Senior Design Final Datasheet & Project Report

Title	MakerSpace Badging System
Author Names	Julie Darwin SE, Joseph Kale DE, Daniel Park DE, Andrew Sellers DE, Ian Wallace DE, Zach Winters PM
Submission Date	4/29/16

Summary

- Describe what this product does- aim this description at the end user.
- Provide background information assuming the user does not know the context this product is used in.
- Describe briefly all inputs and outputs that are relevant to the end user (not a technical description)

This "Smart Badging System" monitors usage and controls access of various Maker Space resources, including 3D printers, a Vacuum Former, Industrial Chiller, Laser Cutter, Vinyl Cutter, and others in the Maker-E Maker Space. In addition to the installation of this system in the Maker-E Maker Space, we will provide the necessary instructions for all users and parts list for the future installation of this system in the 3 other maker spaces on campus. For the non-makers reading this, the equipment listed above requires certain levels of training, and the current system in place involves paper sign up sheets and IDs. Because we live in the 21st century, we aspire for something more advanced and our "MakerSpace Badging System" provides that. With the completion of this project, Bucknell's makers will all have their own RFID badge that can be scanned at each Maker Space resource, either granting them access to use the resource or allowing them to request the necessary training. Our beautiful user web interface allows both makers and admin alike to access different features. A "Smart Tech Box" equipped at each resource will have all the capabilities of scanning Maker IDs, controlling power to the resources, and connecting to the database that contains all pertinent information. While makers can submit training requests, search users and resources, and view recent projects, admins have all the same features with the added ability to view usage logs, add makers or resources, and even add student techs that work at the maker spaces. With the "MakerSpace Badging System" makers can now express their maker spirit with ease!

Instructions

The report should be in a standard font and no smaller than 10 point. **DO NOT REMOVE TEXT OR BULLETS FROM UNDER SECTION HEADINGS!**

Part A: Technical Datasheet

Description

- Describe what this product does for a technical user short on time (be brief). This should be based on your problem statement.
- Write a bulleted list of the products features
- Provide enough background information to put the project in a technical context.
- Describe briefly inputs and outputs of the product (a technical description but not tables of data).
- Provide a level 0 functional block diagram. This block diagram should be a physical layout of the product showing the physical location of connections to the device.

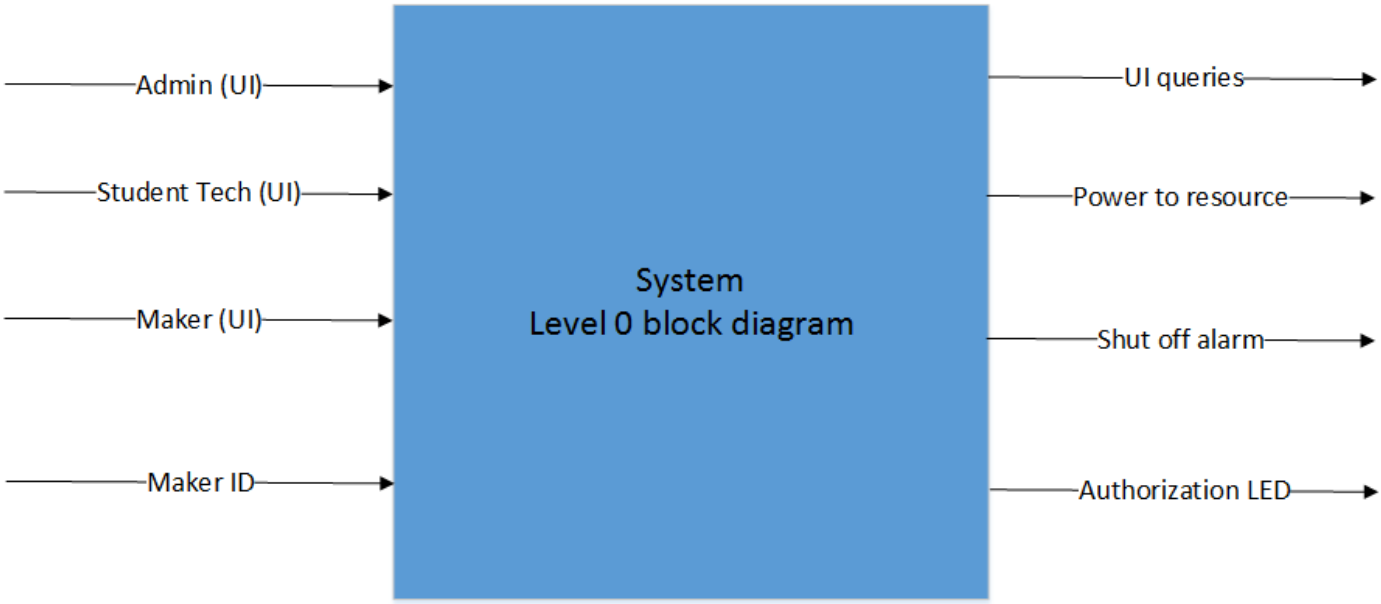
The purpose of the "MakerSpace Badging System" is to replace an out of date login/monitoring system that is currently in place in the maker spaces on campus here at Bucknell. With the resources of a group of talented electrical and computer engineers, as well as the technology available to us in the 21st century, we have developed a project that replaces the paper based system currently in place with a more effective and more efficient way to track who is allowed to use what and when something is being used. Our system, with a MySQL database acting as the backbone, stores pertinent information that promotes safety and security within the maker spaces on campus. RFID scanners equipped at each piece of equipment (resource), connected through a Raspberry Pi which is connected either via Wi-Fi or Ethernet to our database will either allow or restrict use to the desired resource. The method of restriction is based on the restriction of power to the resources through the use of a solid state relay. The Raspberry Pi II acts as the brains for the solid state relay as it checks whether or not a maker is authorized which is contained in the database. The RPi recognizes the the maker's RFID card (UID) and references the database as to whether the scanned UID is authorized for the resource attempting to be accessed. If the maker is authorized, power to the resource will flow through the SSR, allowing use. The UID must remain on the scanner in order for the resource to remain operable. If the UID were to be removed, intentionally or accidentally, a student tech's ID can be placed on the scanner to continue use of the resource. If no ID is placed after a predetermined period of time, power to the resource will be cut off.

Along with this RPi/MySQL database based scanning system, we provide a web based user interface system with several functions. At the main page, found at www.eg.bucknell.edu/makerspace, a beautiful WordPress developed website will give the user an option to log on as a maker or an admin. As a maker, the user is given the option to request training for any resource type, which will notify an admin, search users, search which resources they are authorized to use, and search resources throughout the maker spaces on campus. As an admin, all these functions are also available, as well as the ability to add new resources, add new makers, and view resource usage. There are two levels of admin capabilities. Student techs who work at the maker spaces have all of the previously listed capabilities, while maker space admins have these capabilities plus the ability to add student techs. The website also provides links, available to anyone, to separate sites that display maker profiles, the maker spaces on campus, and the recent projects that Bucknell's makers have worked on (courtesy of Bucknell's Maker Space Instagram page.)

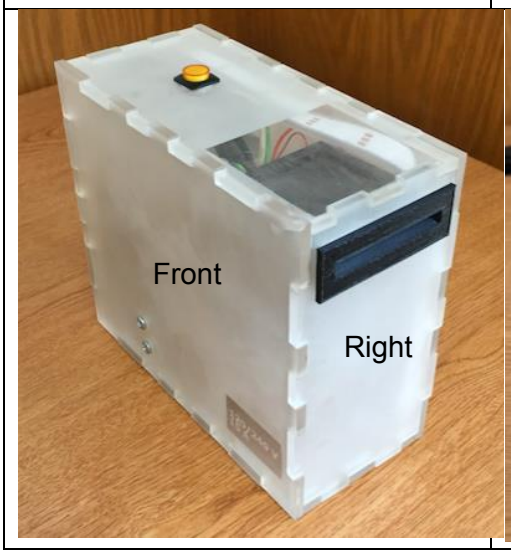
A list of the "MakerSpace Badging System's" features can be seen below:

- A simplified, more technology fulfilled method for students to become registered makers
- An easier method for makers to use resources they are authorized to use and to request training for resources they are not authorized to use
- A way to log and monitor the usage of the maker spaces on campus, organized by resource type, location, etc.
- A more secure system that allows around the clock use of maker spaces on campus
- A more secure way to prevent unauthorized makers from using equipment they are not authorized to use
- An easier method for makers to request training for resources
- A more efficient way for admins and student techs to see makers who would like to be trained and administer training
- A user interface that eradicates the need for paper badge and paper log in systems
- A better maker experience

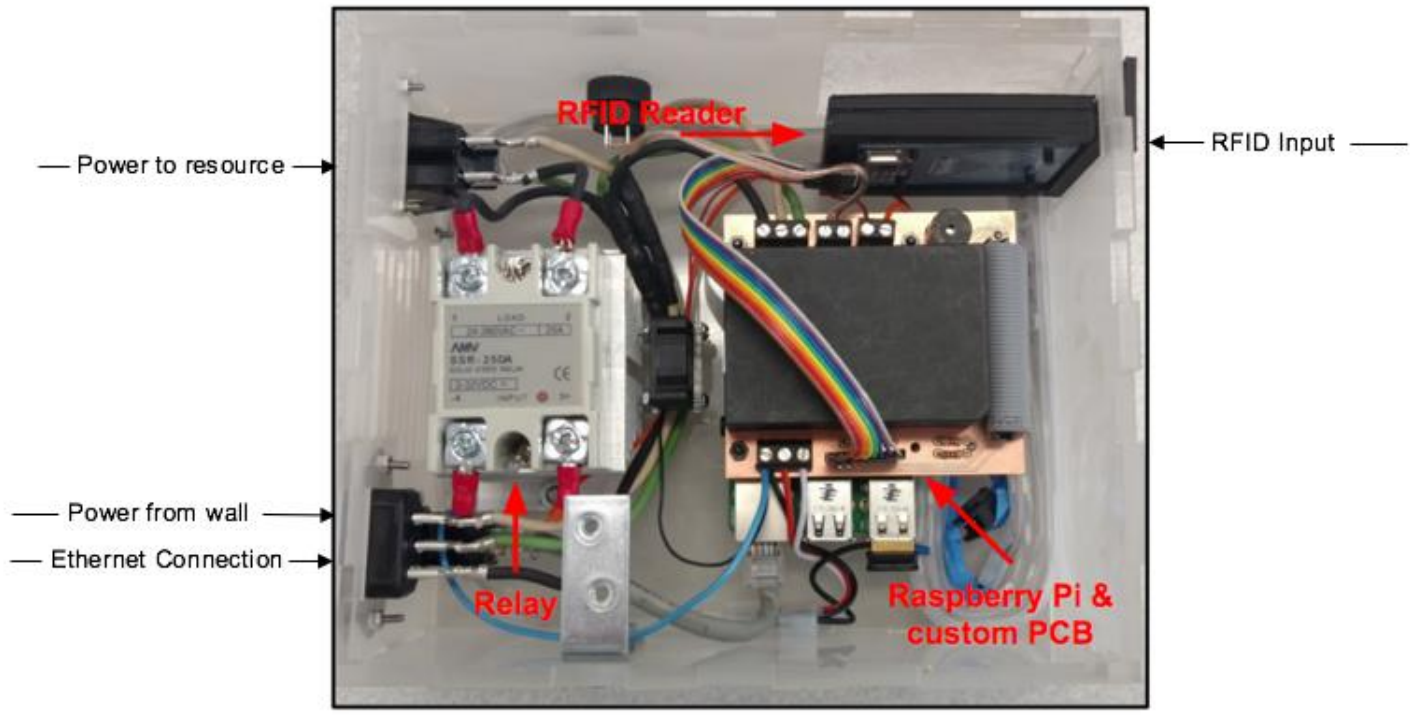
Level 0 block diagram of the system:



