

Florida International University
Knight Foundation School of Computing and Informational Sciences

Computer Science Focus

Final Deliverable

Project Title: Campus Lost and Found

Team Members: Dahiver Guerra, Jonathan Hori, Cristian Gamboa, Randy Arbolaez, Christian Stewart

Product Owner: Nicholas Leal

Instructor: Masoud Sadjadi

Abstract

This Final Deliverable document provides a comprehensive overview of the development process for the Campus Lost and Found application which was designed to aid students and staff search for, report and claim items in the Campus' Lost and Found. The current system doesn't solve the issue of easily being able to access the lost and found system on campus. This application will make the process user friendly and interactive. The development process entailed implementing the scrum framework and using user stories which utilized React for the frontend, Spring Boot for the backend and Kotlin for the mobile application. Following methodologies, the project included sprint plans and review sessions that ensured continuous enhancements and alignment with user requirements. This document will also include our sprint review reports which highlight the collaborative nature of the project itself.

Table of Contents

Table of Contents.....	3
Introduction.....	4
Current System.....	4
New System.....	4
User Stories.....	5
Implemented User Stories.....	5
Pending User Stories.....	6
Project Plan.....	7
Hardware and Software Resources.....	7
Sprint Planning Meeting.....	8
System Design.....	10
Architectural Patterns.....	10
System and Subsystem Decomposition.....	11
Deployment Diagram.....	12
Design Patterns.....	12
Glossary.....	13
APPENDIX.....	15
Appendix A - Sprint Review Reports.....	15
Appendix B - Shortcomings.....	21
Appendix C - Feature Wishlist.....	22

Introduction

Florida International University is a large and dynamic campus, home to over 56,000 students. With so many people coming and going each day, it's no surprise that personal belongings are often misplaced. Unfortunately, the current lost and found system does little to ease the stress of losing something valuable. It's outdated, hard to navigate, and leaves many students and staff feeling uncertain about whether their items will ever be found. As the campus continues to grow, so does the need for a more reliable and accessible way to manage lost and found items, one that reflects the pace and scale of university life today. Our Campus Lost and Found is a modern, user-friendly platform designed to streamline the reporting and retrieval process.

Current System

While the current lost and found system at Florida International University technically fulfills its basic purpose, it leaves much to be desired in terms of visibility and user engagement. At its core, the system operates as a simple submission form. Users can report lost items, but there is little feedback or follow-up beyond that initial step. Items that have been submitted are not displayed publicly, meaning students and staff have no way of browsing what's been found or checking if their lost belongings have been turned in. This lack of transparency and interactivity creates a disconnect between users and the system. Many students aren't even aware the system exists, and those who are often find it unhelpful or outdated.

New System

The purpose of our new Campus Lost and Found system is to create a more transparent, accessible, and user-friendly platform for reporting and recovering lost items at Florida International University. Unlike the current setup, our system will allow users to browse a real time list of submitted lost and found items, complete with images, descriptions, and locations. By streamlining communication between students, staff, and campus facilities, the goal is to reduce the stress and uncertainty that often comes with losing personal belongings. Ultimately, this project aims to modernize the lost and found experience and make it easier for the FIU community to reconnect with what they've lost.

User Stories

This section outlines the user stories implemented across the five springs of the “Campus Lost and Found” project. Each story represents a key feature or improvement, from user registration and item reporting to admin approvals and role based access. These stories reflect the step by step development of the system and how the team delivered core functionality over time.

Implemented User Stories

- #001: As a developer, I want to create and connect a database to store item details, images, user accounts, and locations, so that all lost and found items are persistently saved and organized.
- #002: As a developer, I want to design a user-friendly and responsive interface for both desktop and mobile users so that users can easily navigate and interact with the platform.
- #003: As a user, I want to submit a lost or found item with a description, photo, and location so others can identify it easily
- #004: As a user, I want to browse the submitted items with images and a description so that I can identify any that have been lost or found
- #005: As a user, I want to search for lost or found items by keyword, category, or location so I can quickly narrow down the results
- #006: As an admin, I want to view and manage user posts so I can remove inappropriate or false listings to keep the platform safe.
- #007: As a developer, I would like to set up a github for the project so that I can work on it with the team.
- #008: As a developer, I would like to research other lost and found apps so that I can have a good idea of what we have to work on.
- #009: As a user, I want to upload images for the posts so that they can be a lot more informative
- #010: As a user, I want to be able to claim the item so that listings remain up-to-date and accurate
- #011: As an admin, I want to be able to delete items that are posted.
- #012: As an admin, I want to be able to refresh the list of items.
- #013: As a user, I want the platform to be mobile-friendly so that I can use it easily on my phone while on campus
- #014: As a dev, I want to be able to share the project in its entirety and have it easily accessible so that others can clone and use it.
- #015: As a user, I would like to know how the lost and found functions as I’m using it so that I can use it efficiently.

