Labriola Training Tracker: Final Report

Labriola Lab Rats

Bethany Boehmer
Ben Isenhart
Tristen Erpelding
Alex Heckman

Revised December 11th, 2025

CSCI 370 Fall 2025

Prof. Donna Bodeau

Table 1: Revision history

Revision	Date	Comments
Initial Requirements	Sept. 7th, 2025	Detailing the fundamentals for our project, including requirements, risks, completion criteria, and a short bio for each team member
System Architecture	Sept. 21st, 2025	A detailed description for our current understanding of the major problems and intended solutions involved with our project, along with possible concerns
Ethical Considerations	Oct. 10th, 2025	An overview of the potential ethical problems associated with our project and how we plan to best mitigate these concerns
Results Intermediate	Oct. 25th, 2025	An updated version of our implemented, work in progress, and to-do features now that coding is heavily underway. Includes the current results of our testing and lessons learned
Future Work & Lessons Learned	Nov. 30th, 2025	Our results have been updated with their final state. This work has been reflected on to determine possible work for future teams and our own team's experiences with this project.
Major Final Paper Revisions	Dec 7th, 2025	Numerous changes reflecting peer review feedback, primarily in terms of the paper having multiple perspectives throughout, reflecting different stages of development.
Finishing Touches	Dec 11th, 2025	Final minor revisions and updates to round out the semester

Table of Contents

I. Introduction

The Labriola Innovation Hub is a campus space dedicated to design, creation, and innovation, and is comprised of multiple shops and spaces that are open to all students and staff. Certain shops and machines require training for users to safely enter and operate; a user must display valid training to use a machine when applicable. To streamline work for TAs and managers, Labriola had a prior field session team build an application to display student training status concisely on workstation screen. Our team was tasked with continuing to develop new features for this project. Specifically, we aimed to and have successfully implemented the following:

- Added additional Canvas API functionality, allowing for TAs to update a student's grade from the application directly.
- Created an SQL database to map Canvas training quizzes to their IDs.
- Integrated OreConnect to the sign-in process, allowing for Blastercard tap-scans to serve as a new sign-on method.
- Renovated much of the application UI to display additional useful information to TAs.
- Added SSO sign-on for Teachers & TAs for better authorization and authentication protocols.

II. Functional Requirements

These updates to the InnoHub's check-in system have been built on the current application. The old application pulled data into a CSV based on data from Labriola's Canvas page before displaying the data on a web application. We have continued to use this web application while adding new features such as updating training statuses and an SSO system for TAs and managers to sign in. We also have implemented a subroutine for RFID card scanners such that students can scan their Blastercards to check into a shop. Additionally, we were hoping to migrate the existing solution to a server-based solution, moving away from Raspberry Pis. However, due to issues regarding a software review with ITS, this has been postponed for a future field session team and is elaborated on in more depth later in this paper.

III. Non-Functional Requirements

Our team had several requirements we deemed necessary for a satisfactory system. We wanted to move away from being hosted on an individual Raspberry Pi and onto a singular cloud instance. We wanted to utilize Okta for our SSO's to make them secure and integrate with Mines existing infrastructure. We wanted to ensure our check-in system follows all ITS requirements so that it is no longer under the preview of "shadow IT" and can easily have continued support. We also wanted trainings to expire after a certain amount of time and the web application to display who has checked into a shop within the last hour. Finally, we have written our backend with JavaScript and our frontend with JavaScript and Vue.

IV. Risks

The major risk that this project faced is that it depended a lot on Mines ITS. While we initially wanted to put the application on the cloud, that required going through ITS for both the infrastructure and a software review. Since that process was largely unfruitful, development of these features was at risk of not being able to meet some of the requirements. This included accessing Labriola's Canvas course & OreConnect page with scoped API keys and securing a Microsoft Azure API key for the web application,

Additionally, we had some concerns with renovations being done to the Blastercard tap-scan system used across campus. We do not anticipate these changes impacting our project in coming years, but it will be necessary to ensure that both newer and older versions of Blastercard scans work with our final product. As the Blastercard system continues to develop, ensuring that scans are still mapped appropriately through OreConnect is necessary.

V. Definition of Done

Our team considers a new feature or piece of software done once the code has been written, the feature works, the code has tests written, and the feature is physically tested and reviewed from the perspective of an end user and approved by our clients. Additionally, the new feature must not break necessary features that existed previously.

For our product specifically, these necessary features included the following: the application must maintain all features which existed prior to our project, including pulling data based on Labriola's Canvas page and displaying the data on a web application. Additionally, the following features were required for our clients to be satisfied with our project and our work to be considered done. We have implemented a database as the landing point for the Canvas data rather than the old solution (a CSV file). We are utilizing a SSO system for TAs and managers to sign-in. We have created an RFID scanner routine which allows for scanning the students' Blastercards to check into the shops which utilizes ITS's mapping (via OreConnect) for converting the scanned value into the user's Mines email.