Views of the "Smart Tech Box"

Isometric View	Right Side View	Left Side View
		

Layout of the "Smart Tech Box" with inputs labelled (Front Side View):

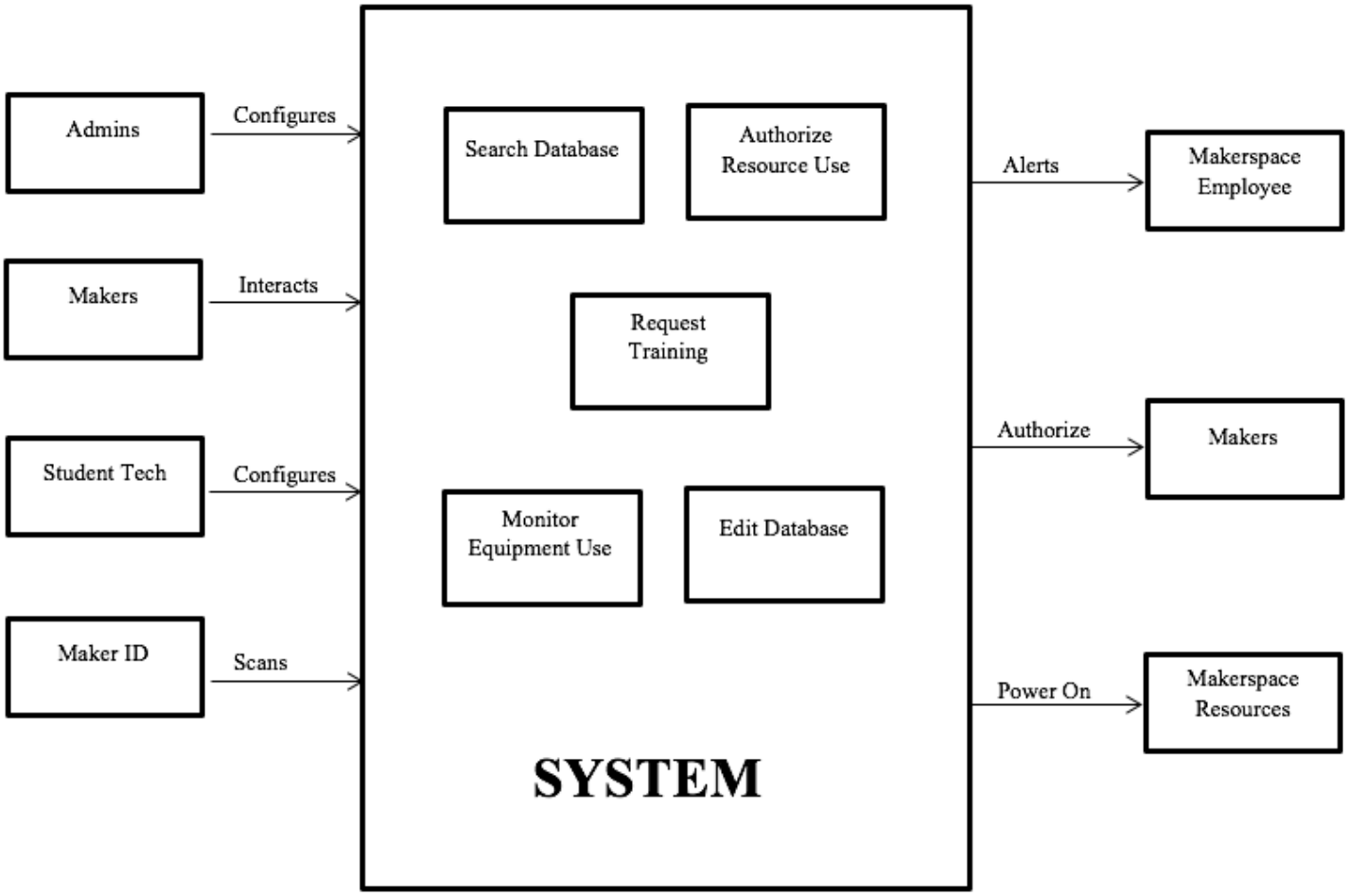


Product Block Diagram and pin out diagram

- On one sheet of paper provide a level 1 functional block diagram of your product.
- On a separate sheet of paper provide a table and/or diagrams of the inputs and outputs of the project as shown on your technical block diagram.
    - ONLY INCLUDE INPUTS AND OUTPUTS OF THE SYSTEM FROM THE TECHNICAL BLOCK DIAGRAM.
    - DO NOT INCLUDE INTERNAL CONNECTIONS/FUNCTIONS THAT ARE NOT ACCESSIBLE TO THE USER.
    - If your device has visual/tactile/audible user interfaces (such as displays, sounds, buttons) include these in the table.
    - Make sure you include detailed diagrams of any connectors with pin numbers.
    - The pin numbers and connector designations should match the Reference Designators from your layout and also the terminology used in your level 1 block diagram.
    - If you are doing a project that has a significant high-level (e.g. C or Java) software component, include a table of public functions.

NOTE: No text on any drawing or schematic can be less than 10 point font, and the minimum component dimension on any drawing is ¼" (6 mm). Your block diagram must identify each part of your system and who was responsible for that part. Your block diagram may be on a separate page if required. If needed you are encouraged to insert a page larger than the standard letter size. To do this: 1) *Insert* → *Break* → *Page Break*. 2) *File* → *Page Setup*. In the dialog box select the *Paper* tab and set the size to custom. Then select *From this point forward*, from the *Apply to:* pull-down menu. 3) Select the paper size you need for your drawing. Sizes up to 22"×22" are supported. 4) Insert another page break. Follow the instructions in step 2 to set the rest of the document back to letter size (from the new page break forward)

Level 1 functional block diagram:

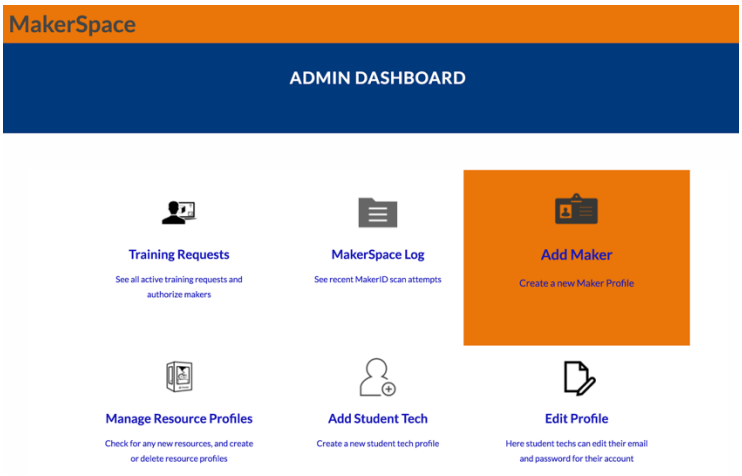
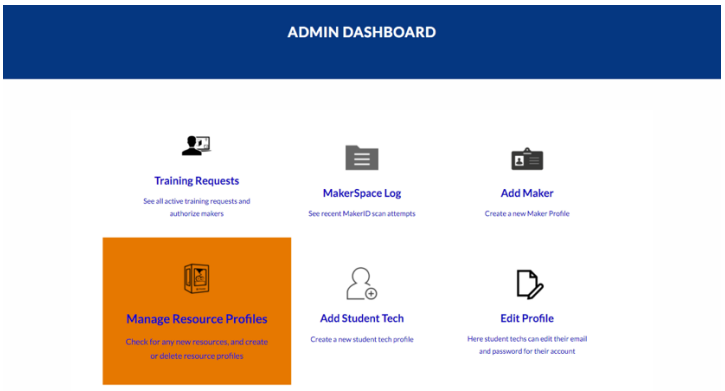


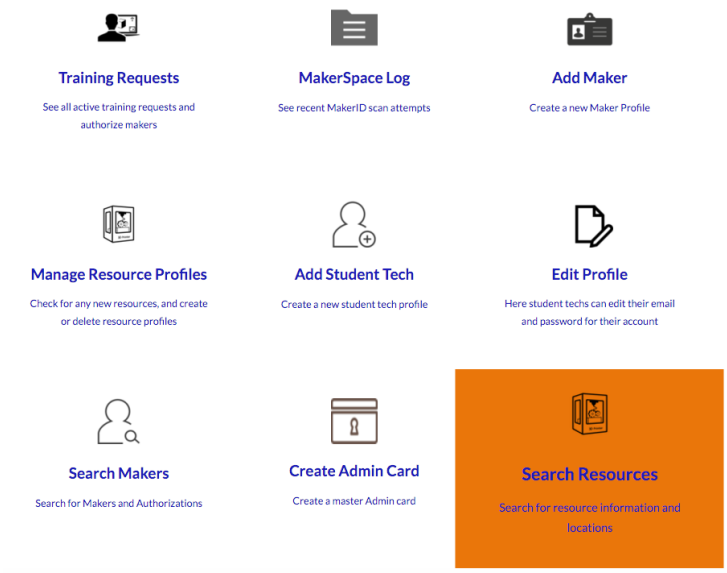
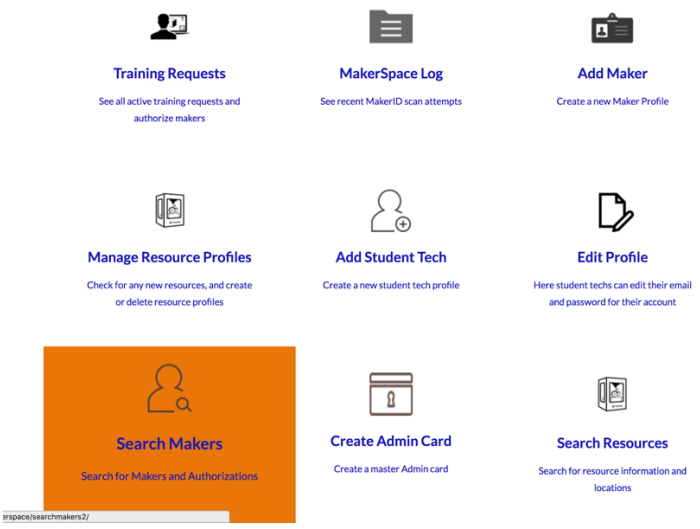
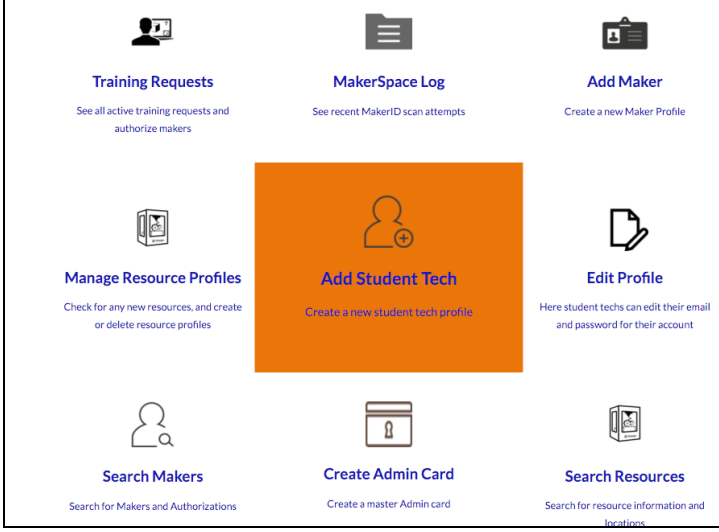
Technical Block Diagram Inputs:

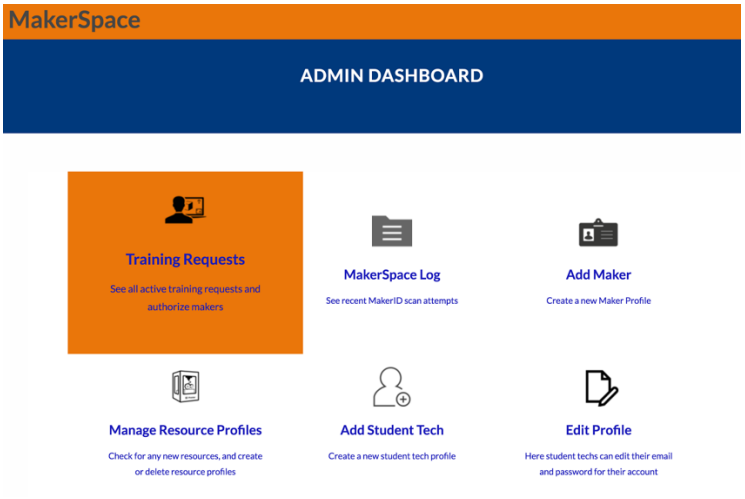
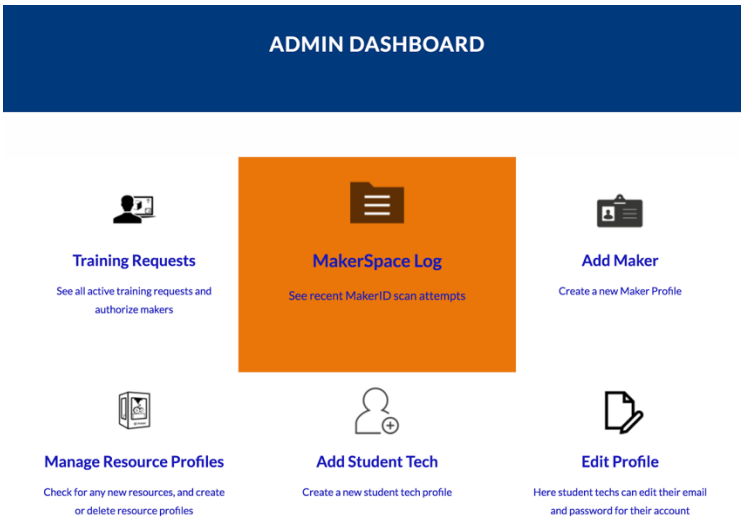
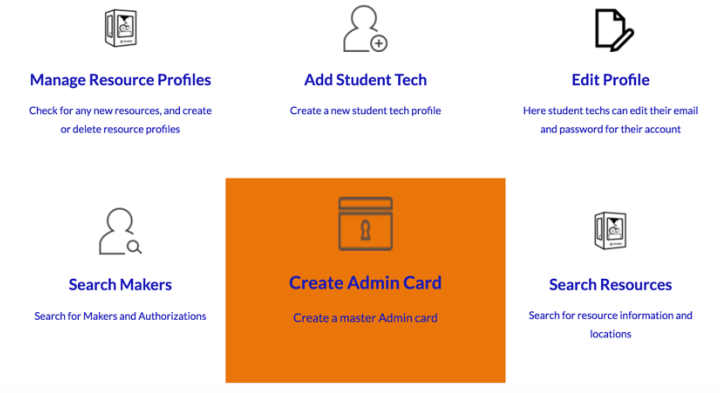
Smart Tech Box Inputs:

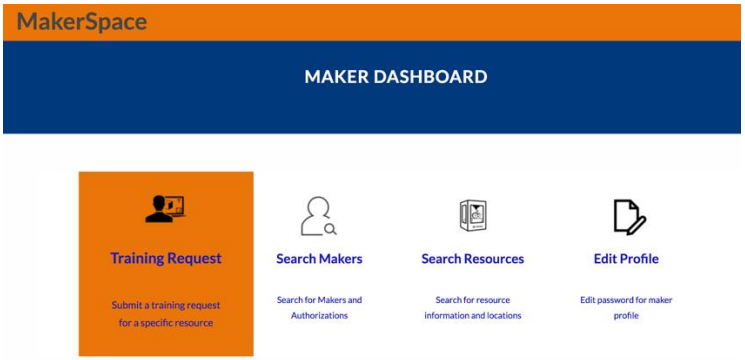
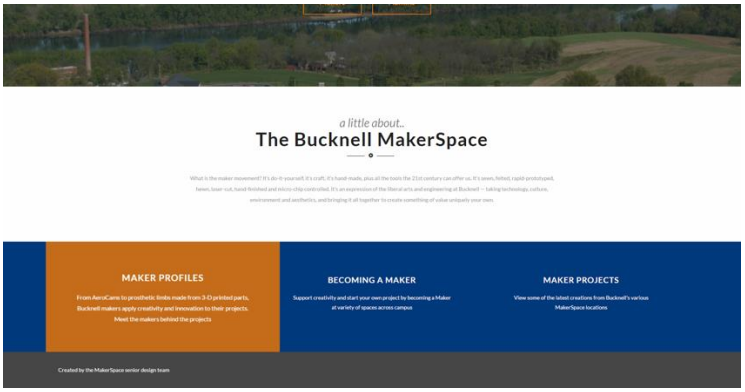
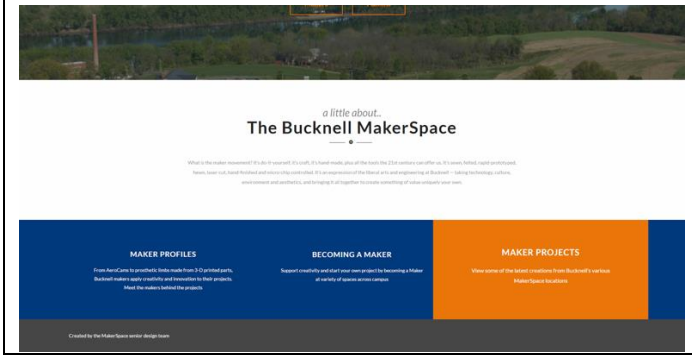
Input	Description
RFID Card	Each Maker ID is an RFID card with a unique UID
120VAC Power	Power is routed through the Smart Tech Box to the Resource
Cat 5e Ethernet	Connection to the Database can use either Ethernet or Wifi
Wifi	Connection to the Database can use either Ethernet or Wifi
Button Press	Pressing the button on the top of the box allows the user to “log off” from using the resource

Web App Inputs:

Input	Description
Add Maker 	When an admin logs in through the Web App, they can add a maker by entering the maker’s information
Create New Resource/Delete a Resource 	When initially setting up a new resource in this system, an admin must assign the resource, which is located by the Mac Address on the RPi in the “Smart Tech Box” When a resource is no longer in use at a maker space, an admin can remove it from the database pool of resources

Search Resources 	An admin or maker can search for resources either by type or location
Search Makers 	This allows an admin to search for individual makers and see what they are authorized to use and if their profile is private. A maker can search other makers as well, but private makers will not be displayed
Add a Student Tech 	An admin has the authority to add a student tech who will have all the abilities of an admin, with the exception of adding additional student techs

Training and Approving Makers 	An admin or a student tech can grant resource access to makers who have completed training
Using the MakerSpace Log 	An admin can search usage for any resources, which is separated into currently being used, previous successful attempts, and previous failed attempts
Create a New Admin Card 	Admin can make addition admin cards, which can be used when a maker would like to use more than one resource at a time

Become a New Maker 	Any student can select the link titled "Becoming a Maker" which will direct them on how to become a maker
Request Training 	A maker who is not authorized to use a resource can request training for a resource type, which will notify student techs that the maker would like to receive training
Access "Maker Profiles" Link 	This redirects Web App users to Bucknell's Maker Profile Page
Access "Maker Projects" Link 	This link will take Web App users to a separate page that will pull and display photos from the Maker Space Instagram page

Technical Block Diagram Outputs:

Smart Tech Box Outputs:

Output	**Description**
Power to the Resource	When a maker is authorized, power will flow to the resource plugged into the "Smart Tech Box"
LED Lighting	LED lights illuminate the box green when authorized by a maker, yellow when authorized by an admin card, and flashing red when a maker is not authorized
Buzzer	When an authorized card is removed from the "Smart Tech Box" the buzzer sounds, warning the maker that the resource will power down if the card is not replace

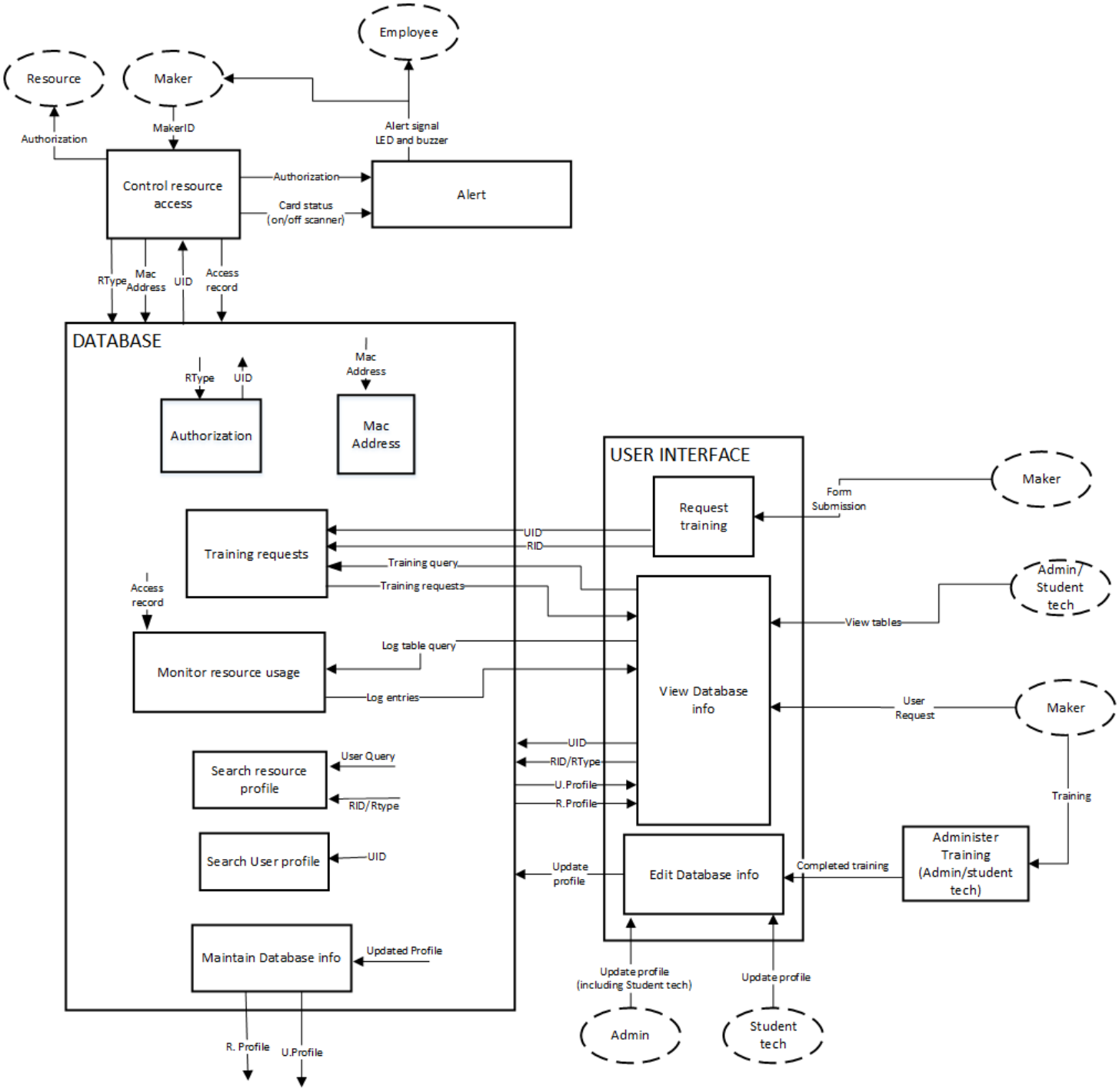
Web App Outputs:

Output	**Description**
Web App User Interface 	The WordPress developed user interface will display buttons for each Web App input and will redirect users to the appropriate page for each action

Detailed Technical Description

- Provide your CSD in readable form (see above) to show the user how it works.
- Describe the principles of operation of your device in detail.
- Provide references to any other research or data a technical user would need to understand the operation of your device.

