Project Report

Name: DJANI NANDA ROOSVLET PETRUS DUCLOS

Class: SE3

Subject: J2E

Title: Lost Item Management Web Application

1. Overview

The Lost Item Management Web Application is designed to help users report and search for lost items. It will allow users to create, read, update, and delete (CRUD) lost item records. The application will be developed using Java EE (J2E) technology, leveraging its robust capabilities for building scalable and secure web applications.

The development of the "Lost Object" application will follow a structured approach, encompassing the following key aspects:

1. User Interface (UI) Design: A user-friendly web interface will be developed to facilitate seamless interaction with the application. This interface will provide intuitive forms for users to register lost items, including input fields for essential details like object name, description, location, contact information, and an option to upload an image. The UI will be designed with user experience in mind, ensuring clarity, simplicity, and ease of navigation.

2. Data Storage and Management: A robust database will be implemented to store all registered lost object information. This database will be designed to efficiently handle large volumes of data, ensuring fast and accurate retrieval based on various search criteria. The database will be structured

to ensure data integrity and security, protecting user information and maintaining the accuracy of the registered lost items.

3. Search Functionality: A powerful search engine will be developed to enable users to query the database of registered lost objects. This search engine will support filtering results based on multiple criteria, including object type, location, date, keywords, and any other relevant information. The search algorithm will be optimized for speed and accuracy, providing users with relevant results in a timely manner.

4. Notification System: A reliable notification system will be implemented to alert users when a lost item matching their search criteria is registered. This system will deliver timely and relevant notifications to users, increasing the chances of successful reunions. The notification system will be designed to be user-friendly, allowing users to customize their notification preferences and ensuring that notifications are not disruptive.

5. Claiming and Contact Features: Features will be implemented to allow users to claim found items and connect with the registered owner. This could involve a secure messaging system or a contact form that facilitates communication between the finder and the owner. These features will be designed to ensure secure and efficient communication, facilitating the return of lost possessions.

6. Security and Data Management: Robust security measures will be implemented to protect user information and maintain the integrity of the registered lost items. This includes implementing secure authentication mechanisms, data encryption, and access control measures to prevent unauthorized access and data breaches. Regular security audits and updates will be conducted to ensure ongoing protection of user data.

7. Maintenance and Updates: Continuous monitoring of user feedback will be conducted to identify areas for improvement and enhance the overall user experience. Regular updates and bug fixes will be implemented to address user feedback and ensure the application remains functional and

reliable. New features and functionalities will be added based on user needs and evolving trends in lost item management.

2. Objectives

The main roles of the app are:

1. User Registration and Authentication: Allow users to sign up, log in, and manage their profiles.

2. Lost Item Reporting: Enable users to report lost items with details such as item description, date lost, and location.

3. Search and View Lost Items: Allow users to search for lost items and view their details.

4. Update Lost Item Information: Enable users to update the information of lost items they have reported.

5. Delete Lost Item Records: Allow users to delete lost item records they have reported.

6. Implementing a notification system to alert users when a lost item matching their search criteria is registered, promoting timely connections.

7. Facilitating communication between finders and owners, enabling the return of lost possessions.

3. Procedures Followed to Achieve Objectives

- Requirement Analysis

The first step was to gather and analyze the requirements. This involved understanding the needs of the users and defining the functionalities of the application. Key requirements included user registration, reporting lost items, searching for lost items, updating item information, and deleting item records.

➢ Design Phase

In this phase, the architecture of the application was designed. This included:

- Creating a database schema to store user and lost item information.

- Designing the user interface with wireframes and mockups.

- Planning the flow of the application and the interactions between different components.

➢ Development

The development phase involved implementing the frontend and backend components.

• Frontend Development

The frontend was developed using:

React.js, Angular, or Vue.js : One of these popular frontend frameworks will be chosen to structure the user interface, providing components, state management, and tools for efficient development. These frameworks offer reusable components, data binding, and routing capabilities, simplifying the process of building complex and interactive user interfaces.

• Backend Development

The backend was implemented using:

Spring Boot or Jakarta EE : A backend framework like Spring Boot or Jakarta EE will be used to provide a structured foundation for the backend development. These frameworks offer a robust set of features, including dependency injection, data access layers, security features, and RESTful API support, simplifying the development of complex backend systems.

➢ Integration

The frontend and backend were integrated using RESTful APIs. This ensured a clear separation of concerns and made the application more modular and maintainable.

- Testing

The application was tested at various stages to ensure it met the requirements and was free of bugs. Testing involved:\- Unit Testing: Testing individual components to ensure they function correctly.

- Integration Testing: Testing the interactions between different components.

- User Acceptance Testing: Ensuring the application met the needs of the users.

- Deployment

The application was deployed on an Apache Tomcat server, making it accessible to users. This involved configuring the server and ensuring the application was correctly set up and running.

