SMALL EQUIPMENT CHECKOUT LOCKER SOFTWARE

Laura Mejía, Ben Johnson, Camille Cramer, Ainara Machargo del Rio, and Jon González

OVERVIEW

Problem

ETG is not open 24/7 but students are often times working on projects after ETG is already closed for the day.

Solution

A system of lockers with rental equipment available all hours for short term rental capabilities.

Use case

Students: will be able to check out small equipment from GUI after hours
ETG Staff: will be able to provide students even more support, also will be able to maintain lockers through an admin page.

DESIGN REQUIREMENTS

Functional Requirements:

- Self-service touchscreen.
- Automatic locker opening.
- Functional Admin site.

UI Requirements:

- **User friendly interface:**
  - Clear instructions and visuals.
- **Reservation website:**
  - Uniform ISU HTML template.
  - ISU credentials login.
- **Admin site:**
  - Easy navigation and equipment management.
  - Accessible rental information.
- **Student Experience:**
  - Spinner for component selection.
  - Duration of rental specification.

Database Requirements:

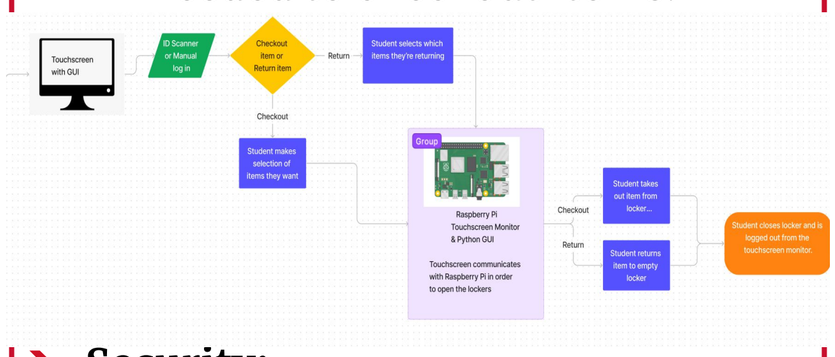
- Hosted on a Raspberry Pi.
- Log equipment checkout and return information.

TECHNICAL DETAILS

- The project uses industry-standard tools, including a Javascript/HTML/CSS front end, Flask API backend, and a MariaDB database on a Raspberry Pi.
- The Flask API leverages several python libraries, including SQLAlchemy to communicate with the database, Marshmallow to serialize data, and the ETG provided locker control software to manage the locker hardware

DESIGN APPROACH

- **Front-end webpage:**
  - Allows users to login, select items to checkout, and open/close the locker.
  - Includes admin page for ETG staff to monitor checkout history.
  - Displays a confirmation page and sends user a confirmation email.
- **Database:**
  - Stores all user data needed for the application.
- **Raspberry Pi:**
  - Runs a server to connect the front-end and database.
  - Operates the locker hardware as needed to checkout items.



- **Security:**
  - Authenticate users via netID to ensure only ECpE students can use the locker
  - User input is sanitized to prevent SQL injections
  - Error handling to prevent data leaks and application crashes
- **Project Management**
  - We used a hybrid waterfall + agile project management approach
  - Adopting this approach allowed us to balance the dependencies between the front and backend
  - This resulted in additional flexibility when making changes

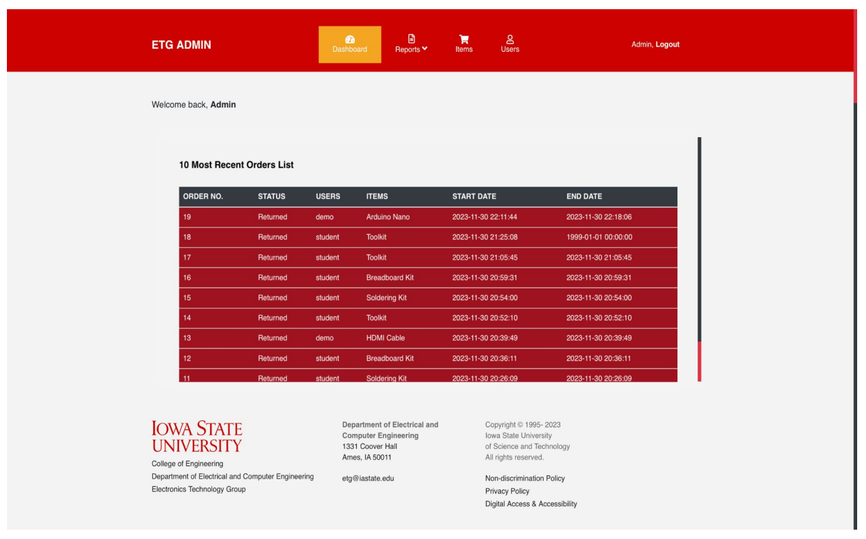


Figure 1: Admin Page.

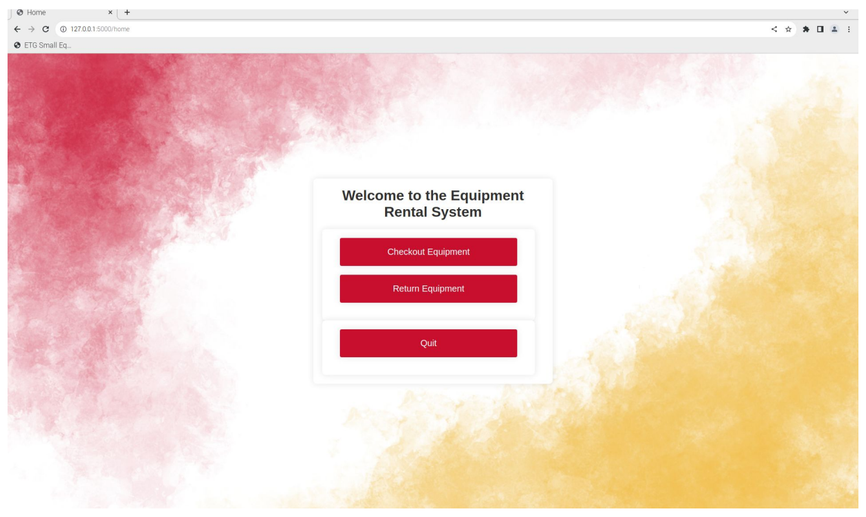


Figure 2: GUI Home Page

TESTING

- **Frontend:**
  - Checking look and feel on various web services.
  - Ensuring user interface is responsive and intuitive.
  - Testing for visual consistency and ease of navigation.
- **Backend:**
  - Ensuring data integrity and efficient performance.
  - Evaluating the Flask API and MariaDB database for smooth interaction.

RESULTS & IMPACT

- **Results:**
  - Front-end is user-friendly, enabling students to easily check out equipment from the GUI after hours.
  - Consistent display and high responsiveness enhance user experience.
  - Back-end, with Flask API and MariaDB, ensures reliable performance, even during peak usage by students.
- **Impact:**
  - Facilitates after-hours equipment checkout for students, increasing accessibility.
  - Empowers ETG staff to provide enhanced support and efficiently manage equipment.
  - Admin page functionality allows staff to maintain and monitor lockers effectively.

CONCLUSION

- **Integration and Access:**
  - Efficiently merged technology with user needs.
  - Simplified student access to after-hours equipment.
- **Staff Empowerment:**
  - Enhanced ETG staff's management capabilities.
- **Future Outlook:**
  - Complete the implementation of the admin page by integrating proper authentication into the login page and updating the users page accordingly.