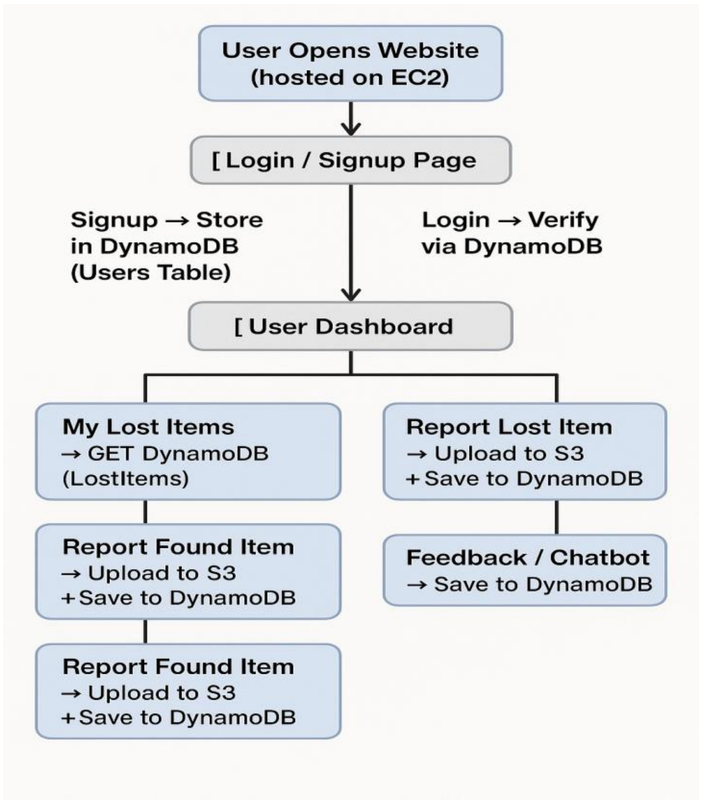
- **Description:** Administrators can view, edit, and delete user records.
- **Functional Requirements:**
    - The system shall allow administrators to view user records in DynamoDB.
    - The system shall allow administrators to edit user records in DynamoDB.
    - The system shall allow administrators to delete user records in DynamoDB.

4. Architecture

4.1 User Portal Architecture

- **Web Browser (User):** Accesses the web application.
- **Frontend (React/HTML):** Provides the user interface, hosted on EC2.
- **Backend API:** Handles logic, authentication, and submissions, hosted on EC2.
- **DynamoDB (Users Table):** Stores and verifies user credentials for signup/login.
- **S3 + DynamoDB:** Used for storing images and metadata for Lost Item Reports.
- **S3 + DynamoDB:** Used for storing images and metadata for Found Item Reports.
- **DynamoDB:** Used for querying records by user for "My Items View".
- **DynamoDB (Feedback):** Stores user feedback/chat messages.

[user working portal flowchat]