Pending User Stories

#016: As a user, I want to be notified whenever important changes are made to my listing

#017: As a system, I want to detect and suggest merging duplicate item listings to keep the platform clean

#018: As a user, I want to report any suspicious or false listings so that moderators can review and maintain the quality of the platform

Project Plan

This section outlines the planning process that guided the development of the Campus Lost and Found project. Following agile methodology, the team organized the work into structured sprints, each with specific goals and deliverables. Details of the sprint planning, task distribution, and workflow management are included. Additionally, this section highlights the key software tools, frameworks, and hardware selected to support both development and deployment of the application.

Hardware and Software Resources

Listed below is a list of all hardware and software resources that were used in this project:

- *GitHub* - Version control and collaboration platform
- *Java* - Backend programming language
- *React + Vite* - Frontend framework and build tool
- *Spring Boot* - Java framework for building the backend API
- *MySQL* - Relational database for data storage
- *JWT (JSON Web Tokens)* - Used for secure, stateless authentication
- *Spring Security* - Role-based access control and security config.
- *Axios* - HTTP client for frontend-backend communication
- *Gradle* - Build automation tool for backend
- *Postman* - API testing during development
- *React Router* - Client-side routing for web applications
- *CSS* - Styling for frontend
- *Android Studio* - mobile components
- *Localhost Servers* - running backend and frontend locally during development

Sprint Planning Meeting

Sprint 1: Project Setup and Initial Research

- Objective: Setup project environment and research other lost and found sites.
- Tasks:
 - Research React and other tech stacks
 - Setup GitHub repository
 - Install necessary dependencies
 - Research lost and found apps
- Deliverables:
 - Project repository with initial setup
 - Research documentation for lost and found apps
 - Documentation on tech stacks

Sprint 2: Design frontend and Setup Backend

- Objective: Setting up the frontend and creating the backend.
- Tasks:
 - Create a backend database to store lost and found items
 - Develop the frontend design
 - Implement basic features
- Deliverables:
 - Implemented basic frontend and backend with database
 - Added simple features.

Sprint 3: Implement the webapp features for Lost and Found

- Objective: Setup fundamental features for the webapp
- Tasks:
 - Search and Filter Items feature
 - Delete posts and edit
 - Admin panel
 - Backend work
- Deliverables:
 - Created logic for Search and filter feature
 - Allowed for deletion of posts from database through frontend
 - Major changes to the backend

Sprint 4: Implementation of more lost and found features

- Objective: Continue implementing features
- Tasks:
 - Admin permissions on the site
 - Mobile Experience connected with Webapp
 - Item refresh and claim
 - Image uploads
- Deliverables:
 - Fully functioning admin privileges
 - Working mobile app
 - Image url uploads
 - Successful item submissions and claims.

Sprint 5: Testing, Documentation and Finalization

- Objective: Test, document and finalize the project
- Task:
 - Create and edit documentation for the project
 - Test all features
 - Finalize project for final deliverable
- Deliverables:
 - Comprehensive project documentation
 - Project poster
 - Presentation ready project

System Design

This section details the key design decisions made during the development of the Campus Lost and Found project. It outlines the overall system architecture, breaks down the main subsystems, and explains the design patterns used to ensure modularity, security, and ease of maintenance.

Architectural Patterns

The system follows a modular client server architecture, separating frontend and backend responsibilities to improve scalability and maintainability. Key patterns include:

- **Client-Server Architecture:**
The client is built using React + Vite, offering a responsive and dynamic user interface. The backend is powered by Spring Boot, exposing RESTful endpoints for secure data access and processing.
- **Modular Design:**
Each layer (frontend, backend, database) is logically separated. Backend responsibilities are divided into controllers, services, and repositories, while the frontend maintains separation of concerns between views, components, and routing.
- **RESTful API:**
The application uses RESTful principles to communicate between the client and server, allowing for clean, scalable, and stateless interactions.
- **JWT-Based Authentication:**
Security and user roles are enforced using JWT tokens and Spring Security, ensuring authenticated access to protected routes and role-based privileges.