The Conceptual Solution Description (CSD) for the MakerSpace Badging System:



As shown by the Conceptual Solution Description above, there are two input ends of the system: a scan of a User ID card (UID) and a user interface through the WordPress webapp. In the former, the only real input is a scan of the User ID (UID). The number stored on the UID, which is inherently stored within the RFID card that serves as the UID, will then be recognized by the "control resource access" block. This block passes the UID to the database, which contains information

on which UIDs are trained for which resources. Each resource is known by the database as a resource type, which is distinguished by its Resource ID (RID). The resource ID is paired in the database by the Mac Address associated with the Raspberry Pi connected to the resource. If the UID has received training, the "control resource access" block will authorize use of the resource. If not, the user is not authorized and the device will not power up.

At this point we can shift our focus to the other input side, the WordPress UI. On this end, there is a wider variety of functionality available to the users. The users in this case can be either makers or admin/student techs. The main functions shown in the CSD are training request submission, access to view usage information and user information stored in the database, and granting makers authorization to resources. The CSD outlines who, whether maker or admin/tech, can do what.

The outputs of the system are somewhat self-explanatory after understanding the inputs. On the "control resource access" end, outputs consist of power, or no power, to the resource, as well as an multicolored LED that signifies whether or not the user is authorized and a buzzer that sounds when an authorized user removes their UID while the resource is in use. On the UI side, the output is obviously a webapp. Within this webapp, the outputs include admin logins, training submission forms, and usage/user information.

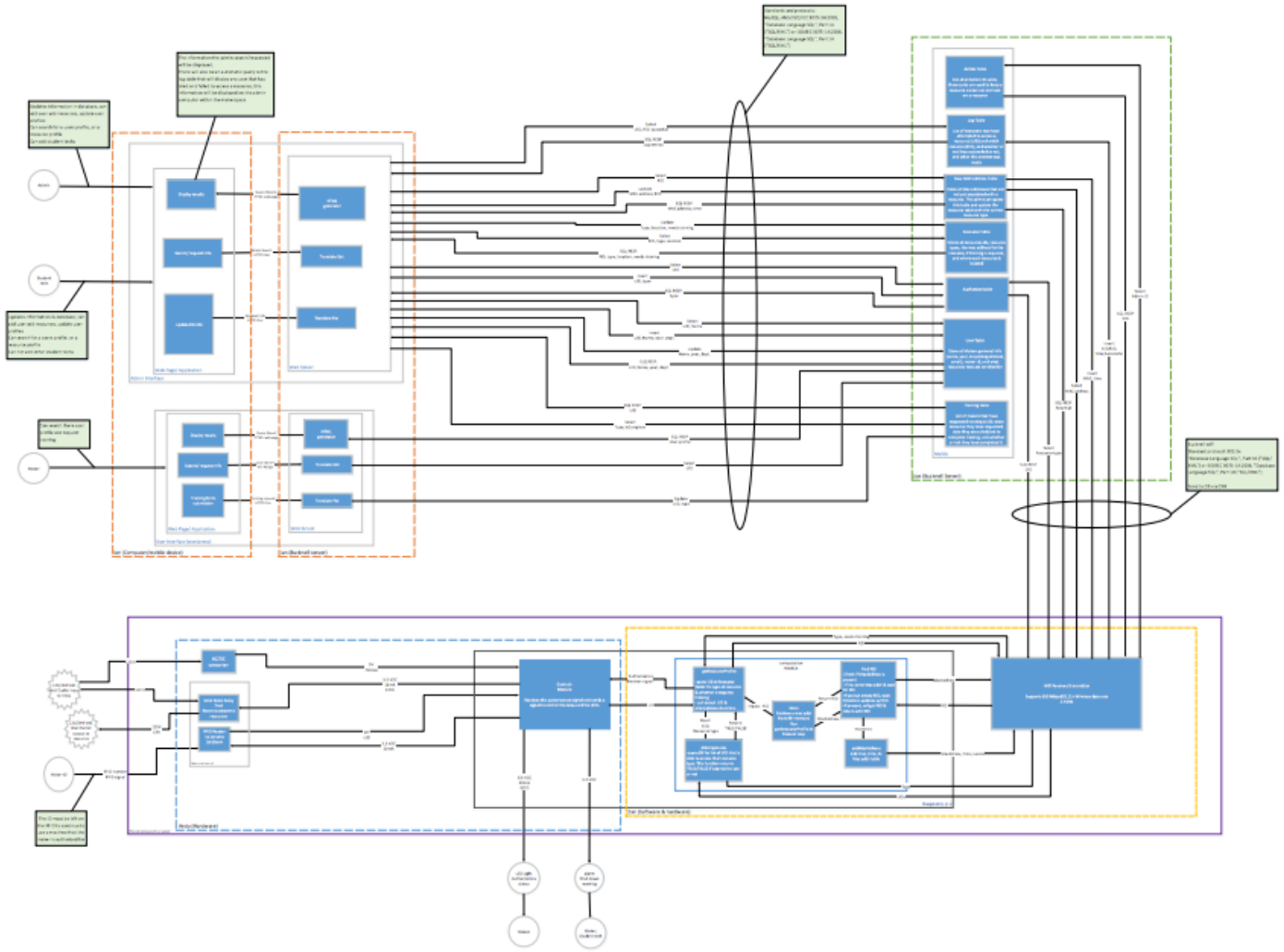
Within the system, there are several inputs and outputs that are critical for operation. Important communication includes the passing of UIDs and resource types between the database and the resource control. The database and the webapp are very heavily integrated, passing most pertinent information along between the two, allowing users to view what is stored within the database.

Tables of Detailed Technical Specifications Explaining your Design

Describe the measurements, models, drawings, simulations, etc. that your team used to understand and predict the performance of your design project. You should write at least one to three paragraphs on each of the points below:

- Provide a brief description of the signals/data at all inputs, outputs and internal test points. Include units where applicable.
- List in tables measured values of all input and output pins/calls to the device during normal operation.
- Provide appropriate graphs or figures of any:
    - Measured time dependent inputs, outputs, and signals at all test points. For high level software components describe data passed (i.e. between functions, databases, I/O devices, etc.) and provide examples of actual data.
    - Measured change in output with changing input parameters (i.e. variation of gain with operating voltage for an amplifier)
    - Considering all components used, list the Absolute minimum and maximum operating ranges.
- Briefly describe the test procedures used to obtain the above values.

The Overall Technical Block Diagram for the MakerSpace Badging System:



In the following sections, a breakdown on each individual block will be shown in the order as follows:

- Resource Access Control Block (1)
- WiFi Receiver/Transmitter Block (2)
- WordPress Webapp Block (3)
- MySQL Database Block (4)

Resource Access Control Block (1):

Overview

The unit tests for my block were split into three major sections; RFID reading, Relay operation, and the Control Module primarily on the software end. Testing is both for hardware and software with both minimal and maximum integration.

Test 1 for Relay Block (SSR)

Rationale and description: Test that the solid state relay (SSR) can be adequately actuated using GPIO voltages from the RaspberryPi. This needs to be tested to ensure that the SSR can be operated using the GPIO of the RaspberryPi directly and does not need an additional control circuit. This test replicates the full range of voltages that can be applied by the RaspberryPi and continues outside of the range to ensure behavior is consistent across the range of voltages that can feasibly be output by the RaspberryPi. It also accounts for possible fluctuation in turn-on and turn-off characteristics by both increasing and decreasing the input voltage across the entire range. Increments of 0.1V provide a rather clear picture of device functionality. GPIO voltages were simulated using a lab power supply. Output voltages were measured using a lab multimeter set to measure AC voltage. I performed this test to ensure that I would be able to operate the relay using the RaspberryPi and would not need to design/purchase a new control circuit as an interface between the RaspberryPi and

the relay. I chose the pass conditions to be within 2% of the nominal wall voltage because this level of fluctuation is well within the tolerance of the resources at the 7th Street Makerspace and would require a significant dip in wall voltage well outside of its typical range.

Test	Desired Level	Actual Level	Comments	Pass/Fail
Apply a simulated GPIO voltage to the relay block and measure the output to ensure the RaspberryPi is capable of activating the relay. GPIO voltage is simulated using a lab power supply increased in increments of 0.1V. The input voltage is increased from 0V to 5V and voltage measurements are taken using a lab multimeter. The power side of the relay is supplied 120V at 60Hz and the load side is connected to the lab multimeter.	< 2%	~ 0.53%	Lower drop in voltage than expected. I was unable to test this outside of a lab environment however, so the range of wall voltages I had access to were limited. We may need to repeat this test once Dan and I integrate. The turn-on and turn-off voltages are easily within the range of the RPi.	P
Repeat the test with input voltage decreasing from 5V to 0V.	< 2%	~ 0.53%	The voltage levels measured were almost identical to the previous test.	P

Test 2 for Relay and Control Module

Rationale and description: Test that the RaspberryPi can source adequate current to actuate the solid state relay (SSR) using GPIO. This needs to be tested to ensure that the SSR can be operated using the GPIO of the RaspberryPi directly and does not need an additional control circuit. This test replicates the full range of voltages that can be applied by the RaspberryPi and continues outside of the range to ensure behavior is consistent across the range of voltages that can feasibly be output by the RaspberryPi. It also accounts for possible fluctuation in turn-on and turn-off characteristics by both increasing and decreasing the input voltage across the entire range. Increments of 0.1V provide a rather clear picture of device functionality. GPIO voltages were simulated using a lab power supply. Output voltages were measured using a lab multimeter set to measure DC current. I performed this test to ensure that I would be able to operate the relay using the RaspberryPi and would not need to design/purchase a new control circuit as an interface between the RaspberryPi and the relay. I chose the pass conditions to be less than 16 mA as this is the maximum current that can be sourced by the RaspberryPi via a single GPIO pin. Though I listed the desired condition as less than 16 mA, that is only the nominal desired level. There are also desired behavioral characteristics I considered for the pass condition. Since the current draw measurements show that the current drawn increases linearly from 0 mA at 0.8 V and maintains this behavior over the tested range, I feel comfortable labeling this test a pass without fear that the current drawn will approach 16 mA with any voltages the GPIO is capable of providing.