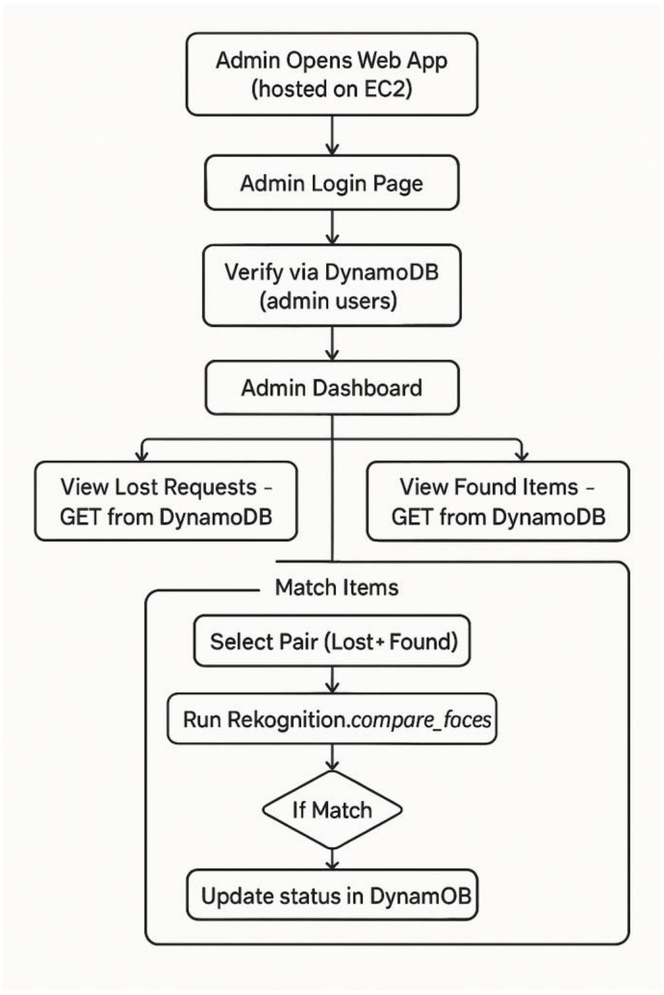

User Opens Website (hosted on EC2)
[Login / Signup Page
Signup → Store in DynamoDB (Users Table)
Login → Verify via DynamoDB
[User Dashboard
My Lost Items → GET DynamoDB (LostItems)
Report Lost Item → Upload to S3 + Save to DynamoDB
Report Found Item → Upload to S3 + Save to DynamoDB
Feedback / Chatbot → Save to DynamoDB
Report Found Item → Upload to S3 + Save to DynamoDB

4.2 Admin Portal Architecture

- **Admin Dashboard:** Admin interface, hosted on EC2.
- **DynamoDB (Users Table):** Used for admin login checks.
- **DynamoDB:** Used to retrieve all reported items for "View Requests".
- **Rekognition:** Used to compare images for "Image Matching".
- **SNS:** Used to send notifications to matched users
- **DynamoDB:** Used for viewing/editing user data in "Manage Users".

.

[admin working portal flowchat]


Admin Opens Web App (hosted on EC2)
Admin Login Page
Verify via DynamoDB (admin users)
Admin Dashboard
View Lost Requests - GET from DynamoDB
View Found Items - GET from DynamoDB
Match Items
Select Pair (Lost+Found)
Run Rekognition.compare_faces
If Match
Update status in DynamOB

4.3 AWS Services Summary

Service	**Purpose**
Amazon EC2	Host frontend/backend and admin dashboard
Amazon S3	Store images of lost and found items
Amazon DynamoDB	Store user, item, and feedback data
Amazon Rekognition	Compare item images to find a match

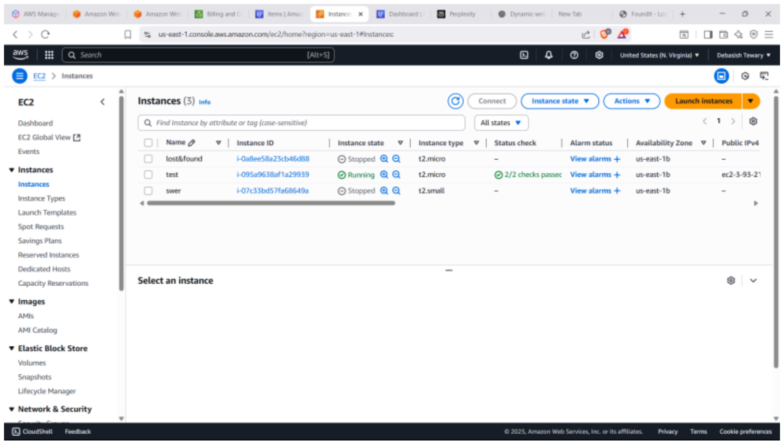
Amazon SNS	Send notification emails/SMS
AWS Lex (Optional)	Chatbot integration

