Web Services - 2024-2025

LOST AND FOUND SYSTEM FOR TUNIS-CARTHAGE AIRPORT

ENHANCING PASSENGER EXPERIENCE AND OPERATIONAL EFFICIENCY

Presented by:Farah Jaouadi

Professor: Montassar Ben Messaoud

Agenda

Introduction

- Tunis-Carthage Airport faces inefficiencies in Lost and Found management.
- Current manual process is time-consuming and inconvenient.
- Need for a digital solution to improve service quality.

Motivation

Long Wait Times

- Long queues and multiple visits frustrate travelers.
- No transparency or real-time updates on lost items.

Operational Challenges

- Inefficient system impacts airport operations.
- Manual processes lead to inefficiencies and errors.

Goals

Improve passenger convenience and align services with international standards.

Problem

Challenges in the Current System

The existing manual Lost and Found process at Tunis-Carthage Airport is inconvenient, time-consuming, and lacks transparency, leading to passenger dissatisfaction and operational bottlenecks that hinder efficiency and customer service.

Objectives

Digitization

Transition the airport's outdated manual processes to a fully digital platform that simplifies reporting, tracking, and claiming lost items, ensuring seamless interactions for both passengers and staff.

User Experience

Design a secure and intuitive interface—whether web or mobile—that prioritizes ease of use while maintaining robust data protection for passengers.