VI. System Architecture

Overview

Our system is centered around a web application that utilizes frontend, backend, and physical components. Below is the high-level design for the system.

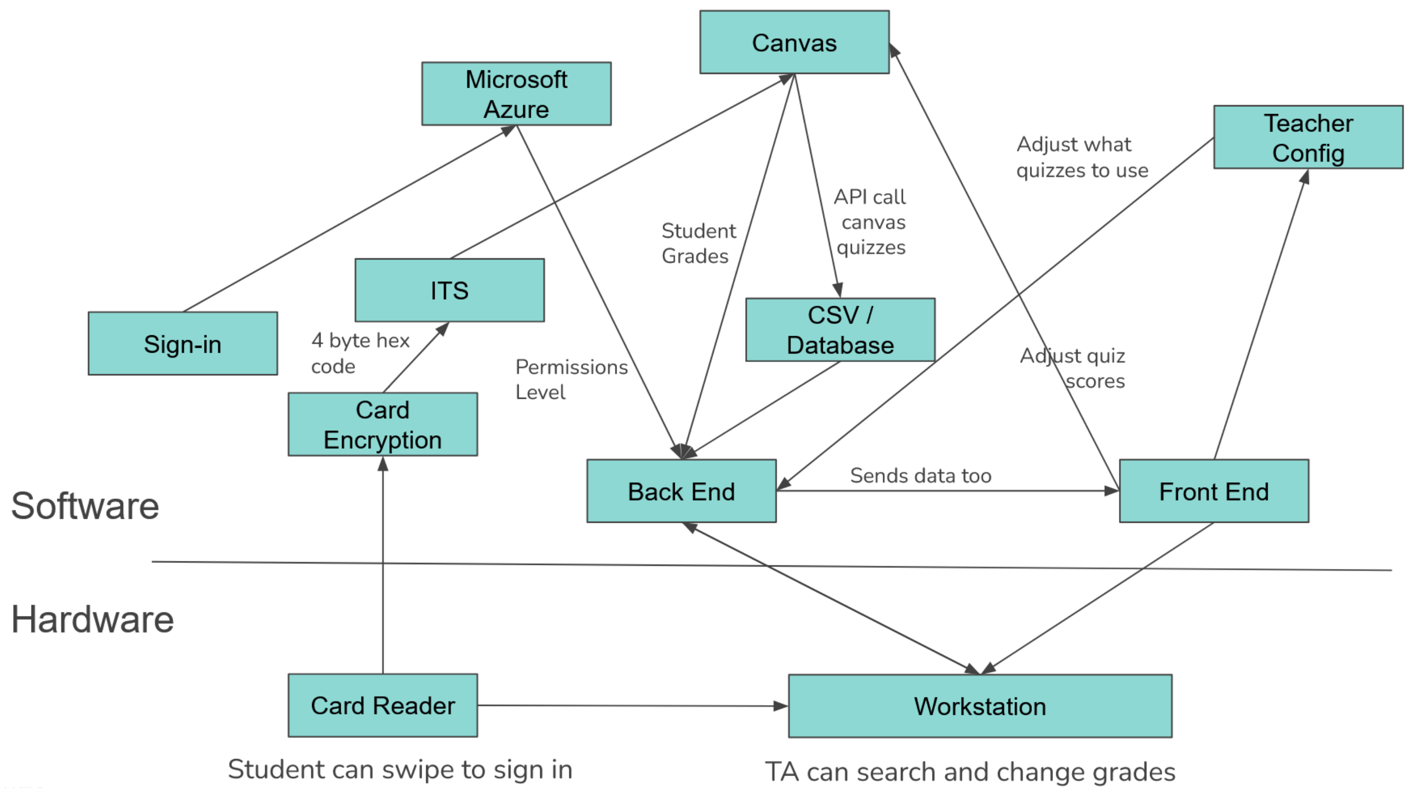


Figure 1: High Level System Architecture

The hardware components describe what users can directly access outside of the application, whereas the software components describe the pieces that run the application and make it work. The card scanner scans Blastercard IDs and maps emails to students as an alternative to having enter login information directly into the application. The workstation is the machine running the application and is what a user can interact with. The Canvas page is the where the student data itself resides and where TAs and managers can edit grades and quizzes. Previously, Canvas was used to pull data for a CSV for the backend and to make API calls about queried students. We have now implemented a database instead and added the ability for changes made to student grades on the frontend of the application to alter grades in Canvas. The API calls about queried students remain the same, as student PII is not something we would wish to store in a database due to FERPA. Below is a more detailed insight into how the database fits into the architecture.