System and Subsystem Decomposition

The system is composed of the following core subsystems:

- **User Management:**
Handles user registration, login, and role assignment (e.g., user or admin). JWT tokens are generated upon login for secure access.
- **Item Management:**
Allows users to report lost and found items, including details like title, description, image, and location. Admins can approve or delete items.
- **Admin Dashboard:**
A restricted-access interface where administrators can view, approve, or remove submitted items. This dashboard supports backend moderation of public listings.
- **Frontend UI:**
Built using React, the UI includes views for browsing items, submitting reports, and admin interactions. It communicates with the backend via Axios and handles routing via React Router.
- **Database Layer:**
MySQL stores all user accounts, items, and roles. The backend interacts with the database using Spring Data JPA.

Deployment Diagram

The system is deployed using a traditional client-server setup:

- **Client-Side:**
Runs in the user's web browser. Built using React + Vite, it communicates with the backend via HTTP requests using Axios.
- **Server-Side**
A Spring Boot application hosted locally. It handles authentication, data validation, API routing, and database communication.
- **Database:**
A MySQL database hosted locally, managing persistent data for users and items.

Design Patterns

The following design patterns were employed to support clean architecture, modularity, and maintainability throughout the project:

- **Model-View-Controller (MVC):**
Used on the backend with Spring Boot to separate concerns, Models represent data (e.g., users, lost items), Controllers handle HTTP requests, and Services act as intermediaries for business logic and data persistence.
- **Repository Pattern:**
Utilized in the backend to abstract and encapsulate database access. Each entity (e.g., User, LostItem) has a corresponding repository interface that handles CRUD operations using Spring Data JPA.
- **Singleton Pattern:**
Applied to components such as the JWT token provider and security configuration to ensure a single, shared instance throughout the application lifecycle.
- **Component Based Architecture (Frontend):**
The React frontend uses reusable and stateful components, improving scalability and reducing code duplication. Components such as item cards, forms, and dashboard views follow consistent structure.
- **Client Side Routing:**
Implemented with React Router, allowing dynamic rendering of views based on URL paths without reloading the page.

Glossary

- **User Interface (UI):**
The part of the application that users interact with visually. In this project, the UI includes components for submitting lost/found items, browsing listings, and navigating the platform on web and mobile.
- **Client-Server Architecture:**
A model where the frontend (client) communicates with the backend (server) to send and receive data. React + Vite handles the frontend while Spring Boot manages backend processing.
- **React:**
A JavaScript library for building fast, interactive user interfaces. Used to build the frontend of the application with reusable components.
- **Vite:**
A modern build tool that improves development speed and performance for React applications.
- **Spring Boot:**
A Java-based framework used to build and run the backend, including API endpoints, business logic, and database interaction.
- **MySQL:**
A relational database used to store users, lost and found items, and -related metadata in a structured format.
- **JWT (JSON Web Token):**
A secure token used for authenticating users in a stateless manner. JWTs allow the frontend to access protected resources after login.
- **Spring Security:**
A module in Spring Boot used for securing endpoints, managing authentication, and enforcing role-based access control (e.g., user vs. admin).
- **Repository Pattern:**
A design pattern that separates data access logic from business logic by using repository interfaces to manage database operations.
- **Model-View-Controller (MVC):**
An architectural pattern that separates application logic into models (data), views (UI), and controllers (logic to handle requests and responses).
- **Component-Based Architecture:**
A frontend design approach using modular, reusable React components for building UI features such as forms, item cards, and dashboards.

- **React Router:**
A library for managing navigation and routing in single-page applications, allowing users to move between views without full page reloads.
- **Axios:**
A promise-based HTTP client used in the frontend to send API requests to the backend and handle responses.
- **Postman:**
A tool used to test backend API endpoints during development by sending various request types (GET, POST, PUT, DELETE).
- **Gradle:**
A build automation tool used for compiling, testing, and packaging the Spring Boot backend.
- **Android Studio:**
An integrated development environment (IDE) used for building the mobile version of the Campus Lost and Found app using Kotlin.
- **Localhost:**
A development environment where the application is run locally on a developer's machine before being deployed or tested on the cloud.
- **Sprint:**
A fixed period in agile development (usually 1–2 weeks) during which a specific set of tasks or features is planned, developed, and reviewed.
- **User Story:**
A short, simple description of a feature from the perspective of the end-user, used to guide agile development tasks.
- **Admin Dashboard:**
A private interface used by staff or moderators to approve, delete, or manage reported items and maintain the integrity of the platform.