Test	Desired Level	Actual Level	Comments	Pass/Fail
Apply a simulated GPIO voltage to the relay block and measure the current drawn from the applied GPIO voltage to ensure the RaspberryPi can source adequate current to operate the relay. GPIO voltage is simulating using a lab power supply increased in increments of 0.1V. The input voltage is increased from 0V to 5V and current measurements are taken using a lab multimeter. The power side of the relay is supplied 120V at 60Hz and the load side is connected to the lab multimeter.	< 16 mA	< 12.34 mA	The current draw remained < 0.00 mA for inputs < 0.9V and then increased linearly. At 3.3V the current draw is 6.91 mA.	P

Repeat the test with input voltage decreasing from 5V to 0V.	< 16 mA	< 12.34 mA	The current levels measured were almost identical to the previous test.	P
Repeat the above tests with a simulated load to determine if loading of the system impacts current draw.	< 16 mA	< 12.34 mA	The current levels measured were almost identical to the previous tests.	P

Test 3 for RFID Reader and SPI Read
Rationale and description: Test the functionality of the RFID reader as accompanied by Python code on the RaspberryPi. This needs to be tested because this is the functionality that will allow Makers to interact with our system. This test tests general functionality under standard conditions and also some potential failure cases. The distance which is fixed in this test (~ 2 cm) can be fixed in implementation by the structure of the device housing. Reading continuously for a somewhat significant duration of time (> 120 minutes) gives us the best chance of detecting errors, although none were detected throughout this test. This block passes because RFID cards can be read very reliably and simple failure cases caused by user failure or malicious intent can be prevented.

Test	**Desired Level**	**Actual Level**	**Comments**	**Pass/Fail**
Test the ability of the RFID reader to accurately read RFID cards from a proximity similar to what will be expected in the final system (~ 2cm). The RFID reader was set to read every 0.5 seconds. The input was a 13.56 MHz (HF) RFID card with the UID prerecorded and each reading was compared to the recorded UID. Connect the RFID reader to the RaspberryPi through the GPIO pins in a setup to use SPI. Power the RaspberryPi with 5VDC from a commercial power supply through the microUSB port.	> 95%	~ 100%	No errors occurred for the duration of this test (16,434 readings). This test was run with open air between the card and reader. The impact of various materials in this dead space will determine what material we house our module in.	P
Test the potential failure case of multiple RFID cards in proximity of the reader. In this test, two cards are stacked on top of one another and placed in front of the reader. The output is a UID read into the RaspberryPi command prompt.	An accurate reading of one or both cards or no reading	No reading	It will be necessary to ensure that there is only space for one card in proximity of the reader, otherwise a second card may result in the revocation of a Maker's permission.	P
Repeat the above test with a mixture of cards and fobs.	An accurate reading of one or both UIDs or no reading	Strange, erratic behavior	Read function is occasionally triggered multiple times; does not produce a consistent output. Makers will not have RFID fobs but this can be protected against by taking the precautions described in the previous test comment.	F

Test if the RFID reader can read BUIDs. Use a BUID as an input and read the output UID from the RaspberryPi command prompt.	No reading	No reading	This test is necessary because I anticipate many Maker's trying to scan their BUIDs at resources.	P

Test 4 for Controls Module

Rationale and description: Test of individual Python functions. These need to be tested because they are responsible for the functionality of the RaspberryPi within my subsystem and are responsible for communication with the rest of the system. These tests simulate inputs that will be encountered in typical operation. These tests passed as they are able to perform all operations required for basic functionality and first round integration.

Test	**Desired Level**	**Actual Level**	**Comments**	**Pass/Fail**
Test of card recognition function. An RFID card is placed ~ 2 cm in front of the RFID reader. After a predetermined duration of time (tested for 0.5, 1, 5, and 10 minutes) the card is removed. At this point, the code should enter a standby loop where it waits for a new card.	Enter standby function when card is removed	Enter standby function when card is removed		P
Test subsystem communication protocol that will integrate with the microcontroller block. This code waits for a UID from the RFID reader and sends it to a sample function ("Dan's_Block.py") that compares it to a list of UIDs.	UID as an unsigned integer	UID as an unsigned integer		P
Test function that will respond to output from the microcontroller. An input is received in the form of an authorization variable from a simulated microcontroller block. This function sets GPIO pins to high or low values dependent on the authorization value received.	GPIO pins set to high or low	GPIO pins set to high or low	Board numbering was chosen over BCM for ease of prototyping. This has been coordinated between myself, Julie, and Dan to ensure we used the same numbering schema.	P

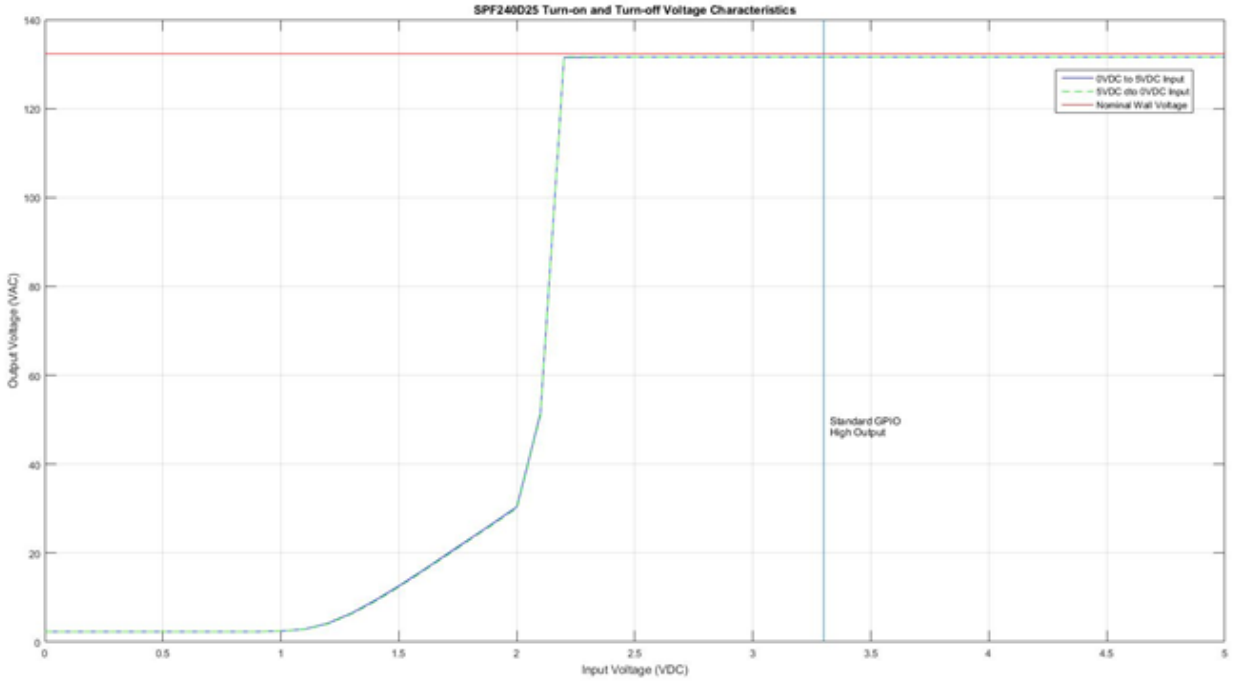
Test 5 for Resource Control Block (Full subsystem and LED)

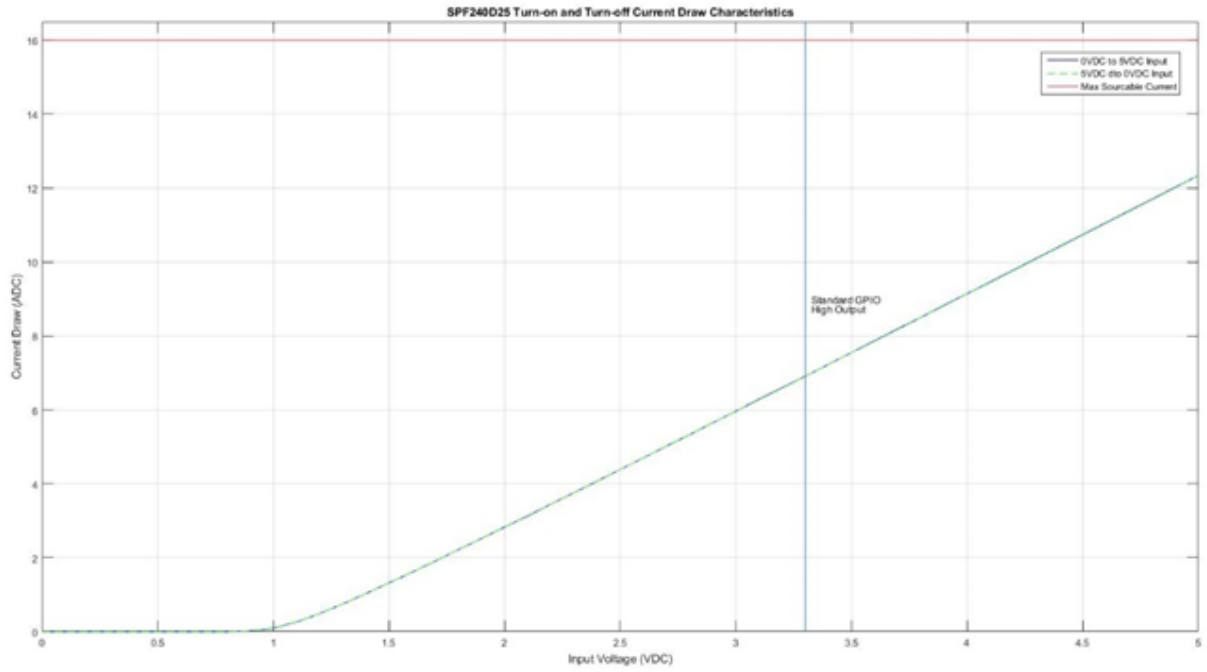
Rationale and description: This is a test for full subsystem functionality as will be required when interacting with other blocks. This test compiles my blocks and interacts with my subsystem in a "black box" manner to ensure that my subsystem can successfully be integrated with the rest of the system. An RFID card will be placed in front of the RFID reader and left there for a number of minutes. While the card is adjacent to the reader, either a red light or green light will turn on to signify if the card was approved or denied. If it is approved, 120VAC is made available to a resource and can be measured at the output of the SSR. Once the card is removed, authorization (if it was received) is removed and the output of the SSR returns to open conditions. If this system is able to operate as a "black box" and provide/deny authorization based on UID, this test will be considered successful and capable of being integrated with the rest of the system.