Backend

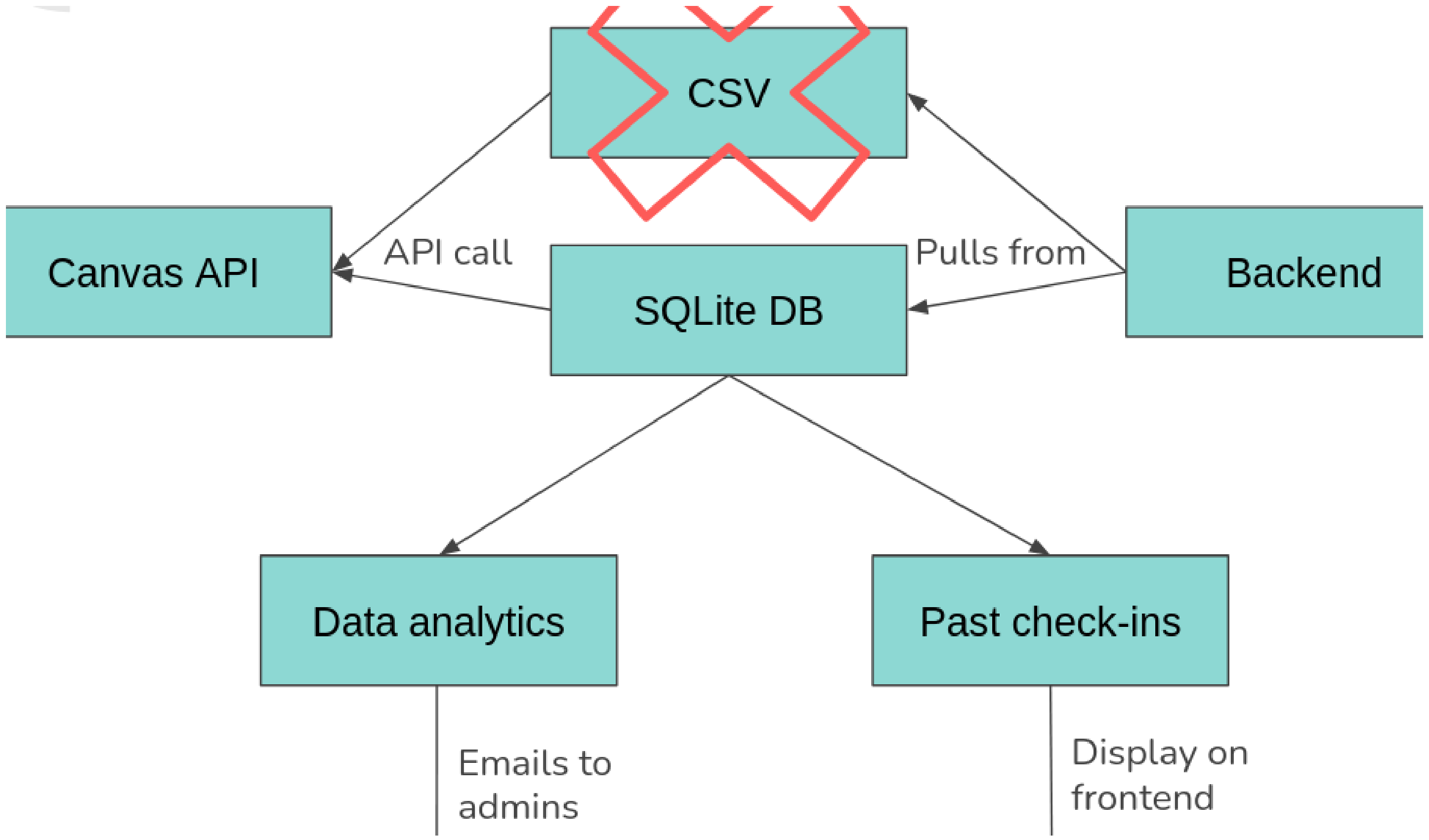
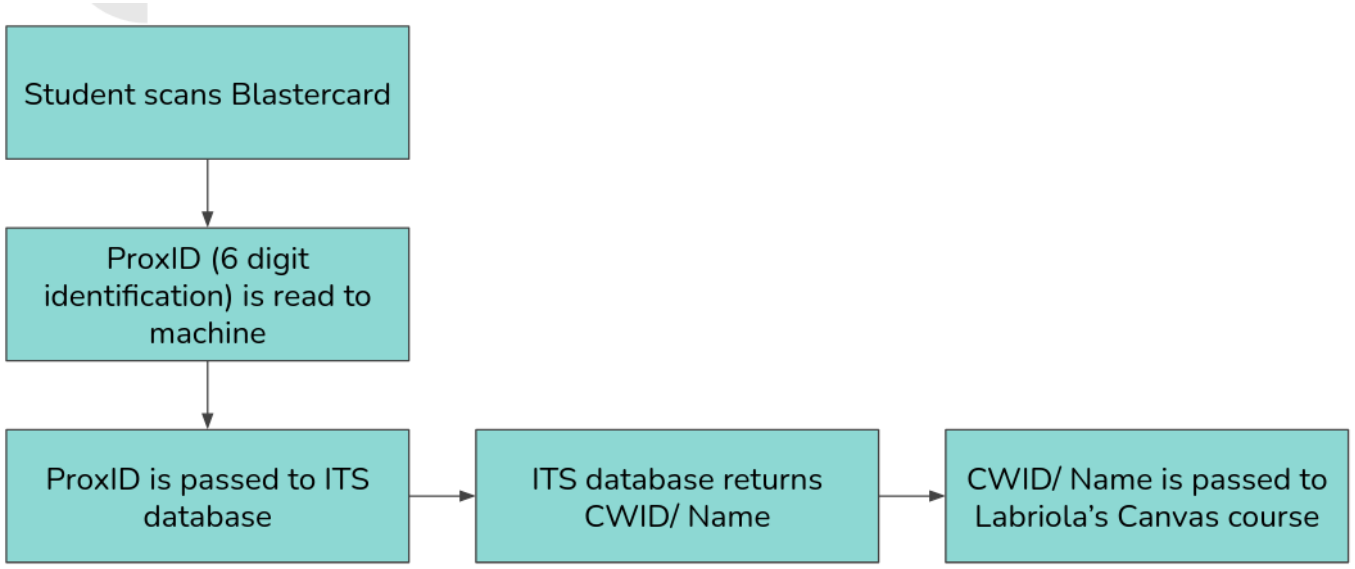