APPENDIX

Appendix A - Sprint Review Reports

Sprint 1: Review Meeting Minutes

Attendees: Christian Stewart, Jonathan Hori, Cristian Pena, Randy Arbolaez, Dahiver Guerra, Nicholas Leal

Start time: 1:00pm

End time: 2:00pm

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

- User Story 1: Setup
 - As a developer, I would like to set up a github for the project so that I can work on it with the team.
- User Story 2: Research Lost and Found apps
 - As a developer, I would like to research other lost and found apps so that I can have a good idea of what we have to work on.
- User Story 3: Research Algorithms
 - As a developer, I would like to find different algorithms and their uses in storing and processing information regarding the lost items.
- User Story 4: Research Stack Peripherals
 - As a developer, I would like to research and choose which database & backend services we will use.
- User Story 5: Research UI
 - As a developer, I would like to research the user interfaces that similar apps use to ensure that the UI is consistent with the industry standard and it is easy to understand

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting.

- At the moment, none were rejected.

Sprint 2: Review Meeting Minutes

Attendees: Christian Stewart, Jonathan Hori, Cristian Pena, Randy Arbolaez, Dahiver Guerra, Nicholas Leal

Start time: 12:00pm

End time: 1:00pm

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

- User Story 1: Backend Database
 - As a developer, I want to create and connect a database to store item details, images, user accounts, and locations, so that all lost and found items are persistently saved and organized.
- User Story 2: Frontend Design
 - As a developer, I want to design a user-friendly and responsive interface for both desktop and mobile users so that users can easily navigate and interact with the platform.
- User Story 3: Lost/Found Item Submission
 - As a user, I want to submit a lost or found item with a description, photo, and location so others can identify it easily.
- User Story 4: View Listings
 - As a user, I want to browse the submitted items with images and a description so that I can identify any that have been lost or found.
- User Story 5: Search and Filter Items
 - As a user, I want to search for lost or found items by keyword, category, or location so I can quickly narrow down the results.
- User Story 6: Edit or Delete Posts
 - As a user, I want to be able to edit or delete items I've submitted in case I made a mistake or found the item.
- User Story 8: Admin Panel
 - As an admin, I want to view and manage user posts so I can remove inappropriate or false listings to keep the platform safe.

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting.

- User Story 7: User Accounts
 - As a user, I want to be able to register and log in to my account so that I can post, manage, and track my lost or found items.

- User Story 9: Report Option
 - As a user, I want to report any suspicious or false listings so that moderators can review and maintain the quality of the platform.
- User Story 10: Mobile Experience (Optional Goal)
 - As a user, I want the platform to be mobile friendly so that I can use it easily on my phone while on campus

Sprint 3: Review Meeting Minutes

Attendees: Christian Stewart, Jonathan Hori, Cristian Pena, Randy Arbolaes, Dahiver Guerra, Nicholas Leal

Start time: 12:00pm

End time: 1:00pm

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

- User Story 1: Search and Filter Items
 - As a user, I want to search for lost or found items by keyword, category, or location so I can quickly narrow down the results.
- User Story 2: Edit or Delete Posts
 - As a user, I want to be able to edit or delete items I've submitted in case I made a mistake or found the item.
- User Story 3: Admin Panel
 - As an admin, I want to view and manage user posts so I can remove inappropriate or false listings to keep the platform safe.

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting.

- User Story 7: Mobile Experience (Optional Goal)
 - As a user, I want the platform to be mobile-friendly so that I can use it easily on my phone while on campus.
- User Story 6: Report Option
 - As a user, I want to report any suspicious or false listings so that moderators can review and maintain the quality of the platform.
- User Story 5: User Communication
 - As a user, I want to be notified when my lost item has been found so that I can collect it.

- User Story 4: User Accounts
 - As a user, I want to be able to register and log in to my account so that I can post, manage, and track my lost or found items.

