Project Proposal: Finder's Keeper - Neighborhood Lost & Found

Team: A

Presenters: Mohammad noor abu asbe 315226472

Roaa Khashab 213262801

Anas farhat 212607345

Executive Summary

Finder's Keeper is a public utility platform designed to quickly connect community members regarding lost or found items such as keys, phones, or pets. By providing a secure, centralized digital bulletin board where interactions are protected by identity management, the platform reduces the friction, delays, and inefficiency associated with traditional lost and found methods. The platform utilizes a straightforward, event-driven, Serverless architecture relying on managed services like Amazon Cognito to maximize security for interacting parties, enhance community trust, and offer significant value to everyday users.

Problem

Community members often struggle to find effective ways to report or claim lost property quickly, relying on outdated or decentralised methods that lead to delays and reduced recovery rates. Furthermore, relying on public forms requires exposing sensitive contact information, risking privacy. Traditional processes lack a secure, central digital repository with managed user access, prolonging the separation of items from their owners and potentially eroding local trust.

Solution

Finder's Keeper solves this by deploying a simple, highly available, and scalable Serverless application that acts as a community-wide digital lost and found. This solution leverages core AWS building blocks such as Lambda, DynamoDB, and API Gateway to handle secure data persistence and retrieval efficiently. Critically, Amazon Cognito is implemented to handle user authentication, ensuring that secure actions such as contacting a poster are performed by verified users. By focusing on managed identity, CRUD operations, and reliable notification handling, the project demonstrates proficiency in fundamental cloud development principles, including secure access control.

Key Features

The core features align directly with secure community needs and demonstrate proficiency in mapping use cases to practical application features:

Use Case / Feature	Description	Value to User
User Registration & Login	Users can securely register and authenticate using Amazon Cognito, which manages user identity and access to secure endpoints.	Provides trust and accountability for interactions on the platform.
Post Lost/Found Item	Authenticated users upload essential details (description, contact info) and an image of the item through a protected web form.	Provides a quick and straightforward way to register an item for retrieval or claim.
Browse Recent Posts	Displays a dynamically updating list of the last 20–30 lost or found items posted directly on the homepage.	Offers immediate visibility into recently added items without heavy query processing.
Secure Search/Filter	Allows searching based on basic criteria such as item type (e.g., "dog," "keys," "wallet") or status (lost/found).	Efficiently narrows down the visible list to highly relevant items.
Secure Contact Notification	An authenticated user can initiate a secure backend workflow to send a message to the original poster. Access control to this sensitive function is managed by Amazon Cognito.	Enables a private, secure connection between the two parties without publicly exposing direct contact information.

Solution and Tools/Technologies

This implementation utilises a Serverless architecture relying on fundamental cloud development services.

Component	Functionality and AWS Service
Frontend	A Static single-page application (HTML, CSS, JavaScript) hosted securely on Amazon S3.
Identity Management	Amazon Cognito (User Pools) provides secure registration and authentication for users needing access to protected features, offering centralized identity management.
Backend Interface	AWS API Gateway (REST API) serves as the secure interface, protecting endpoints using Cognito authorization where required, adhering to core API development patterns.
Compute / Logic	All application logic is handled by scalable, serverless AWS Lambda functions (Python/Boto3), executing authenticated requests, data retrieval logic, and coordinating contact notifications.
Database	Amazon DynamoDB (NoSQL database) provides durable and scalable storage for item records.
Media Storage	Amazon S3 is used for secure, highly durable storage of images associated with each posted item.
Communication	A serverless workflow using Lambda and Amazon Simple Email Service (SES) is utilized to send secure contact notifications.

The proposed architectural pattern is highly suitable for quick deployment (POC) and demonstrates mastery of key objectives, including secure identity management (Cognito),

configured serverless interfaces (API Gateway), handling persistent storage (S3, DynamoDB), and implementing event-driven serverless logic (Lambda).