4.4 Detailed Service Descriptions and Connections

Amazon EC2

- **Description:** EC2 instances serve as the virtual servers for our application. We will use them to host both the user-facing frontend (React/HTML) and the backend API that handles all business logic, database interactions, and communication with other AWS services.
- **Connections:**
  - **EC2 to DynamoDB:** The backend API on the EC2 instance will make API calls to DynamoDB to store and retrieve user data, item reports, and feedback.
  - **EC2 to S3:** The backend API will interact with S3 to handle image uploads from users and to retrieve images for display and analysis.
  - **EC2 to Rekognition:** When an admin initiates a match request, the backend on EC2 will call the Rekognition API, passing the S3 locations of the images to be compared.
  - **EC2 to SNS:** Upon a successful match detected by the backend logic, an API call will be made from EC2 to SNS to send a notification.
  - **EC2 instance is connected to DynamoDB , S3, SNS using python flask**

[EC2 instance image here]


us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#Instances:

Instances (3)

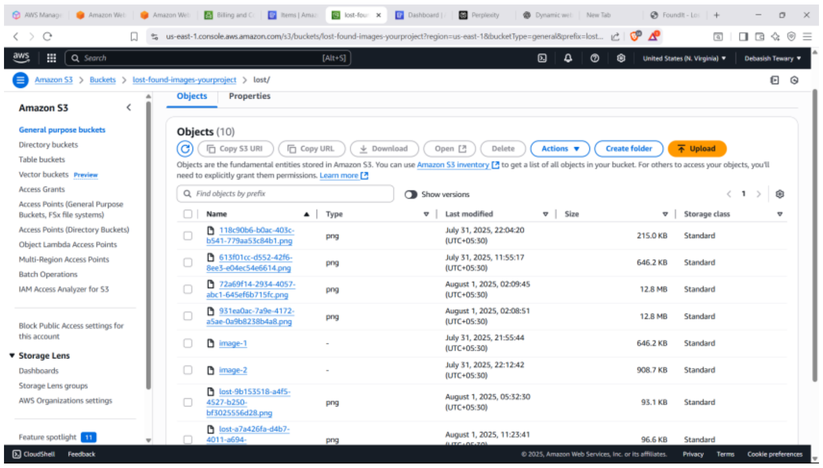
Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4
lost&found	i-0a8ee58a23cb46d88	Stopped	t2.micro	–	View alarms +	us-east-1b	–
test	i-095a9638af1a29939	Running	t2.micro	2/2 checks passec	View alarms +	us-east-1b	ec2-3-93-21
swer	i-07c33bd57fa68649a	Stopped	t2.small	–	View alarms +	us-east-1b	–

Select an instance

Amazon S3 (Simple Storage Service)

- **Description:** S3 is used for scalable and secure object storage. We will create two dedicated S3 buckets: foundit-lost-items-images and foundit-found-items-images. These buckets will store the images uploaded by users for their lost and found reports.
- **Connections:**
  - **S3 to EC2:** The EC2 backend will handle the direct upload of images to S3 and will also fetch image URLs for display on the frontend or for analysis by Rekognition.
  - **S3 to Rekognition:** The Rekognition service will be granted permissions to access the images stored in the S3 buckets to perform facial/object comparison.

[S3 bucket and access control screenshot here]



us-east-1.console.aws.amazon.com/s3/buckets/lost-found-images-yourproject?region=us-east-1&bucketType=general&prefix=lost...

Amazon S3 > Buckets > lost-found-images-yourproject > lost/

Objects (10)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Name	Type	Last modified	Size	Storage class
118c90b6-b0ac-403c-b541-779aa53c84b1.png	png	July 31, 2025, 22:04:20 (UTC+05:30)	215.0 KB	Standard
613f01cc-d552-42f6-8ee3-e04ec54e6614.png	png	July 31, 2025, 11:55:17 (UTC+05:30)	646.2 KB	Standard
72a69f14-2934-4057-abc1-645ef6b715fc.png	png	August 1, 2025, 02:09:45 (UTC+05:30)	12.8 MB	Standard
931ea0ac-7a9e-4172-a5ae-0a9b8238b4a8.png	png	August 1, 2025, 02:08:51 (UTC+05:30)	12.8 MB	Standard
image-1	-	July 31, 2025, 21:55:44 (UTC+05:30)	646.2 KB	Standard
image-2	-	July 31, 2025, 22:12:42 (UTC+05:30)	908.7 KB	Standard
lost-9b153518-a4f5-4527-b250-bf3025556d28.png	png	August 1, 2025, 05:32:30 (UTC+05:30)	93.1 KB	Standard
lost-a7a426fa-d4b7-4011-a694-	png	August 1, 2025, 11:23:41	96.6 KB	Standard