Figure 2: Reimplementation of Database

The database stores data on the Canvas quizzes, which allows quiz IDs to be mapped to the training it corresponds to, and check-in data about the number of students who have checked into a space at what time. This data is then used to collect analytics of student check-ins in certain spaces, per the request of the managers, and to display the students who have checked into a space in the past hour.



Student scans Blastercard
ProxID (6 digit identification) is read to machine
ProxID is passed to ITS database
ITS database returns CWID/ Name
CWID/ Name is passed to Labriola's Canvas course

Figure 3: Blastercard Flowchart

The Card Scanner section of the system utilizes an RFID card scanner so students entering a space can simply tap a Blastercard. From there, the ProxID (numeric identifying value) is read as keyboard input to the machine and redirected to ITS's database through OreConnect. It is then mapped to the student's email, although full name or CWID could've been done as well. This information is returned to our application and can then be passed on again to Canvas for querying, thereby replacing the need for a student to type in their credentials to view training statuses and sign into a workspace.

It is important to note that this process was highly dependent on a software review with ITS which was contingent upon a variety of factors (accessible system, continued maintenance, secure/FERPA-compliant code, etc.). Without a successful software review, ITS is unwilling to grant access to their database mapping ProxIDs, meaning that the RFID card readers for Blastercards become functionally useless. Since we were unable to get a successful software review during the Fall, OreConnect has been used as a middleman, because have gone through this software review process.

Using OreConnect in this manner requires storing the cookies of a user registered with "Officer" permissions in Labriola's OreConnect page. When the application starts, the TA or manager is prompted to sign into OreConnect to store their cookies for the duration of the shop being open. Using their cookies, we are posting sign-in requests to a hidden event which registers the user. From this user registration, we are parsing the page's HTML to read their email as a unique identifier to link the Blastercard scan to a Canvas profile.

The database and card scanner were integrated in the backend of our system, which is written in JS and makes API calls needed from Canvas. This sends data to the frontend, which runs a web application on localhost to display on a workstation. Additionally, the frontend may now send data as well as request it, allowing TAs to change Canvas grades directly from the application. The frontend also has SSO integration with a Microsoft Azure instance, allowing TAs and managers to sign in via Mines Microsoft logins instead of having a set password on the application. Below is the system for how our SSO integrates with the application.