Sprint 4: Review Meeting Minutes

Attendees: Christian Stewart, Jonathan Hori, Cristian Pena, Randy Arbolaes, Dahiver Guerra, Nicholas Leal

Start time: 8:00am

End time: 9:00am

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

- User Story 2: Edit or Delete Posts
 - As a user, I want to be able to edit or delete items I've submitted in case I made a mistake or found the item.
- User Story 5: User Communication
 - As a user, I want to be notified when my lost item has been found so that I can collect it.
- User Story 6: Report Option
 - As a user, I want to report any suspicious or false listings so that moderators can review and maintain the quality of the platform.
- User Story 8: Image Uploads for Posts
 - As a user, I want to upload images for the posts so that they can be a lot more informative.
- User Story 8: Item claimed
 - As a user, I want to be able to claim the item so that listings remain up-to-date and accurate
- User Story 9: Item Delete
 - As an admin, I want to be able to delete items that are posted.
- User Story 10: Items Refresh
 - As an admin, I want to be able to refresh the list of items.

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting.

- User Story 3: Admin Panel
 - As an admin, I want to view and manage user posts so I can remove inappropriate or false listings to keep the platform safe.

- User Story 7: Mobile Experience (Optional Goal)
 - As a user, I want the platform to be mobile-friendly so that I can use it easily on my phone while on campus.
- User Story 4: User Accounts
 - As a user, I want to be able to register and log in to my account so that I can post, manage, and track my lost or found items.
- User Story 1: Web Notifications
 - As a staff member, I want to receive notifications on changes on the items, so that I can be on top of the recent changes.

Sprint 5: Review Meeting Minutes

Attendees: Christian Stewart, Jonathan Hori, Cristian Pena, Randy Arbolaes, Dahiver Guerra, Nicholas Leal

Start time: 8:00am

End time: 9:00am

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

- User Story 2: Admin Account
 - As an Admin, I want to be able to login and edit or delete any posts needed so that I can moderate the posts.
- User Story 4: Mobile Experience
 - As a user, I want the platform to be mobile-friendly so that I can use it easily on my phone while on campus.
- User Story 6: Descriptive GitHub
 - As a dev, I want to be able to share the project in its entirety and have it easily accessible so that others can clone and use it.
- User Story 7: How-to Section
 - As a user, I would like to know how the lost and found functions as I'm using it so that I can use it efficiently.
- User Story 9: Item filter
 - As a user, I want to be able to filter items by their name and other attributes.

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting.

- User Story 10: Follow up
 - As a user, I want to be notified whenever important changes are made to my listing.
- User Story 8: Duplicate Detection
 - As a system, I want to detect and suggest merging duplicate item listings to keep the platform clean.
- User Story 5: Item claimed
 - As a user, I want to be able to claim the item so that listings remain up-to-date and accurate
- User Story 3: Report Option
 - As a user, I want to report any suspicious or false listings so that moderators can review and maintain the quality of the platform.
- User Story 1: Web Notifications
 - As a staff member, I want to receive notifications on changes on the items, so that I can be on top of the recent changes.

Appendix B - Shortcomings

Due to time constraints we weren't able to fully implement all the features that we would've liked for this project. Below is a list of the minor and major shortcomings:

Minor:

- No notification system implemented.
 - No support for communication between users.
 - *Communication is solely mediated by admin.*
 - No analytical panel
 - *Admins cannot track usage or view the history of listed or claimed items.*
-

Major:

- Two-factor authentication or email verification is not supported.
 - *If a user loses access to their account, they cannot reset their password.*
 - *Emails could be used to claim items by unauthorized individuals.*
- No sanitization of file uploads.
 - *Files are uploaded directly to our backend without being scanned/checked, potentially allowing injection of malware.*

These limitations were identified during development, and may be addressed in future iterations of the application.

Appendix C - Feature Wishlist

While the current version of the Campus Lost and Found application provides core functionality for reporting and managing lost items, several features were identified for future implementation. These are categorized into short term enhancements and long term innovations that would improve scalability, user experience, and system intelligence.

Short Term Goals:

- **Mobile Admin GUI**

The current admin interface is limited to the web application. Adding a mobile-friendly view and authentication system would provide more flexibility for staff managing lost items on the go.

- **Individualized Lost & Found Pages**

Currently, all reports are stored in a single centralized database. A future version could partition this database to allow different departments or organizations to manage their own dedicated lost and found pages.

Long Term Goals:

- **Machine Learning for Post Tagging**

Training a machine learning model to auto-tag posts (e.g., by item or category) would improve filtering and search functionality for users.

- **Interactive Map View**

Incorporating a map feature would allow users to view nearby lost item reports and improve location-based filtering.

- **Analytical Tools**

Admins could benefit from tools that track item trends such as the most commonly lost items or frequently reported locations to inform prevention or awareness efforts.

These enhancements are intended to extend the platform's functionality and adaptability as the system matures and scales across different campuses or departments.