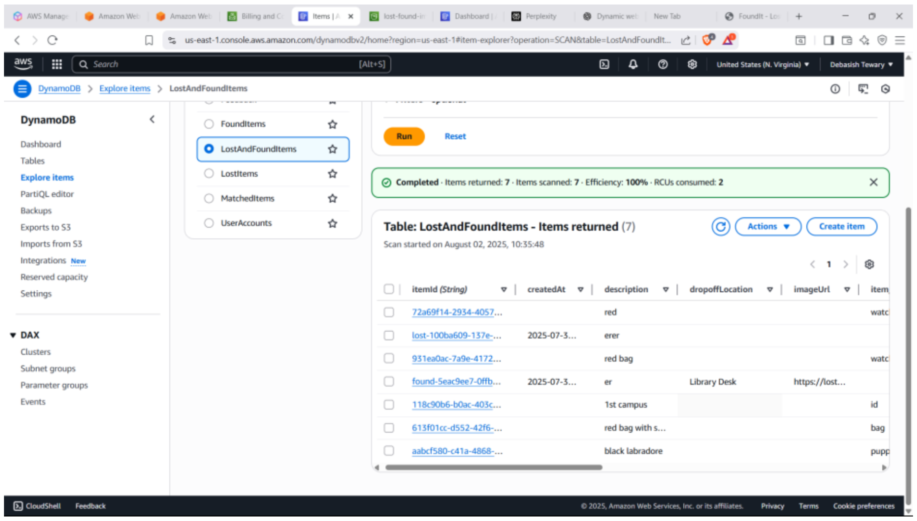
© 2025, Amazon Web Services, Inc. or its affiliates.

Amazon DynamoDB

- **Description:** DynamoDB is a fast, flexible NoSQL database service. We will use it to store all application data, structured into several tables:
  - Users: Stores user registration data (name, email, password hash, roll number) and admin credentials.
  - LostItems: Stores metadata for lost item reports (item description, user ID, image S3 URL, report date, status).
  - FoundItems: Stores metadata for found item reports, similar to LostItems.
  - Feedback: Stores user feedback or chatbot conversation logs.
- **Connections:**

- **DynamoDB to EC2:** The backend API on EC2 will be the primary point of access for DynamoDB, performing all read, write, update, and delete operations.

[DynamoDB table schema]



us-east-1.console.aws.amazon.com/dynamodbv2/home?region=us-east-1#item-explorer?operation=SCAN&table=LostAndFoundIt...

DynamoDB > Explore items > LostAndFoundItems

Completed · Items returned: 7 · Items scanned: 7 · Efficiency: 100% · RCUs consumed: 2

Table: LostAndFoundItems - Items returned (7)

Scan started on August 02, 2025, 10:35:48

itemId (String)	createdAt	description	dropoffLocation	imageUrl	item
72a69f14-2934-4057...		red			watc
lost-100ba609-137e-...	2025-07-3...	erer			
931ea0ac-7a9e-4172...		red bag			watc
found-5eac9ee7-0ffb...	2025-07-3...	er	Library Desk	https://lost...	
118c90b6-b0ac-403c...		1st campus			id
613f01cc-d552-42f6-...		red bag with s...			bag
aabcf580-c41a-4868-...		black labradore			pupp