Authentication/Authorization

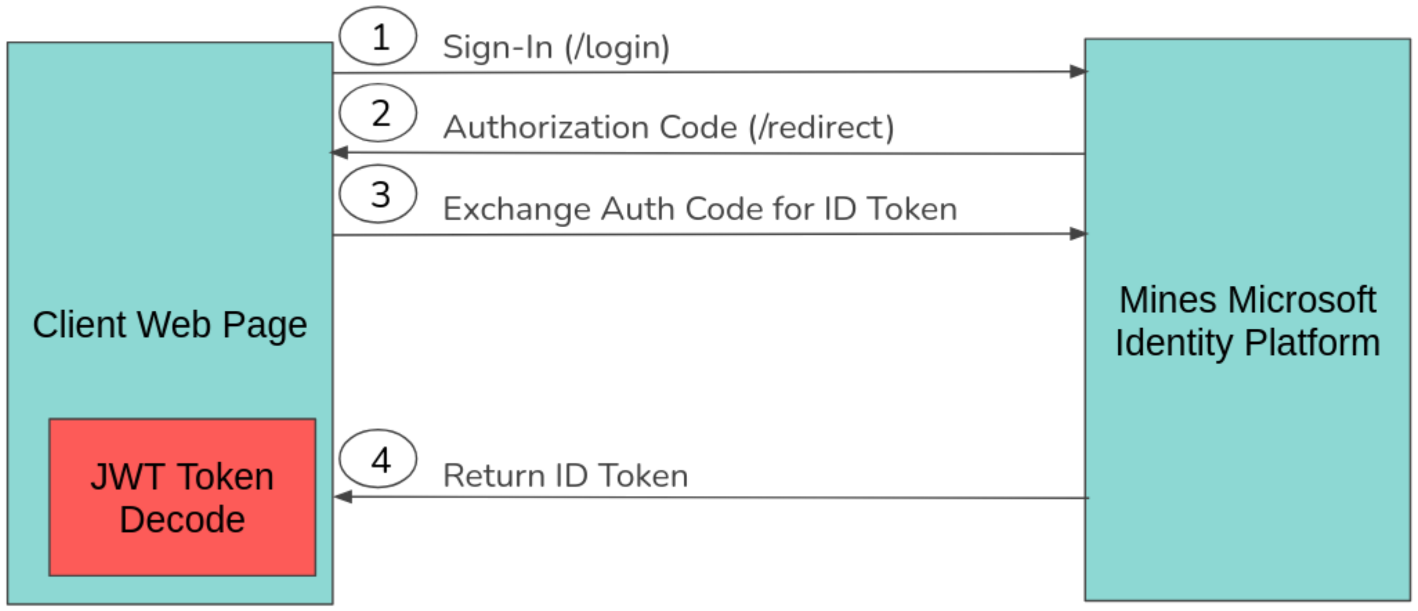

1
Sign-In (/login)
2
Authorization Code (/redirect)
3
Exchange Auth Code for ID Token
Client Web Page
Mines Microsoft Identity Platform
JWT Token Decode
4
Return ID Token

Figure 4:

The previous iteration of the program planned to use Okta Verify as the SSO and assign user roles based off their permissions in Canvas. This method would still work - however, it would require further approval from ITS for very little benefit. Instead, we switched to Mines Microsoft Azure utilizing OpenID Connect (OIDC) as the authentication protocol while still utilizing OAuth2 for the authorization protocol.

Mines has their Azure portal open so that any Mines student can create an application and utilize the organizations Tenet ID, meaning that anyone with a valid Mines account can sign-in and use an instance. The Azure application by default provides username, name, and email, in the Java Web Token (JWT) when signing in through OIDC which is all the information we need. Therefore, no extended tokens need to be sent to pull any unnecessary information.

We have also decided to pivot and that the sign-in authentication/authorization page is only needed for Teachers/TAs to "open" a desired space. This was decided because we are transitioning to Blastercard/CWID check-in for students, so there is no need for them to sign-in with their Mines account. However, to ensure that it was a Teacher/TA who signed in initially, we are still using the previous process of querying Canvas (see Figure 5) to view what role they have within the class. If a teacher or TA is returned, they are successfully signed in.

The roles are defined as:

- Teacher (Signed-in) -> Level 2 (Full access, including system config)
- TA (Signed-in) -> Level 1 (Access to all checked-in student data, including updating quiz scores)
- Guest (Signed-out) -> Level 0