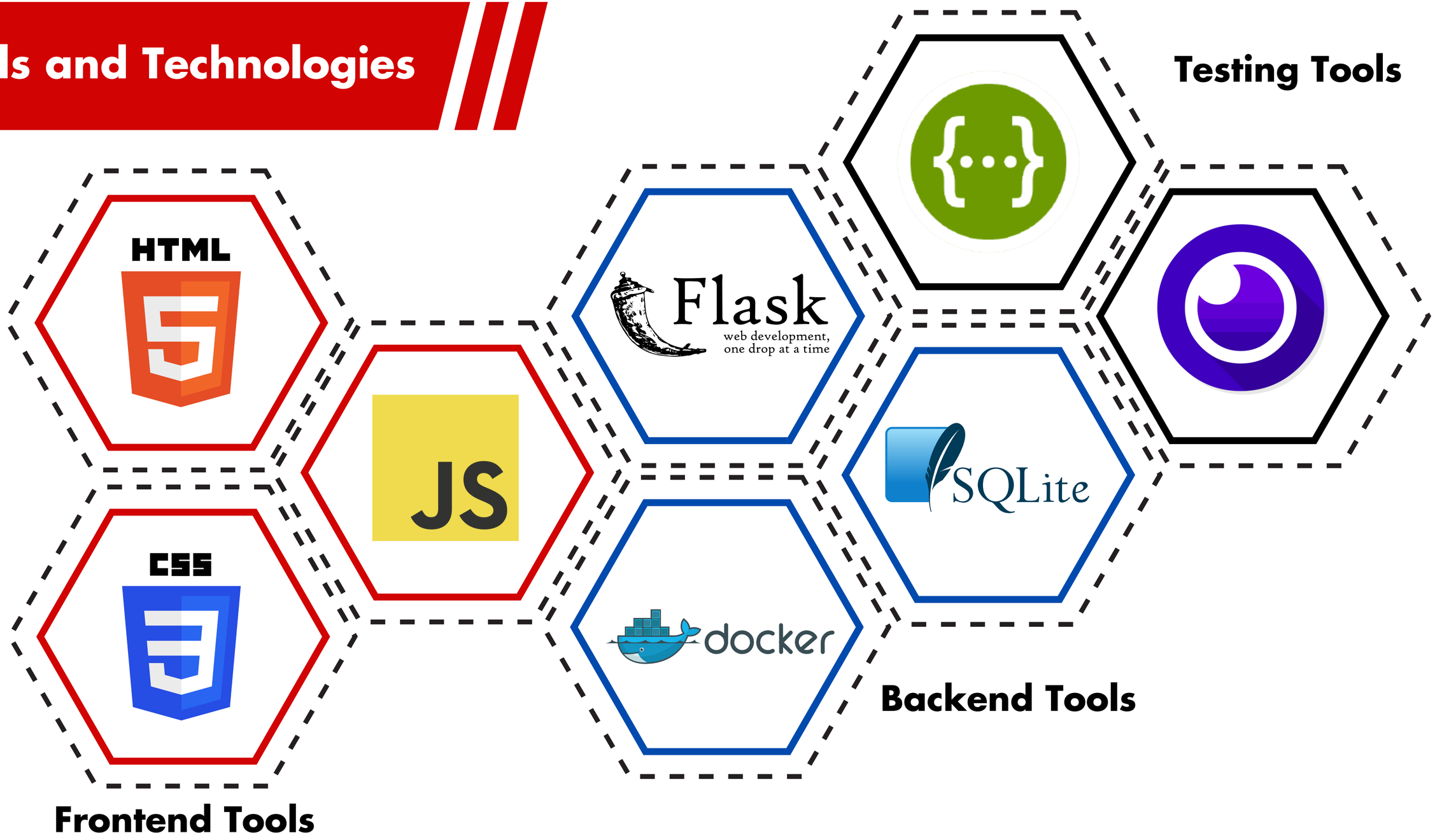
Automation

Streamline operations by enabling passengers to self-report, search, and claim lost items online, reducing dependence on physical visits and manual paperwork.

Transparency

Enhance service quality by providing real-time updates on the status of lost items, offering clear and accessible communication channels for passengers.