Test	Desired Level	Actual Level	Comments	Pass/Fail
Test full subsystem functionality. Attach 120VAC at 60Hz to the AC input of the SSR. Attach ground and a GPIO pin of the RaspberryPi to the DC inputs of the relay. Connect the RFID reader to the RaspberryPi through the GPIO pins in a setup to use SPI. Power the RaspberryPi with 5VDC from a commercial power supply through the microUSB port. An RFID card is placed at a distance of ~ 2 cm from the RFID reader. The UID is on a list of authorized UIDs so the output will be measured to be 120VAC with a lab multimeter and a green LED will be illuminated. The green LED will remain illuminated for the duration that the card remains adjacent to the reader.	120VAC Green LED	120VAC Green LED	Based on the observed/measured functionality, my subsystem is ready for first-round integration with the rest of the system.	P
Repeat above test with an unauthorized UID. The red LED will remain illuminated for the duration that the card remains adjacent to the reader.	0VAC Red LED	0VAC Red LED		P

Data from testing:

Relay Testing:



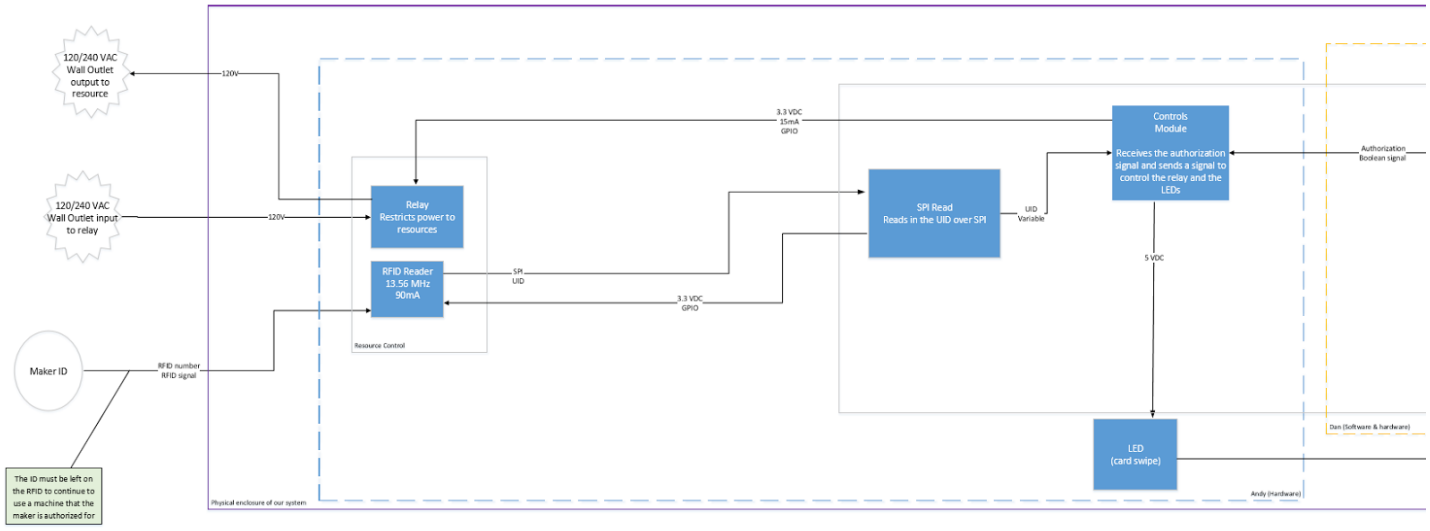


RFID Testing:

Duration:	2:20:00
Correct Readings:	16,434
Incorrect Readings:	0
Average Interval:	0.51 s
Accuracy:	~100%

Technical Block Diagram

Resource Access Control Block:



WiFi Receiver/Transmitter Block (2):

Test Protocol

The tests that involve the microcontroller block for this project are primarily interacting with two other blocks: inputs/outputs from Andy's control module involving the remaining hardware aspects and the input/output from MYSQL block (Joe's block). The tests that are conducted for this block revolve around different scenarios that could occur when the use of microcontrollers on makerspace resources are implemented on a wide scale. Initially, I implemented tests that mocked inputs that would provide the expected output. The next set of tests were conducted to see if an unexpected output can occur that can disrupt the execution due to an unexpected event.

Hardware/Control Block Interaction

There are two main inputs from Andy's block. In order to mock his block I created a python script that would wait for UID input. This was a straightforward task where I am looking for an instance when the RFID reader gives an unidentified UID or the variable is of a different type. There are two different UID functions that Andy will provide: one that will return when a UID is no longer detected after a certain period of time, and another function that will return the UID as an unsigned integer each called in a different case or scenario. As the time it takes to give access to the Maker is almost instantaneous these tests are more geared towards likely and unlikely scenarios.

Table 1. hardware Control tests conducted with RPi

Key: P = Pass, F = Fail, U = Undetermined (more in depth test needed)

Test	**Desired Outcome**	**Actual Outcome**	**Execution Time**	**Comments**	**Result (P/F/U)**
Mock UID input that is not of unsigned integer being used to retrieve appropriate authorized RID	Function return: False	False	0.1336s	Expected, time is due to query MYSQL	P
UID Function call interrupted due to physical RFID malfunction	Red LED blinks 2x	RED Led blocks 2x	43ms	LED just signal of RFID malfunction	P
Unexpected RPi power disconnect (Keep GPIO ports at current state when reconnected)	Resources (LED) remain off	Resources (LED) remain off	80.3ms	Will stay on till Python script runs on boot	U-->P
Wifi disconnected temporarily	Wait for UID state	Wait for UID	--	Handling all exceptions incorporated in each function, except user interrupts	P

The conditions and scenarios that revolve around the hardware interface with the RPi are primarily those that can occur due to device failure such as an unworking RFID reader or even cases where unidentifiable UIDs are scanned or a temporary wifi disconnection that can disrupt and stop the script due to an error. I've modified my script so that these scenarios will not pose a major problem in the durability of my script. Again I've incorporated in Table 1 the execution time of certain scenarios while the script is running using built in time functions in python and found execution time to be unnoticeable by human standards.

Database Block Interaction

Joe's block handles the organization of the MySQL database where it is separated into 6 different tables all having different functions and holding different data types. There is a bit of overlap in terms of testing where I am querying/writing to the MySQL but my block isn't emphasized in testing the durability of data storage and transmission but more emphasized toward the logic carrying out specific functions that will allow authorization for a user to a specific UID. To mimic Joe's MySQL tables, I am creating an additional empty table in the existing database using Bucknell

network that I will use to carry out my tests using python by querying and writing to it. It will include the 5 data types that I will require: RID, ResourceType, UID, needsTraining, macAddress. I will include the execution time in the table as it will reflect a more realistic time taken to execute the python script and data transmission via Bucknell wifi.

Table 2. Database Block Testing with RPi
Key: P = Pass, F = Fail, U = Undetermined (more in depth test needed)

Test	**Desired Outcome**	**Actual Outcome**	**Execution Time**	**Comments**	**Result (P/F/U)**
Python script auto starts at bootup	Run script at bootup and ask for UID	Run script at bootup and ask for UID	<30s	Run without interrupts	P
Look for a resource's RID when that resource has been added to the MAC Addresses table.	Function return: RID	RID	0.14s	Expected behavior	P
Add a MAC address to the MAC Addresses table when there is no RID.	Wait for Admin to enter RID	Wait for Admin to enter RID	0.23s	Will enter wait loop till RID is entered.	P
Execute main() when RID is detected and UID is authorized	Green LED ON	Green LED ON	1.266s	Takes a little too long for access. Look for ways to optimize	P
Execute main() when RID is detected and UID is unauthorized	Red LED ON	Red LED ON	1.365s	Takes a little too long for access. Look to optimize.	P
Wifi/MySQL disconnected temporarily	Wait for UID state	Wait for UID	--	Handling exceptions incorporated in each function	U->P
Recognizing new Resource previously not added with user authorization	Green LED ON	Green LED ON	2.329s	Scaling to new resources added to labs	P
Admin deletes a resource from the Resource Table while RPi is running.	Nothing/ resource boots down	Nothing/ resource boots down	1.255s	After bootup and retrieves RID, will retain RID. Reboot causes wait till RID is assigned	U
Admin adds UID to authorized table. Rescan UID with authorization, RPi no reboot	Green LED ON	Green LED ON	2.310s	Previously unauthorized, Admin adds authorization. Allow access	P
Mock UID input is removed while program is providing power to resource	RED LED blink 3x	RED LED blink 3x	32.41s	Waits 30s to detect UID card before shutting off.	P

Mock UID input is removed and placed back on RFID while program is providing power to resource	Green LED blink 3x	Green LED blink 3x	1.26s	Mock input with additional RFID function	P
Mock UID input is removed and different unauthorized UID is placed back on RFID while program is providing power to resource	RED LED ON	RED LED ON	788ms		P
Mock UID input is removed and different authorized UID is placed back on RFID while program is providing power to resource	Green LED ON	Green LED ON	1.134s	Allow access but not shut off resource	P
Mock RFID scanner is off temporarily while resource is powered. Reconnect power within 30 second time frame	Constant Green LED ON	Constant Green LED ON	--	Should not disconnect power, error handling so program still function	P
Mock RFID scanner is off temporarily while resource is powered. Reconnect power after 30 second time frame	Constant Red LED ON	Constant Red LED ON	--	Should disconnect then reboot	P

Technical Block Diagram

WiFi Receiver/Transmitter Block:

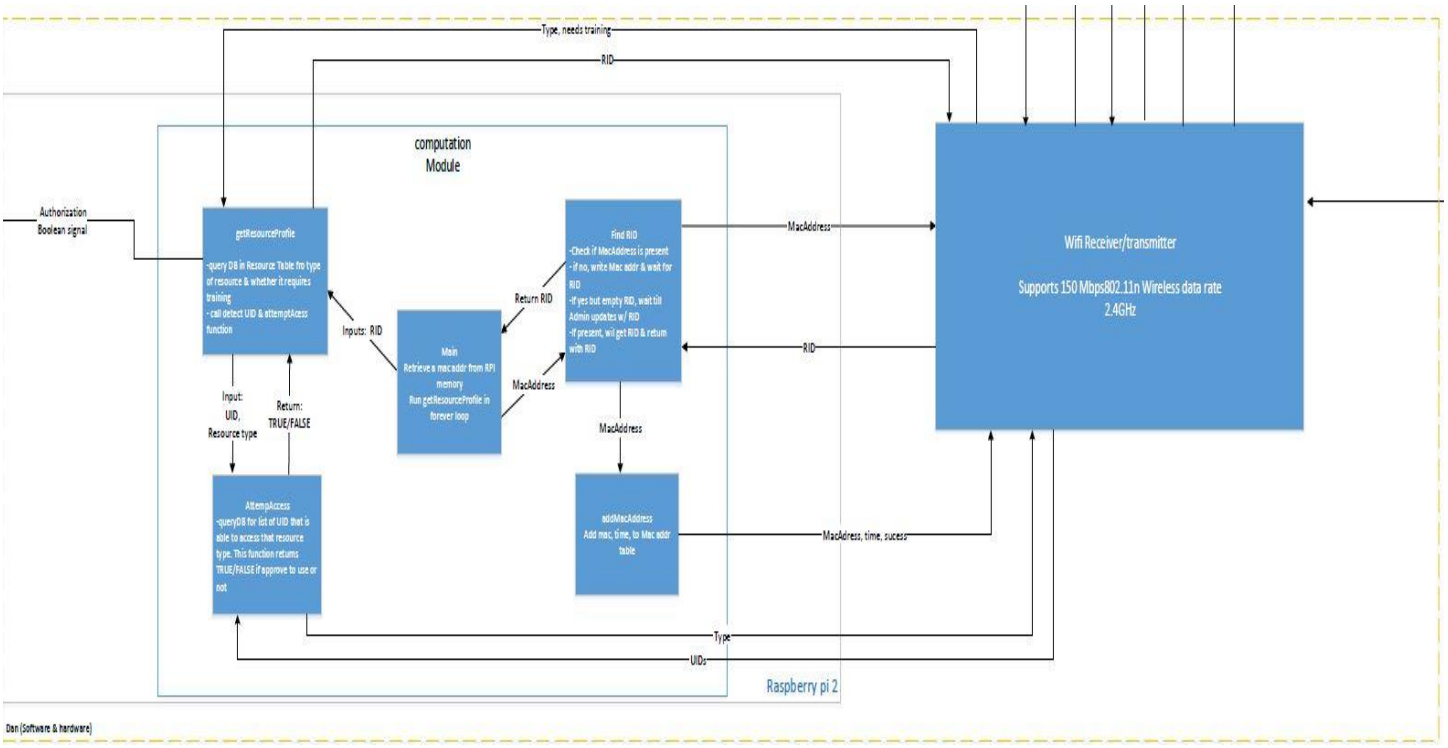


Figure 1: The microcontroller block within the system

Internal Function Testing Protocols

The inputs and outputs that my block interacts with have been covered in the previous test protocol section. There are several other tests that measured the performance optimization of the microcontroller. The RPi runs on a 900 Mhz quad-core ARM Cortex CPU with an ethernet port as well as 4 USB ports, one of which I utilized a wifi dongle to wirelessly interact with the database. An rivaling microcontroller such as the Arduino Mega which has a clock speed of 16 MHz will have a slower performance rate because of the continuous interaction with MySQL because it will primarily query for different data types multiple times within a 30 second interval. The tests I have executed are more focused on data transmission via ethernet/wifi. The obvious advantages of Wifi is lack of physical resources around the labs and ease of implementation, however ethernet intuitively may clearly present a more reliable option. In order to see the durability of both options I have measured the performance of speed, execution time.