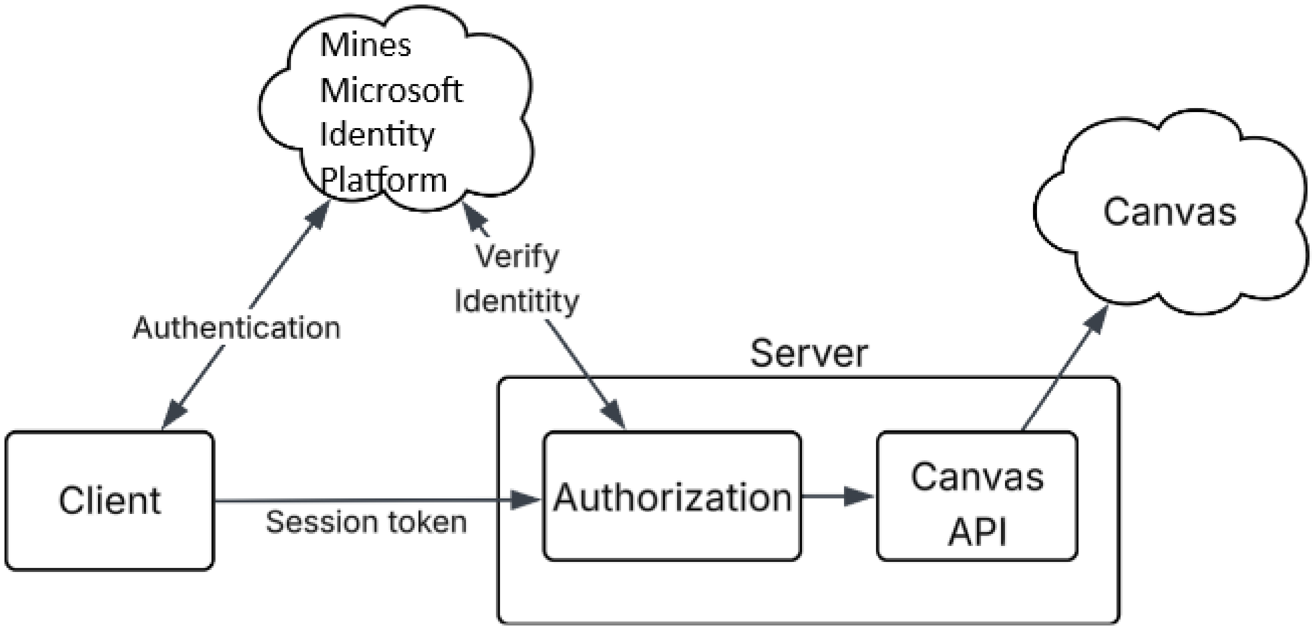


Figure 5: Authentication Process Flowchart

Using this system in place of Okta provides a simpler process while still providing the same end goal. Microsoft Azure only provides basic information when compared to Okta, however the basic information it does provide is the only needed information, and there is less overhead when it comes to ITS because we are only pulling the basic information.

VII. Software Test and Quality

Activity	**Rationale (How it will ensure code quality)**
PR Review	We are having at least one other person review our code before merging a PR. This ensures that code is readable, as well as written and documented in a way that follows best practices and is clean. Furthermore, these PRs are ensuring that a feature is fully completed prior to being merged into production. We have often discussed the idea of "done" software vs "done done" software. If the PR does not showcase "done done" software where the feature is as fully developed as we intend it to be, then whatever that feature is missing must be added before the PR is approved and merged. This may include: • Missing or lackluster documentation • Missing or insufficient unit testing • Being closed to future development • Inconsistent or no style guide
Unit testing	We are writing unit tests for our code to ensure that when we make a new feature, it works as intended, and as we add more features, the prior ones continue to work. These unit tests are intended to not only test the entire feature, but also the components of said feature. This is necessary to ensure that when other software or other features of our software interact with newly developed features, the information passed between features is consistent and can allow classes to interact with each other. These unit tests better ensure certain sub-components of larger features continue to work, and if a smaller sub-component unit test fails, then there is less to look at and fix. Additionally, by ensuring that new features are open to future development through unit tests, our application will be easier to iterate on should a new feature be necessary.
Integration testing	We are writing tests to make sure that new features work the way we intend them to in the greater system, and that no new features break the system. Along with that, we also have tests for our integrations with other systems. That way as new features are implemented, or API services change, we can test our software to ensure that API's can still be called and the expected result is returned. This further includes hardware testing (specifically RFID scanner integration).
User testing	Any time that we make changes or add features that alter the frontend or display, we have brought the application to one or more end users to ensure that it still looks and feels comfortable to use.

	This testing is necessary to not only ensure that our project remains easily usable for existing users, but also to minimize friction when onboarding new users (i.e. freshmen design students). Additionally, this user testing has gradually increased both accessibility and visual appeal in our design as well. User tests also help ensure that our front-facing UI/ UX components can properly handle user-input and correctly pass user-input to the proper backend channels.

VIII. Project Ethical Considerations

The most pertinent ethical considerations for the innovation hub project are the ones that have to do with non-discrimination and privacy. This includes sections 1.4, 1.6, and 2.9 of the ACM Code of Ethics, and the sections underneath part two of the IEEE code of ethics. This is because the project is for a part of an educational institution, which is held to stringent laws regarding non-discrimination and how PII is handled, including Title IX and FERPA. Since Labriola cannot deploy the project unless they are in compliance of these laws, we have ensured that none of our code is in violation of it; however, since there is still a need to handle student information (such as to map an identifier to a student to pull their grades from Canvas), sections 1.6, 1.7, and 2.9 of ACM's ethics code are in most danger of being violated. If we cannot adhere to these ethical principles, the project itself may not be able to be deployed. Mines, and Labriola by extension could be fined for any FERPA violations, and of course, they could face the reason why FERPA exists in the first place: unauthorized individuals could gather student information if it isn't properly secure, which is detrimental to a student's privacy. Students have reasonable expectations that an educational institution can and will keep their PII secure.

In addition to the above ethical principles, we also strive to follow sections 2.2 and 2.3 of ACM's ethics code and sections four and five of IEEE's ethics code. These sections all have to do with acting in a professional, competent, and lawful manner, and we should hope to ensure we are acting in a way that would reflect ethical conduct.