Tools and Technologies



Database Integration

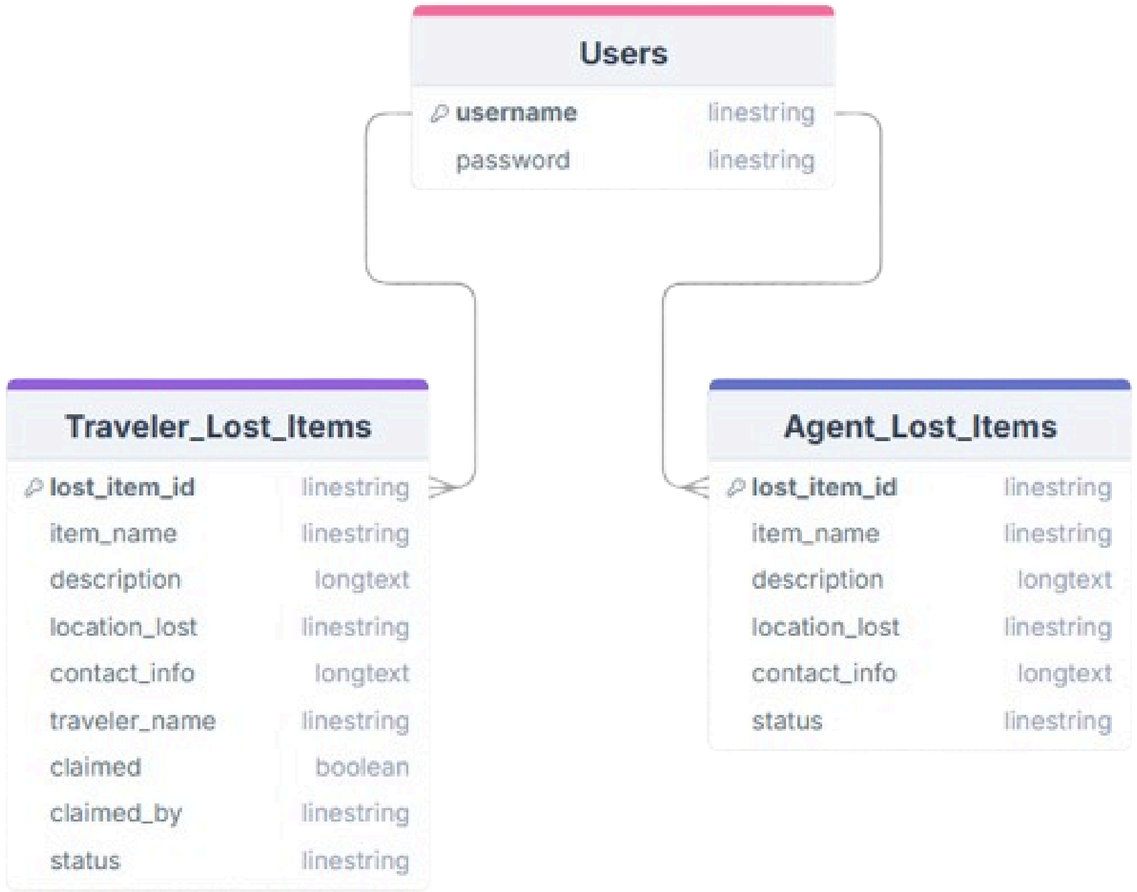


Figure 1: Database Schema of the Lost and Found Application

Database Integration with SQLite

SQLite is a lightweight, file-based database, ideal for small-scale applications like Lost and Found management. It integrates easily with Flask and eliminates the need for a separate database server.

SQLAlchemy was used as an Object-Relational Mapping (ORM) for simpler database interactions.

Key Tables:

- *traveler_lost_items*: Stores items reported by travelers.
- *agent_lost_items*: Stores items reported by agents.
- *users*: Stores user details for authentication and interaction.

API Endpoints and Testing

Agent Endpoints:

- **Agent Register:** POST /api/register
- **Agent Login:** POST /api/login
- **Add Lost Item:** POST /api/lost-items/report-by-agent
- **Update Item Status:** PUT /api/lost-items/{item_id}
- **Delete Lost Item:** DELETE /api/lost-items/{item_id}

Traveler Endpoints:

- **Traveler Register:** POST /api/register
- **Traveler Login:** POST /api/login
- **Add Lost Item:** POST /api/lost-items/report-by-traveler
- **Claim Lost Item:** POST /api/lost-items/claim/{item_id}

Common Endpoints:

- **Search Lost Items:** GET /search?description={description}
- **Get All Lost Items:** GET /api/lost-items

API Testing

Tested using Insomnia for:

- Endpoint functionality
- Data validation
- Response accuracy

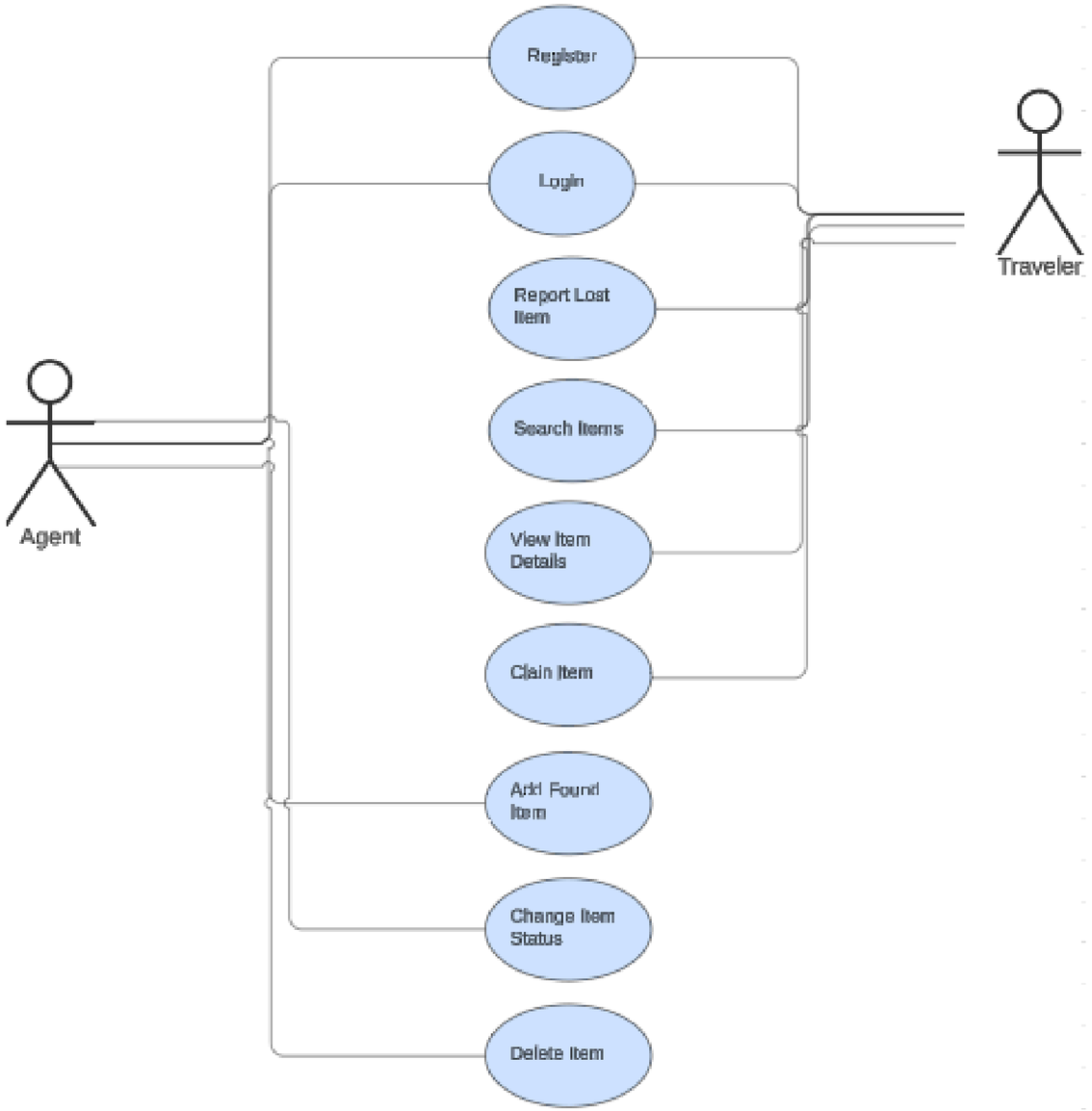
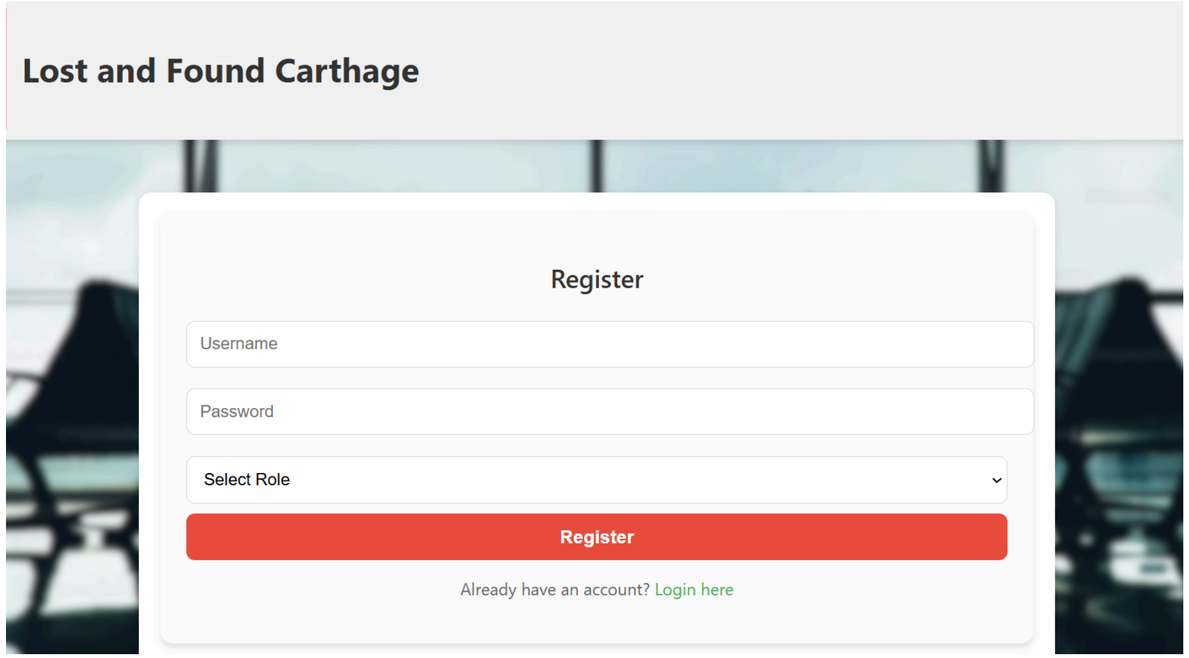