Amazon Rekognition

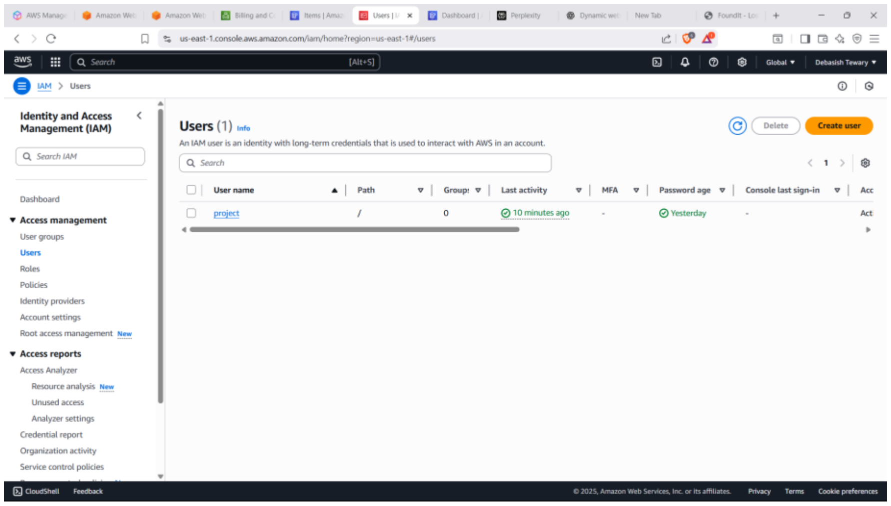
- **Description:** Rekognition is a powerful image and video analysis service. In this application, it will be used specifically for image comparison. The CompareFaces or CompareObjects API will be called to compare the image of a lost item with the image of a found item to determine if there is a match.
- **Connections:**
  - **Rekognition to S3:** Rekognition requires permissions to read images from the S3 buckets to perform its analysis. The EC2 backend will provide the S3 object keys as input to the Rekognition API call.
  - **Rekognition to EC2:** The EC2 backend will receive the comparison results, including a confidence score, from Rekognition. The backend will then use this score to determine if a match is successful.

Amazon SNS (Simple Notification Service)

- **Description:** SNS is a highly available, scalable, and durable messaging service. We will use it to send notifications to users. Upon a successful item match, a message will be published to an SNS topic, which can then be delivered to the relevant user via email or SMS.
- **Connections:**
  - **SNS to EC2:** The EC2 backend will publish a message to an SNS topic after a successful match is confirmed in the database.

- **SNS to Users:** SNS will handle the delivery of the notification message to the user's registered email or phone number.

[Amazon Rekognition, Amazon SNS and S3 are connected with iam user]



Users (1) Info

An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.

User name	Path	Groups	Last activity	MFA	Password age	Console last sign-in	Acc
project	/	0	10 minutes ago	-	Yesterday	-	Act

5. Non-Functional Requirements

5.1 Performance

- The system shall provide a responsive user experience with minimal latency for page loads and data submissions.
- Image upload and retrieval times should be optimized for quick access.
- Rekognition image comparison should complete within a reasonable timeframe (e.g., a few seconds).

5.2 Security

- User credentials stored in DynamoDB shall be hashed and salted.
- All communication between the client and server shall be encrypted (HTTPS).
- Access to S3 buckets and DynamoDB tables shall be secured using appropriate IAM roles and policies.
- Admin access shall be restricted and require robust authentication.

5.3 Scalability

- The AWS architecture (EC2, S3, DynamoDB) provides inherent scalability to handle an increasing number of users and data.

- The system should be designed to scale horizontally to accommodate potential growth in user base and item reports.

5.4 Usability

- The user interface (UI) for both the user and admin portals shall be intuitive and easy to navigate.
- Clear instructions shall be provided for reporting lost and found items.
- Error messages shall be clear and helpful.

5.5 Reliability

- The system shall be available 24/7 with minimal downtime.
- Data integrity shall be maintained for all stored information in DynamoDB and S3.
- Backup and recovery mechanisms for data should be in place.

6. Future Enhancements (Optional)

- Integration of AWS Lex for a more sophisticated chatbot.
- Automated matching for newly reported items using Rekognition without explicit admin initiation.
- Advanced search and filtering options for lost and found items.
- Push notifications in addition to email/SMS for matched items.