In applying for Michael Davis's Reversibility Test, we believe our project passes. It is easy to imagine ourselves as an end user of our product, as our team consists of students who may access the shops and spaces of Labriola, and a TA who is required to use this technology when they are working. Thus, we are already end users of the project, and should our project have a negative impact due to ethical violations, we ourselves would feel it. In applying for the Legality Test, we choose to steer clear of choices that would violate any existing privacy laws. In particular, we've already had to make the decision to avoid any routes that would violate FERPA or introduce shadow IT technology, despite the alternatives requiring much more bureaucracy and introducing constant roadblocks for us. However, legally speaking, violating FERPA is not an ethical route to take.

In the case that our software quality plan is not robust enough or implemented improperly, the project faces the potential to introduce ethical violations from messy, insecure, or unmaintainable code. If the project is messy, it will not consider the user experience and may not be to student satisfaction. If the project is insecure, as mentioned previously, we face legal implications. If the project is unmaintainable, the project will not meet what the client is seeking and will not be viable for the long-term. All of these would imply that we violated working in a professional and competent manner for this project.

We have many considerations to take regarding security with this project; the biggest being that student information should be handled in a way that respects privacy and confidentiality. As mentioned before, the project is subject to FERPA, as we handle student names, emails, and CWIDs. This means that there should be sufficient authentication and authorization so that not just anyone can see and edit other data. As a result, we have hardend any sources of persistent data, such as our database. We will also need our code to be up to ITS's standards and are attempting to working closely with them to ensure this.

IX. Project Completion Status

Our MVP has been fully completed. As explained in our functional requirements, the minimum we deemed necessary for the project to be considered successful is the following:

- A transition away from using Mines sign-on for students & workspace users to Blastercard tap-scans for sign-on. This has been completed after the better part of a semester of project pitching and discussion with various groups on campus (ITS, Blastercard Office, Registrar's Office, and most recently TRAIL). As the current implementation is supported underneath OreConnect and thereby TRAIL, their application is being utilized as a middleman for students checking in and our application's backend. This is now complete.
- Replacing the old CSV file used as a database with a proper SQLite database to protect against data corruption and improve security. This has been fully implemented with testing, reviewed by team members who did not work on the feature, and pushed to production. This upgrade has been integrated into production for approximately six weeks and has not caused any problems.
- Implement SSO for TAs and other faculty members via Microsoft Azure. This has been completed and has been tested on mockup Canvas courses. This feature has been tested using a temporary Azure instance. The code is complete and is included in final delivery; however, ITS has not yet approved an Azure instance for Labriola's use. As a result, we are using a temporary year-long key until Labriola can work through the entire software review process successfully.
- Allow TAs and other faculty members to update Canvas quiz scores directly from our application without having to use Canvas. This has been implemented and successfully peer reviewed. There are a number of other Canvas-related changes and UI updates that have been wrapped up in this feature as well. This is considered completed and has been included in our final delivery.
- Badge printing is available for students when they scan into a space. This feature has been tested and currently can show the PDF that would be printed. While there is no hardware for actually printing the badges at this time, the implementation is there for when it becomes available in the future.
- Moving the application onto the cloud is no longer in scope for our team. We were told by ITS that based on the current state of the application and what our team was going to do with it, a software review would almost certainly return with a rejection. This is a goal that should exist as a very long-term if desired, although we now recommend moving to a centralized Raspberry Pi server for current future use.

Our testing has taken on a tiered approach for each new feature where, depending on the scope of changes being implemented, more strict testing and validation may be required. For features that have a narrower scope (i.e. updating the application's UI), primarily unit & integration testing is involved to ensure that the feature both does what we expect it to do and does not impact the flow of the rest of our application. For the UI specifically, this has been done primarily via user testing with ourselves and our clients at Labriola to develop an interface that is simultaneously more intuitive for users and does not complicate the workflow of our application. For features with a much broader scope (i.e. Blastercard tap-scans), we are intending to perform system testing as well. This is due to interfacing new hardware (RFID scanners) as well as third-party services (primarily OreConnect & TRAIL) which both must interact with our application for it to work.

X. Future Work

There exist a variety of other features which have been requested by our clients that need further discussion and, while not realistic for the scope of the project this semester, would be compelling for a future semester's Field Session. These are listed below in anticipation of a Summer 2026 Field Session group continuing to work with Labriola and, hopefully, finalize the application. Loosely in order of importance, they are:

- Migrate the Raspberry Pis to a centralized server such that the application's data can be stored centrally. This has additional implications for data science purposes in the future regarding the busyness of workspaces

- Allow for trainings to expire within Canvas such that users who have not been in or been trained in a workspace recently (6 – 12 months) must be retrained
- Enable users signed into a workspace to be clickable on the workstation, popping up a display of all their trainings for the TA to see after their sign in

Additionally, with the transition to using OreConnect for Blastercard sign-ins, there exists the possibility to transition much of the current workload done on Canvas to OreConnect such that the application is more self-contained and would no longer require a Canvas API token. This would involve a number of changes listed (non-exhaustively) below:

- Converting all Canvas quizzes to Microsoft or Google Form quizzes accessible through OreConnect
- Creating badges and/or tags to track user statuses and generic shop permissions. This could extend to machine access as well, but would likely be too cumbersome an approach to use
- Migrating the existing Canvas quiz scripts to OreConnect and developing new scripts to handle student graduations and updating permissions as they expire
- Training faculty, TAs, and introducing students to OreConnect broadly as it is a system that most individuals at Mines are far less familiar with than Canvas

XI. Lessons Learned

The primary lesson our team has taken away from the project thus far has been an appreciation for and reminder of how time consuming and oftentimes cumbersome integration with multiple applications can be. For the Blastercard tap-scan sign-ons in particular, the initial expectation was that this task would have roughly three stages: getting access to Mines ITS's internal database mapping Blastercard ProxIDs to CWIDs; writing the code to take a ProxID from the scan, pass it to ITS's database, and receive the corresponding CWID back; and finally testing to ensure both the hardware and integration was successful. While this predicted workflow was accurate, the amount of time it has taken to receive access to ITS's database was significantly longer than expected, involving more meetings and pitches than any of us had expected. Even as the feature is finally being implemented, we have never received direct access to ITS's database and are instead using OreConnect as an intermediary in this process because they were willing to work with our project.

Another lesson we are continuing to learn during our development is the importance of data security and storage. The summer team which had worked with Labriola previously had used a csv file for storing basically all of the important information for this application which, while effective on paper, is prone to corruption. We have migrated this to a SQLite database to minimize said corruption risk, but there remains security concerns pertaining to the information we are storing and how easy it would be to hack.

One last lesson we have all learned at various levels is how to create and simulate testing environments when there does not exist a readily available testing method. This pertains primarily to integration and system testing since we cannot easily modify Labriola's Canvas course page without impacting all of the students and faculty that use the page daily. As a result, we have had to make mock courses for testing, using some of our Canvas profiles as faculty and others as students to ensure that the appropriate information is available to each role prior to pushing our changes to the real Labriola Canvas page.

XII. Acknowledgments

Our team has benefited greatly from the support of a few key individuals & organizations on Campus that deserve special recognition. These include:

- Our clients at Labriola, Julia Roos and Victoria Bill who have been a consistent part of the development process, ensuring that we are able to stay in touch and deliver a product that aligns as closely to their vision as possible. They have helped us on many an occasion to get in touch with other individuals listed below who were able to share insights necessary to the completion of this project.
- Susan Gieg and by extension TRAIL for supporting the project and offering OreConnect as a middleman to support Blastercard sign-in when ITS proved to no longer be a possible development avenue regarding mapping

ProxIDs. Ben would like to offer his personal thanks for answering many questions and offering multiple demos for how to use OreConnect for what we were interested in doing on short notice, ultimately leading to a subroutine which may hopefully be used by other student groups on campus

XIII. Team Profile

	Bethany is originally from the Chicago area and is a senior in computer science with a specialty in computer engineering. She has interned as a software developer at Parsons and is looking forward to becoming a full-time developer after graduating this May.
	Ben is originally from Texas before growing up just south of the Golden area and is currently a senior majoring in computer science with an emphasis on data science. He has worked with the entry-level course CSCI128 as a TA before taking on additional responsibilities involving course development and maintenance for the past 2 years.
	Tristen is originally from California and is a senior in computer science with a specialty in computer engineering. Tristen has previously worked as a DevOps Engineer at Parsons and as a Cyber Engineer at ICR. Tristen is currently the President of OreSec which is the Mines Cybersecurity Club and has competed in numerous cybersecurity competitions with the club over the last 3 years and plans to continue to do so for his 4th and last year at Mines. Once Tristen graduates in May from Colorado School fo Mines, he is looking forward to working in the cybersecurity field.
	Alex is from Fort Collins Colorado and is a senior studying Computer Science specializing in computer engineering, likely to enter the energy industry post-graduation.

Appendix A – Key Terms

Include descriptions of technical terms, abbreviations and acronyms

Term	Definition
PII	*Personal Identifying Information, or information that can be used to map a user to who that person is, such as name, CWID, government IDs, etc*
FERPA	*Federal Education Rights and Privacy Act, or a US law that protects student and their right to privacy regardless their educational data, such as grades and PII*
Labriola shops and spaces	*All areas at the Labriola Innovation Complex intended for students to build and innovate that are managed by Labriola Innovation Hub. These areas are staffed by TAs and/or managers and have machinery that require trainings to operate. These areas are currently made up of the woodshop, machine/metal shop, makerspace, electronics shop, and paint & composites shop.*
ProxID	*Proximity ID; acts as the identification data on a Blastercard. The ProxID must be used in conjunction with Mines ITS's database to map the ProxID to the CWID, name, or email associated with the Blastercard's owner.*