Travis Heavener, Matthew Voynovich, Yiqi Liang

Term Project Final Writeup

ITWS 1100: Intro to IT

# Proposal Summary

The project at RPI aims to centralize its lost and found system online, addressing the issue of multiple unconnected services across the campus. This web application (“Hound”) will enable users to report lost or found items and search for items in the system, making it easier to recover lost belongings. While proactive measures to prevent losing items are challenging, this reactive approach is more feasible.

Hound prioritizes data handling with the lost and found database and JavaScript functionality, including form validation. HTML and CSS design are also vital. The application’s structure leads users to a landing page, explaining the app’s purpose and allowing users to report lost items or declare found ones. Users can browse or report lost items, with details like time, date, and location matching the database entries. A similar approach is used for found items. The website’s physical structure includes a root directory with separate directories for lost and found items, ensuring easy navigation and efficient resource organization.

# Project Details

In order to solve RPI’s issue with lacking a centralized lost and found database (“Public Safety.”), the development of this term project, colloquially referred to as “Hound” as in “Lost and Hound”, aims to fill this void. For one, the importance of allowing easy access to a lost and found database for all students and faculty across campus means that should someone misplace an item, they can rest easier knowing that it will be significantly easier to locate an item via a centralized collection compared to localized receptacles across campus. In addition, the intentionally mobile-friendly design of Hound will make submission of each item entry significantly easier, preventing users from giving up on their lost or discovered belongings out of frustration or difficulty navigating the application. Due to the fact that Hound will allow for users to upload an image, the time they lost or found the item, the location, and contact information, users will be able to easily search for lost items and catalog items they’ve found, as well as deposit them at the appropriately designated place (such as the RPI Office of Public Safety).

To demonstrate this proof of concept, two test “personas” were developed: Reese S. Peace, a student who has lost their lime green AirPods case on campus, and Prof. Candice “Candy” Korn, a physics professor who frequently finds misplaced items on jogs while training for a 10K. In Reese’s situation, having lost his AirPods between classes caused him to panic, searching everywhere but not knowing where to look besides his bag. After being informed of Hound’s existence, Reese was able to navigate to and through the website where he was able to submit a lost item declaration form, allowing his AirPods to be discovered by another user soon after. Because of Reese’s quick action, he was reunited with his AirPods by retrieving them from the Office of Public Safety that same day. As for the Professor, Candy’s frequent discovery of items from around campus on her training jogs for a 10K made her a perfect candidate to declare these items as discovered when she encountered them. After being informed by one of her students, Hound became a prominently used application on her mobile phone, as Candy would quickly snap a picture, attach her information, and note where and when she found the item. At the end of her jogs, she would grab the items from her bag, deliver them to the Office of Public Safety, and return to her day. In both instances, the users were able to easily and quickly search for and declare items as lost and found, enabling campus-wide access while still retaining application usability and efficiency.

# Project Plan

In order to achieve this solution, Hound will employ a MariaDB/MySQL database, accessed via SQL queries that would be called at the users’ discretion. For example, a user who would like to report an item they’ve found would have the ability to attach all the necessary information–such as an image of the item, the location found, and preferred contact information–which would then be submitted in a POST request to the PHP script. Similarly, a user who would like to search all lost items for a particular item by its description and characteristics would follow a similar procedure, by defining the characteristics of the item and submitting a GET request to the PHP script.

The following table outlines milestones and deliverables for this project over the semester alongside their expected and actual completion date:

|  |  |  |
| --- | --- | --- |
| **Deliverable/Milestone** | **Expected Completion** | **Actual Completion** |
| Create database | Oct. 27th | Oct. 27th |
| Wireframe for web pages | Nov. 5th | Nov. 6th |
| Create PHP scripts to query database | Nov. 5th | Nov. 5th |
| Integrate front/backend w/ PHP & JS | Nov. 12th | Nov. 5th |
| Format web pages with CSS | Nov. 19th | Nov. 23rd |
| Final UI Touches | Nov. 26th | Nov. 30th |
| Term Project Final Presentation | Dec. 4th | Dec. 2nd |
| Term Project Submission | Dec. 7th | — |

Lastly, the development is divided among team members: HTML page design, headed by Yiqi Liang, backend management with the database and subsequent database mediator PHP scripting, headed by Travis Heavener, and general bug fixes and all around maintenance and development by Matthew Voynovich.

# Architecture and Methodology

In order to bring Hound to fruition, several key components needed to communicate harmoniously to deliver content to the user and process and store content uploaded from the user. The Hound application consisted of three core HTML documents: one for the landing page that would link to the other two pages, one for lost items and one for found items. On the home page, contact information for our development team as well as the RPI Public Safety Office were attached, along with a mission statement and the anchored links to navigate between pages. The lost items page displayed all items currently uploaded as found around campus, with a form to submit a new lost item on the bottom. Contrarily, the found items page displayed items declared as lost by their owners and allowed form submitters to post a new listing for an item they had found on the bottom. The forms themselves allowed users to attach necessary information about themselves, where and when they had lost or found the item, an image of the item, and any additional comments.