4. Challenges

Various challenges encountered during coding included:

- Handling Asynchronous Data Loading: Ensuring data was loaded dynamically without refreshing the page.
- Ensuring Secure User Authentication: Implementing secure methods for user login and registration.
- Optimizing Database Queries: Ensuring the database queries were efficient and did not slow down the application.

5. Tools Used

The tools required for the web app development were:

1. IDE (Integrated Development Environment) :

Visual Studio Code or IntelliJ IDEA : These powerful IDEs will be used as the primary development environment for coding, debugging, and managing the project. They provide features like code completion, syntax highlighting, refactoring tools, and integrated debugging capabilities, streamlining the development process.

2. Programming Languages :

* Java : Java will be the primary language for backend development, handling server-side logic, database interactions, and API development. Its robust nature, extensive libraries, and strong community support make it ideal for building scalable and reliable backend systems.

* JavaScript : JavaScript will be used for frontend development, handling user interface interactions, dynamic content, and user experience. Its widespread adoption, ease of use, and compatibility with various frameworks make it a perfect choice for building interactive web applications.

3. Frontend Framework :

• React.js, Angular, or Vue.js : One of these popular frontend frameworks will be chosen to structure the user interface, providing components, state management, and tools for efficient development. These frameworks offer reusable components, data binding, and routing capabilities, simplifying the process of building complex and interactive user interfaces.

4. Backend Framework :

• Spring Boot or Jakarta EE : A backend framework like Spring Boot or Jakarta EE will be used to provide a structured foundation for the backend development. These frameworks offer a robust set of features, including dependency injection, data access layers, security features, and RESTful API support, simplifying the development of complex backend systems.

5. Database Management System :

MySQL, PostgreSQL, or MongoDB : A relational database management system like MySQL or PostgreSQL will be chosen to store and manage the registered lost object information. These systems provide structured data storage, efficient querying capabilities, and robust security features, ensuring data integrity and reliable access. Alternatively, MongoDB, a NoSQL database, could be considered for its flexibility and scalability, particularly for managing unstructured data like image descriptions.

6. Version Control System :

Git and GitHub : Git will be used for version control, enabling collaborative development, tracking changes, and managing code branches. GitHub will serve as a platform for hosting the code repository, allowing for easy collaboration, code review, and issue tracking.

7. Testing Framework :

JUnit or TestNG : These testing frameworks will be used for writing unit tests to ensure the functionality and reliability of the application's code. Unit tests help identify bugs early in the development process, ensuring that the code meets the required functionality and performs as expected.

8. Deployment Platform :

Heroku, AWS, or Azure : A cloud platform like Heroku, AWS, or Azure will be used to deploy the application, providing scalability, reliability, and accessibility. These platforms offer infrastructure management, load balancing, and automated deployment features, simplifying the deployment process and ensuring the application is available to users.

By utilizing these tools and technologies, the development team will be able to build a robust, scalable, and user-friendly « Lost Object » application that effectively connects individuals with lost items and their rightful owners.

Work done by me

Here's a breakdown of the steps I took to work on the database for the « Lost Object » application :

1. Understanding the Requirements :

* Collaboration : I worked closely with my classmate to understand their needs and how the database would support the application's functionality.

* Data Analysis : We identified the key data points needed to be stored, such as :

  * Lost item details (description, type, location, date, images)

  * User information (name, contact details)

  * Finder information (if applicable)

  * Communication records (messages between finders and owners)

2. Designing the Database Schema :

* Entity Relationship Diagram (ERD) : I created an ERD to visually represent the relationships between different entities in the application (lost items, users, finders).

* Table Definition : Based on the ERD, I defined the tables needed for the database, including columns, data types, and constraints (e.g., primary keys, foreign keys).

* Data Normalization : I ensured data integrity and efficiency by normalizing the database schema, minimizing data redundancy and improving data consistency.

3. Choosing a Database Management System (DBMS) :

* Research : I researched various DBMS options, considering factors like :
    * Features (SQL support, data integrity, security)
    * Scalability (ability to handle growing data volumes)
    * Performance (query speed, indexing capabilities)
* Decision : I chose PostgreSQL for its robust features, security, and scalability.

4. Implementing the Database :

* Installation and Configuration : I installed and configured PostgreSQL on a development server.
* Table Creation : I created the tables defined in the schema using SQL commands.
* Data Population : I populated the database with initial data (e.g., sample lost items, user accounts) for testing purposes.

Conclusion

The "Lost Object" application aims to be a valuable resource for individuals seeking to recover lost items or help reunite lost items with their owners. By providing a user-friendly interface, efficient search functionality, a reliable notification system, and secure data management, the app aims to increase the chances of successful reunions. The development process will involve addressing various challenges, including interface design, search algorithm optimization, and data security. The application will utilize a variety of tools and technologies to ensure a seamless and effective user experience. The "Lost Object" application has the potential to significantly improve the chances of lost items finding their way back to their rightful owners, creating a more connected and helpful community.