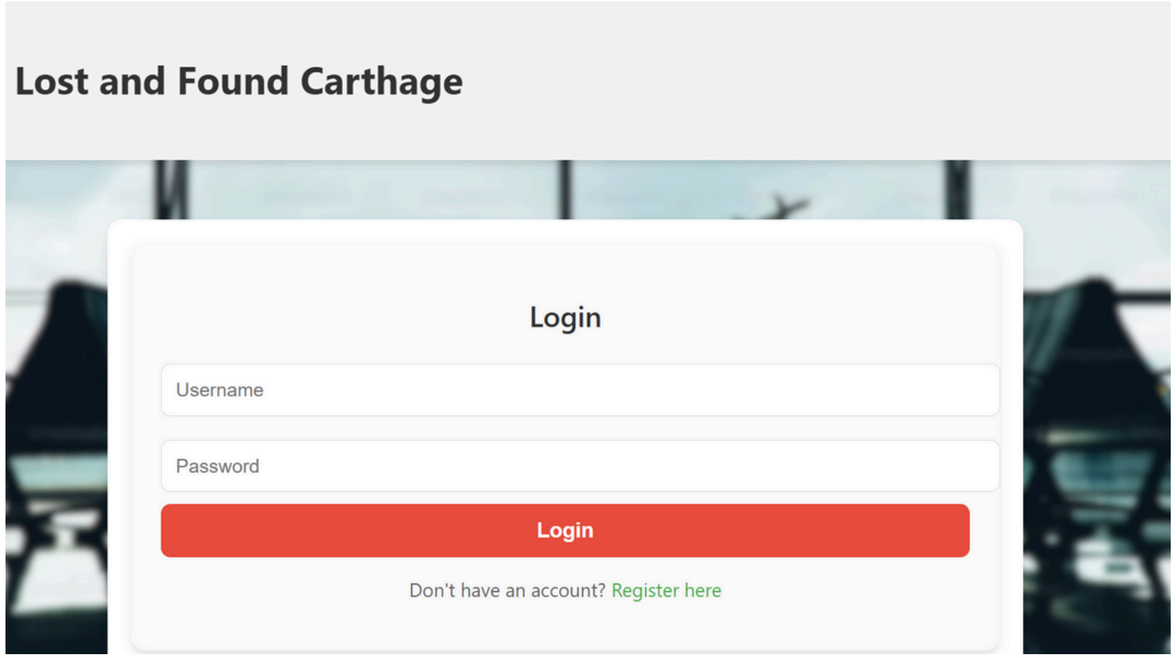
Figure 2: Use Case Diagram

Frontend Integration

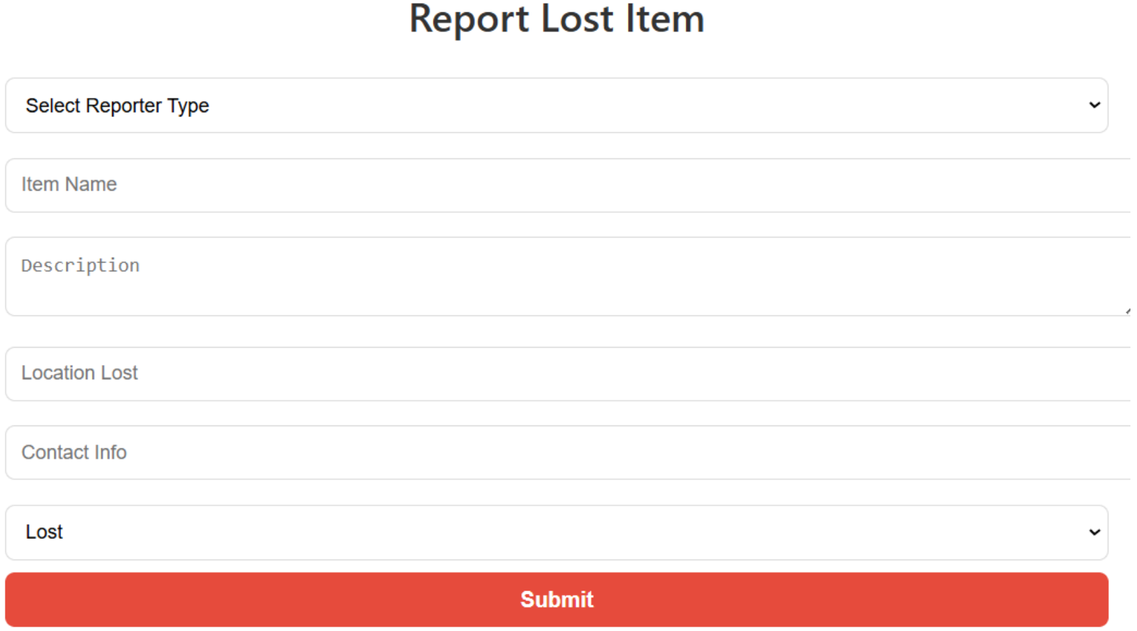
Technology: Built with **HTML, CSS, and JavaScript** for a responsive, user-friendly interface.
Features:



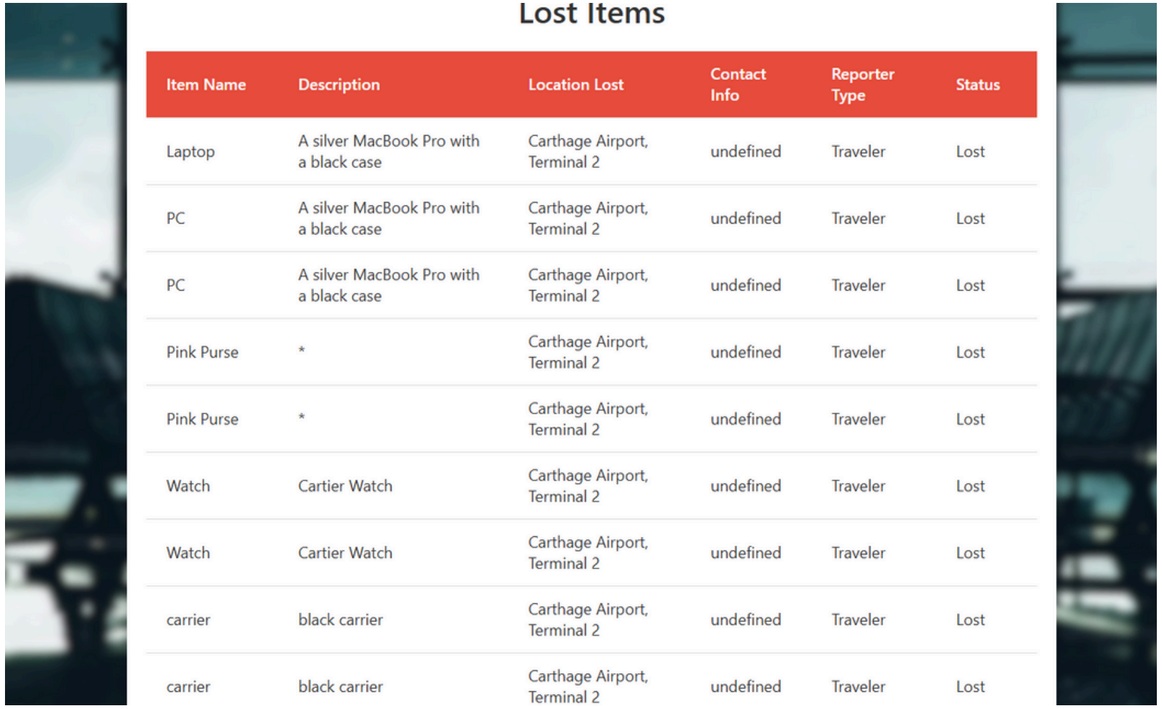
Register: Users can create an account to manage their lost items.



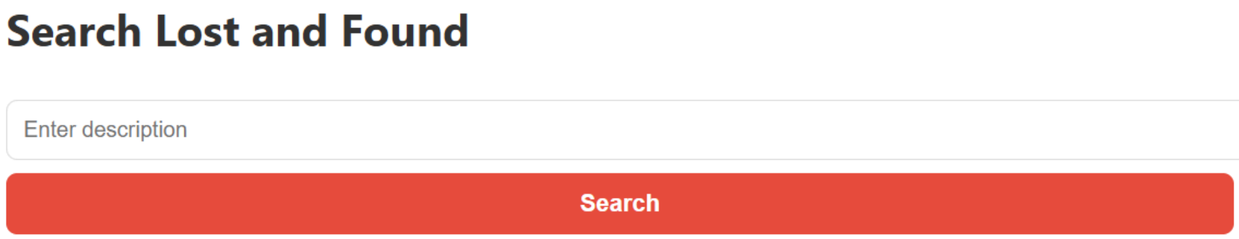
Login: Secure login for users to access their website.



Report Lost Items: Forms for both agents and travelers to submit lost items.