Table 3. Quantitative testing data rates

Key: P = Pass, F = Fail, U = Undetermined (more in depth test needed)

Test	**Desired Outcome**	**Actual Outcome**	**Execution Time**	**Comments**	**Result (P/F/U)**
Time taken to execute attemptAccess function at set intervals (ethernet, Wifi)	Time to execute program should remain linear	Linear but at different rates	See graph	Potentially important in determining allocated time for resource run w/o UID	**U-P**

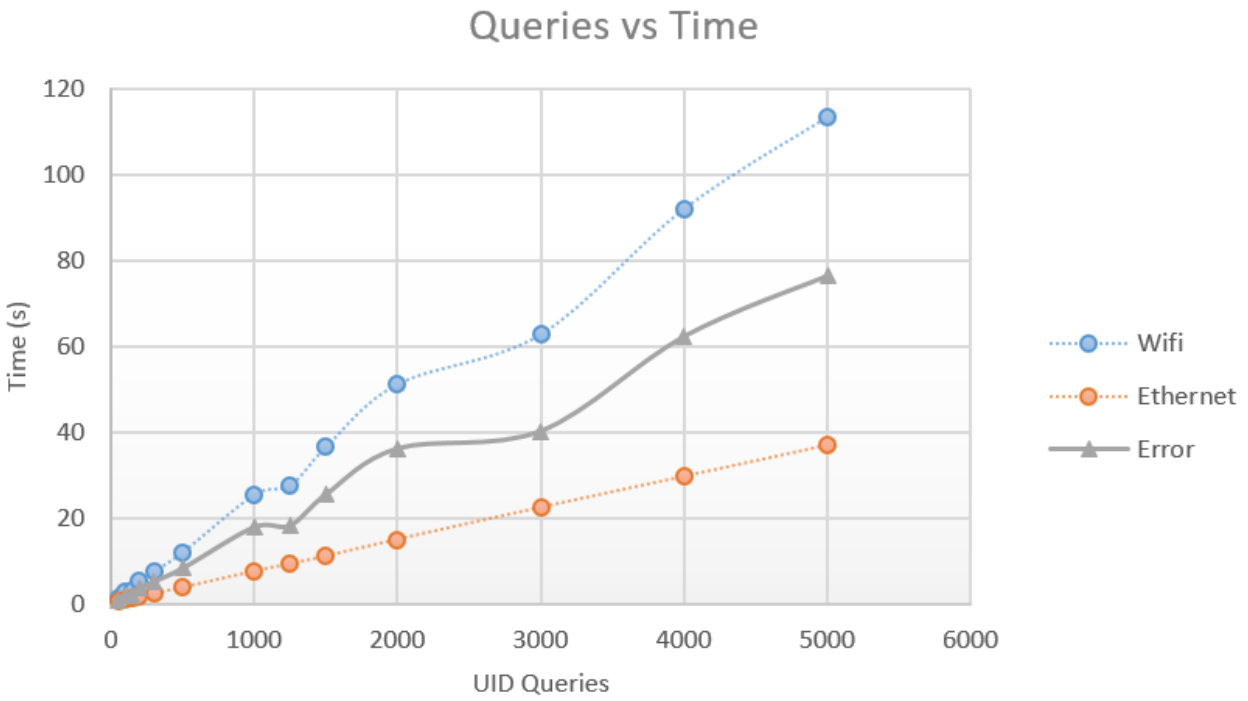


Figure 1. Execution time for continuous queries simulating checking UID when resource is on.

WordPress Webbapp Block (4):

Overview

The unit tests for my block were split into three major sections; website interface and functionality, database communication, and the website back-end performance. For the first section, I wanted to to ensure that website behaved as expected for makers, admins, and anonymous users so I structured my tests to include those difference cases. The communication between my block and the database (Joe's Block) is the backbone to the software part of our system, so I needed to run various PHP and MySQL calls to test certain inputs and outputs. Lastly, I needed to run test on the back-end. This covers things like security, structure, and server functionality.

User Interface

These tests were all about the front-end development of the website. This involves what the user can see and interact with directly on the site. For development tests on the front-end, resources provided by the World Wide Web Consortium were used to ensure current web standards and protocols were being followed as well as WordPress coding standards. On the user interface side, all of the inputs would be through mouse click, button press, keyboard entry etc and were simulated by manual testing.

Key: P = Pass, F = Fail.

Test	Desired Result	Actual Result	Comments	P/F
Test and simulate the access of the website using online resources (Respnsinator and http://ami.responsivedesign.is/)to get an indictation of how responsive site will look on popular devices. Go through clicking links, visiting pages, and using login form using touchscreen	Site is fully functional (links work on button press and text boxes work with keypad) and images are displayed correctly.	Site works as desired	Simulated access on all Desktop, Laptop, Tablet, and modern smartphones. Physically tested on HTC Nexus 9 Tablet, iPhone6, HTC M8 and Lenovo Laptop.	**P**
Check links and anchors on website using W3C Link Checker. This recursively parses through the site ensuring all URLs	Website is crawled and returns with no broken links, invalid	All links that can be accessed or traversed by user	*At completion, web tool threw error because of server	**F***

are valid on the various web pages and there are no 404 error pages	anchors, or other error codes	are valid. Some links to the resource point to broken URI fragments	security, so test wasn't fully ran. All visible links are working, so further testing is needed in regard to potential URI fragment errors. May be caused by using object "name" instead of "id" in code.	
Check the markup (HTML, XHTML etc) validity of website. Ensures language follows standard rules and protocols. Uses W3C HTML checker and Markup Validator tool to check correct syntax of pages. This is to make sure HTML pages are valid across browsers (and browser versions) and display/function correctly.	Website homepage and pages are crawled with HTML parser and corresponding scheme and document is returned with no errors or warnings.	Document checking completed, no errors or warnings to show	One or two pages had "obsolete" language attributes that can be safely omitted, but make no difference to functionality.	**P**
Run W3C CSS Validation Service to check cascading style sheets. This checks if the website complies with the CSS standards set by W3C and ensures cross-browser and future compatibility.	Website is scanned and the returned result shows no critical errors	The tools found 621 errors, mostly focused on the Instagram plugin and Optimizer Theme that is being used on the website.	The errors do not affect website functionality, but rather this means the CSS has the potential to not display correctly is certain instances	**F**
Test WordPress PHP code and JavaScript code using PHPUnit and QUnit respectively. WordPress offers these test suites for unit testing websites and projects. By locally installing and setting up both testing frameworks with an empty MySQL database and test config file,, I can confirm installation code is working as expected as well as any custom written PHP code.	Test suite output prints resulting tests with only passed or skipped tests and no failed test, PHP error/warnings, or incomplete results.	Test output returned as expected, no failed tests	These tests were made for testing the WordPress core so were mostly just confirming WordPress installed correctly.	**P**
Social media plugin Instagram works correctly and updates accordingly whenever the Bucknell Maker-E or Bucknell MakerSpace instagram accounts post a new picture.	When either account posts a new image, the gallery page should update within 20 minutes and shift the images correctly. Clicking the picture should direct the user the the source of the image on Instagram.	Website gallery page updates as expected, shifting pictures over with the newest image first. User can click "Load More" to view more images or can click an image and view on Instagram.		**P**

Test access to Student Tech page on the website. This is through the "Admin" button on the front page. The page will prompt for a username and password and will redirect on authorized login. This is tested so that unauthorized users cannot edit the database in any way. Conduct the test by logging in as an admin, a student tech, a regular user, and by accessing the student tech page directly by URL.	The student tech page should only be visible to Admins or Student Techs who have successfully logged in. Failed login attempts or attempts to access the page directly by URL will present an error message.	Trying to visit the page by typing the URL will present a "protected content" message and a link to sign in. Signing in without correct credentials will give "user not found error." Both admin and Student Tech will redirect to desired page as expected.		**P**
Test access to Admin specific page. Much like previous test, this is to limit access to specific location where student tech profiles can be created. Test by trying to access link on student tech page, this would confirm even logged in techs cannot access page as well as makers.	Page can only be accessed by user signed in with admin account.	Page is protected as expected. URL cannot be accessed directly, and both student techs and users not signed in cannot access page		**P**

Interacting with the Database

As mentioned before, the communication between the website and the database is the main component of the software side of the system. This is where makers will be able to search the database and admins will be able to create, insert, edit, delete, and search the database to find and interact with the desired information. With the choice of MySQL to store our database information, it is impossible to be fully independent from Joe's block because WordPress runs off of the same database. Therefore to test, I used a WordPress plugin to simulate the database tables so that I could view all changes to a test table I created called "Employees." I am also using a WordPress plugin (ABASE) for the communication, so the specific API calls and PHP code is abstracted away from me in this instance. However, I can still edit the shortcodes to create and test the functionality I am trying to achieve.

Key: P = Pass, F = Fail.

Test	**Desired Result**	**Actual Result**	**Comments**	**P/F**
Create a new table in the database. This is done using the CREATE TABLE SQL statement in the plugin shortcode. By specifying the name as "employee" with frields "firstname" and "lastname" a table will be craeted.	A table should be created in the database with name "employees" and fields "id"," firstname", and "lastname."	Empty test table was created with specified fields.		**P**
Create a form to insert a record into a table in the database. This can be used to add training, add resources, among others to various database tables. By using a shortcode I am creating a form which text boxes for first and last name entry as well as a submit button that when pressed will add the record to the specified table.	The desired test form will be created on the page the code is placed. The form will have two text boxes with labels as well as a submit button. On submission, the "employees" database table should be updated with the information submitted.	Form was created correctly. Entered "Ian Wallace" in the text fields and pressed the submit button. On inspection of the database table, "Ian Wallace" can now be found		**P**

Create a search form to search the specified database table for desired information. This is useful for makers and admins who may want to search for a user and determine what training they have. In my test, I want to create a search form that has text boxes and a submit button so that users can search with first, last or both names. The user will redirect on search to a search result page where results will be shown. After creating the form, I searched for "Ian" and "John" separately.	User can search the database using the text fields provided. On successful search, the user will redirect to a page with a table displaying results. On an unsuccessful search, user will redirect to a page with no results.	Search performed as expected. When using the form to search "Ian", I was redirected to a page with the results "Ian Wallace" in a table. When "John" was searched, no results were displayed.	On testing it seems upper and lowercase does not make a difference and partially filled out boxes will still search	**P**
Create a form to update a table entry. This test required a creation of a form with clickable links (in this case lastname) where a user can click on a table entry to redisplay the page with the primary key and value in the query string. This allows the chance to update specific fields (change first name) or delete the entry from the table. For the test I wanted to select the entry "Ian Wallace", change the first name to "Bob" and then delete the entry.	A table will be displayed on the desired page with all entries of the employee table. By clicking the last name, text fields will appear to update the entry and a delete button will appear to delete the entry. By viewing the database table, each change can be viewed as it happens.	Form works as expected. Using the text field to change first name to "Bob" updates the table in the database accordingly. By selected delete, the entry is deleted in the "employee" table and the table is now empty.	All updated fields are currently done through text box, so empty text box counts as valid entry.	**P**

Website Back-end and Performance

The last set of tests I wanted to run were in regard to back-end development. This would include security, performance, information exchange, and server/site failures. With security, I wanted to make sure there was no major security risks with the site or with written code. Performance testing was done to check load times and provide the best user experience. Lastly, I wanted to create tests to determine if the site can hold up to increased load with multiple users accessing the site.