To collect data from these HTML forms, a mixture of PHP and JavaScript was employed. The JavaScript was called on form submission and would process the data for it to then be sent and processed on the backend by PHP. The JavaScript also handled potential errors during HTTP requests and was responsible for executing functions after the request had completed. As for the PHP, the incoming form data was then processed and inserted into prepared SQL statements (to prevent SQL injection attacks) and any uploaded images were renamed and uploaded to a folder on the server. The SQL statements, when executed, stored the data in a backend, relational MySQL/MariaDB database, with a table for lost items and a table for discovered items. Similarly, when loading a page on the front-end the database would be queried for the most recently uploaded items, limiting the call to a fixed value as specified in the JavaScript.

To highlight specific areas of this project, the two main primary and secondary areas of focus were initially labeled as serving data to and from the database (project Area #5) and providing functionality and form with JavaScript (Area #2), respectively; however as the project progressed the secondary area of focus shifted to HTML and CSS page design (Area #1), due to the fact that most of the JavaScript solely sent and received data from the PHP scripts and then added content to the document dynamically.

# Development Obstacles & Solutions

Throughout the web and mobile application development process, a handful of challenges are destined to arise, as was the case with this project. For one, finding the appropriate time to all meet and develop was often a challenge. Scheduling around midterms and class schedules seemed to be a more difficult task than it may seem, yet we were able to implement a functional and effective version of Hound in the time allocated nonetheless.

In addition to minor scheduling challenges, a handful of security issues were tackled with PHP scripting. As with any remotely secure SQL-powered database application, the use of prepared statements to sanitize incoming data was crucial to not allow for SQL injection attacks to take place. Along with prepared SQL queries, ensuring that the server’s database management credentials (accessed via PHPMyAdmin) were not exposed was crucial, albeit less of a concern for the scope of this project. By using environment variables on local machines and Microsoft Azure’s virtual machines, the necessary credentials for the database were not exposed in plaintext PHP, adding an extra layer of security to the backend.

With any mobile-friendly application, there arise many issues with mobile formatting, and this project was no exception. Google Chrome’s Development Tools allowed for the use of simulated mobile phone views from the browser, which enabled us to easily and effectively develop for the target audience of mobile users. Despite this, however, responsive CSS design for both web and mobile use was a critical yet difficult challenge for making Hound accessible. Through the creative use of CSS units and text size-related formatting, the elements on each document and page were laid out in a manner that functioned appropriately and aligned with our initial mockups of the user interface, or UI. Although future development would enlist more extensive mobile development, the current implementation of such CSS and HTML code completes the designated objectives up to par with our initial UI designs.

# Project Summary

At the end of our development period, we finished three pages for our website. On our lost and found pages we were able to create working forms that uploaded lost and found item information onto our database alongside the output of those items. All three pages were formatted with css and made to look simple and intuitive while also retaining a good deal of function.

As is the case with any web and mobile development project, a handful of lessons–both lessons to be and not to be repeated–were learned by the team as a whole. For one, ensuring database security and user privacy was a key concern for developing the backend of the application. By implementing prepared SQL queries in PHP and requiring specific fields to be completed directly in the HTML, form data was processed and stored effectively and securely, despite not being an imperative concern for the scope of this project. Similarly, the credentials for accessing the database via PHPMyAdmin were also protected and not exposed in plaintext via use of server environment variables, which remained consistent across all devices used for testing and development.

Aside from backend security, the frontend also provided a number of key lessons for future projects. Most notably, intuitive CSS design was learned through the implementation of visually-pleasing and comprehensive form design. By placing elements on each HTML document in easily accessible locations, not only is the user experience improved but the developer’s perspective of what effective design looks like is also improved–which was the case. In addition, the use of dynamic HTML injection for displaying content from the backend via HTTP requests tasked our team with learning and enriching our knowledge of jQuery’s AJAX calls. Not only was formatting and designing this system of HTTP requests a challenge but displaying the content returned from the database proved itself to be a design challenge. However, the use of the team’s collective knowledge and understanding of what effective web design looks like allowed us to implement and inject HTML elements dynamically and, most importantly, in a visually pleasing manner.

In the future, we would like to further optimize and develop Hound to become the best version of itself. Alongside improvements to the general look and layout of the website, some of our quality of life changes would include having a filter option for the output of data from the database, dropdown navigation between pages, and being able to claim your items and take stuff off the database. To make sure that people do not claim items that are not theirs we will hold a log table of who picked up what items and their contact information for if they picked up the wrong item or something that was not theirs. This table would be displayed on the website, but unlike the others you would not be able to remove items from it. Additionally to make the process more secure we would add email verification on the forms. This would allow us to then send emails to people about potentially matching items.

If this project does end up going far enough and getting more use we would like to eventually get in contact with public safety and potentially merge it into something that RPI officially endorses and uses. We could achieve this by continuing our development by utilizing RPI’s center for open source class as a bridge between us and the Rensselaer system.

Works Cited

“Public Safety.” *Services | Public Safety*, Rensselaer Polytechnic Institute (RPI),

publicsafety.rpi.edu/services. Accessed 1 Oct. 2023.