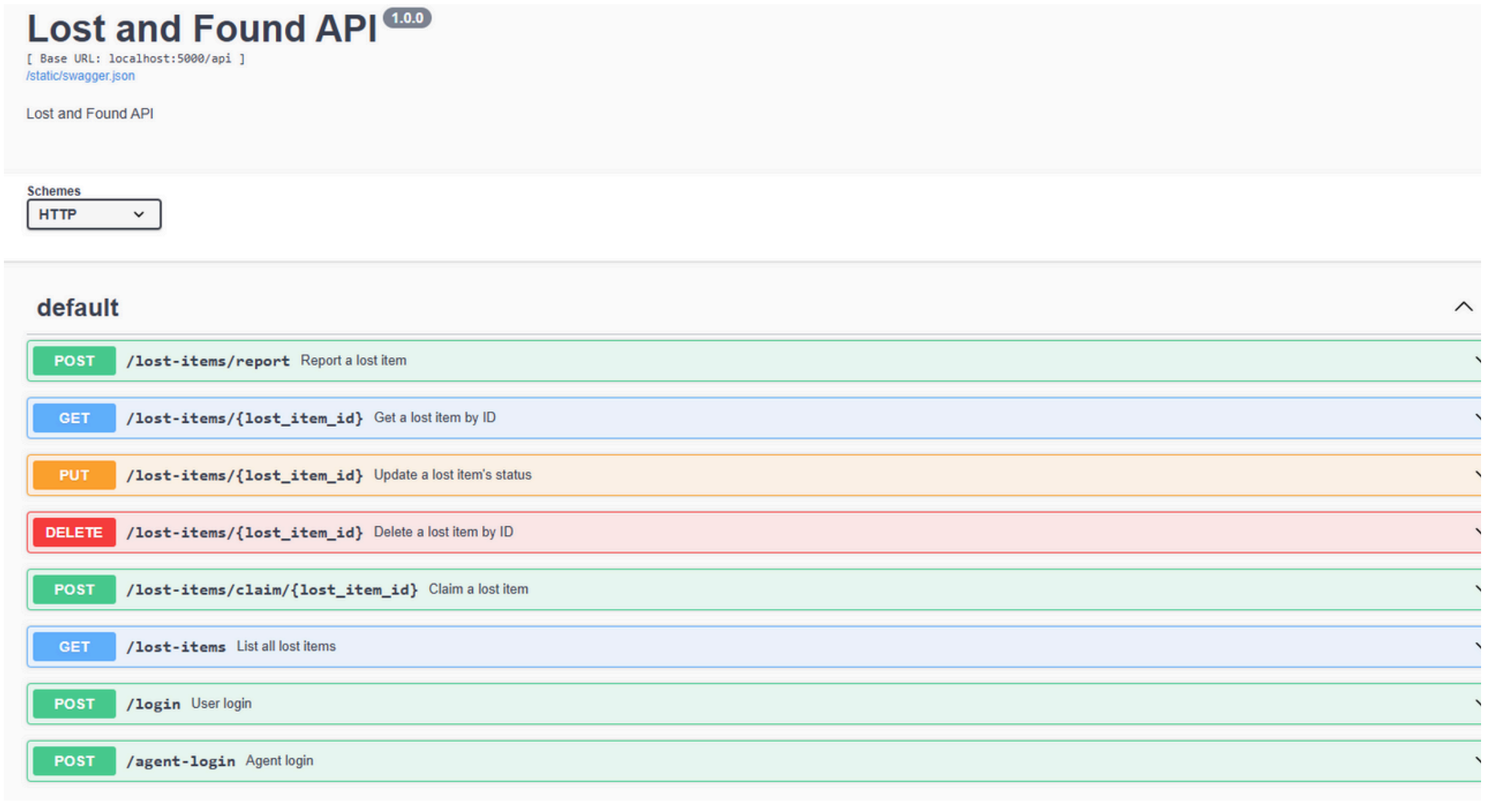
List of All Lost Items: A comprehensive list of all reported lost items, displayed with key details like item name, description, and status.



Search Items: Search bar for querying lost items by description.

Swagger Documentation

- **Purpose:** API endpoints documented for clarity and testing.
- **Features:** Interactive testing, detailed endpoint info, and simplified backend integration.



JWT Authentication

- **Purpose**: Secure user authentication with JSON Web Tokens.
- **Features:** Access tokens for session management, refresh tokens for seamless login, and tamper-proof security.

Future Enhancement

Enhanced Frontend

- Improve visual design and user experience.
- Add advanced filtering options for easier searches.

Image Uploads

- Allow users to attach images of lost items.
- Enhance item identification and tracking.

Integration with External API

- Link lost item reports to specific flights.
- Use a flight arrivals API to streamline the reporting process.
- Simplify tracking for travelers and agents.

Conclusion

The Lost and Found API digitizes Tunis-Carthage Airport’s manual lost item process, improving efficiency and passenger satisfaction. Built with Flask, SQLite, and SQLAlchemy, the system offers a user-friendly interface and secure data management. Its scalable design holds the potential to modernize lost and found operations at the Tunis airport.

THANK **YOU**

Any Questions?