Key: P = Pass, F = Fail.

Test	**Desired Result**	**Actual Result**	**Comments**	**P/F**
Scan the website for security risks, out of date software, malware, and any other security violations. Ran a Sucuri SiteCheck and a WordPress Security Scan to test vulnerabilities.	Results return Malware free with no application, plugin, or other security warnings.	Google safe browse check passed and Sucuri Site Check and WordPress Security scan returned with no security warnings.	The best security protection is making sure WordPress and all Plugins are continuously supported and updated.	**P**
Run various performance tests on the website. Using online resources such as "WebPageTest," "Google PageSpeed Insights," and "Pingdom" to determine web	Passing grade (>60%) on all test websites for both mobile and desktop browsing	Failing grade on mobile by Google (C on Desktop), B grade on Pingdom,	Image compression and poor leverage of browser caching lowered grades. Server Response time	**F**

performance on desktop and mobile and how to improve any issues.		and average B grade by WebPageTest.	was low, but that cannot be changed on my end	
Perform load testing to simulate for an influx of users on the website over a short time frame. Monitor data throughput, information exchanges, and failures over the testing time frame. Use LoadImpact and Neustar Web Performance Management for simulation. The first will simulate an average of 30 users an hour while the second will simulate a ramp up of 100 users in 5 minutes.	For one hour test, data throughput and transactions maintain constant with no disruptions and no failures over the time frame. The rapid ramp up of 100 users also has 0% fail percentage and the site/server handle increased load.	All tests passed as expected, 0% failure rate. See graphs below for more information.	The max testing I could do was 100 users without paying for a service. However Bucknell servers have experience with increased load so results should scale.	**P**

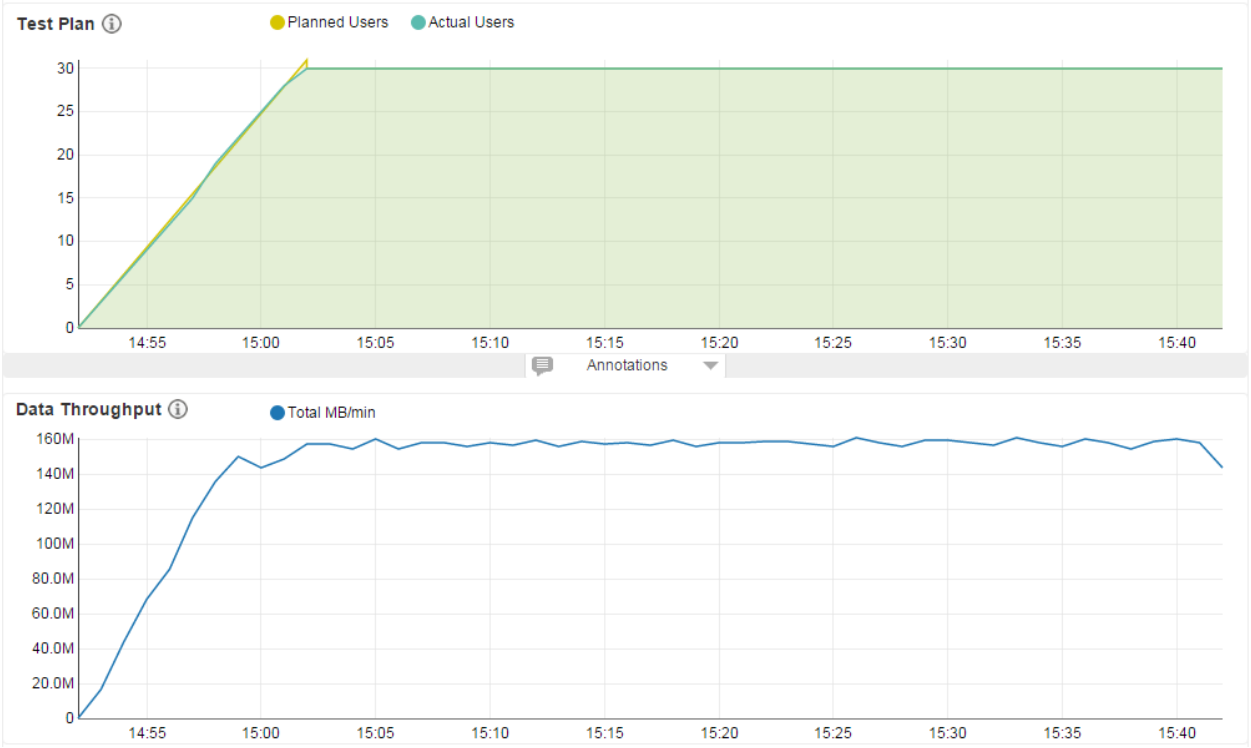


Figure 1: Simulated test plan and data throughput of 30 users 50 minutes.

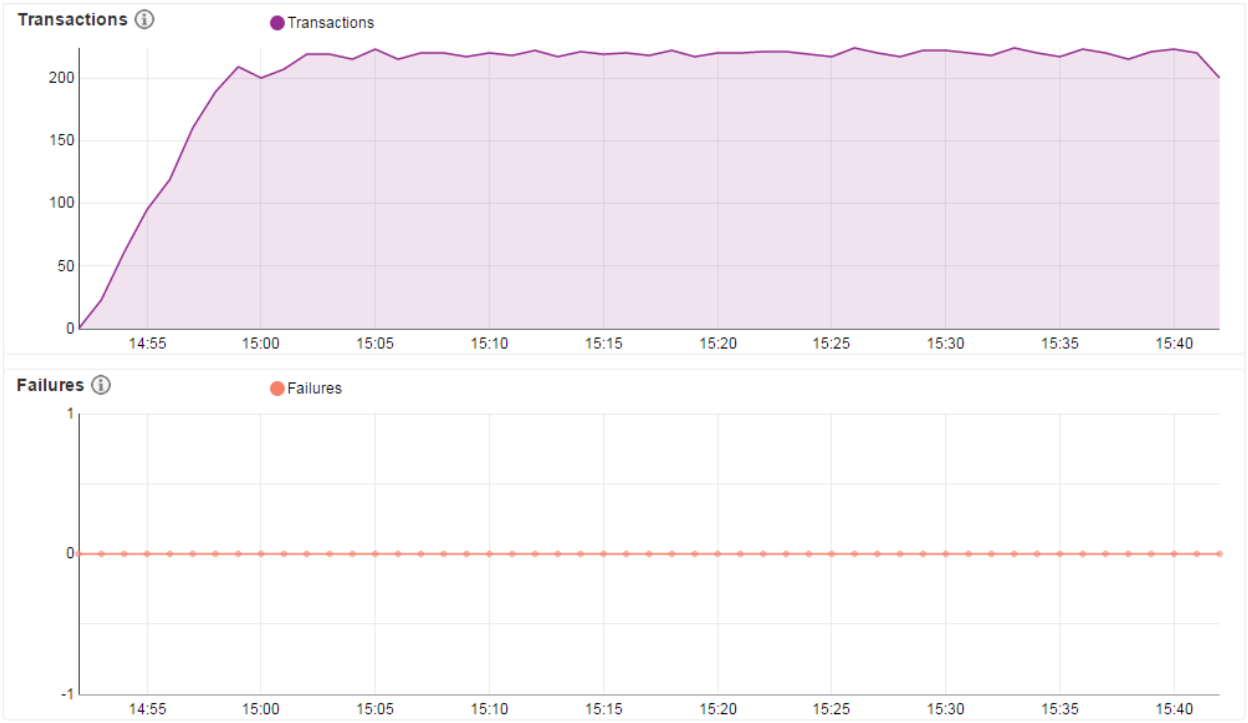


Figure 2: Transactions and failures over time frame described in Figure 1.

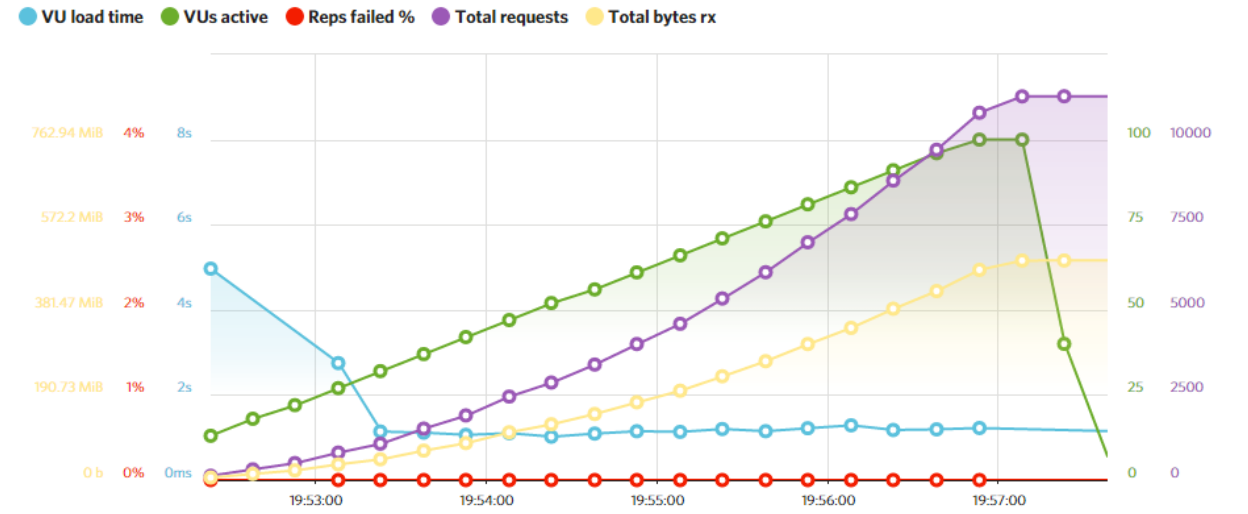


Figure 3: Increase to 100 concurrent virtual users on website in 5 minute time frame.

MySQL Database Block (4):

Test Protocol

The majority of my tests were split into two parts; inputs/outputs from the microcontroller side (Dan's block) and inputs/output from the webapp side (Ian's block). Generally, I tried to implement tests that would be expected inputs from the rest of the system, and timed these tests to ensure they are within performance specifications. I then added inputs that would be unexpected to my block, to ensure it behaved in a normal manner.

Microcontroller Side

To mock the inputs from Dan's block, I created several Python scripts that would use the same calls that he would in interacting with the database. For all database transactions, we are looking for an execution time of at most 20 milliseconds, so this criteria has been omitted from the table.

Key: P = Pass, F = Fail, U = Undetermined (more in depth test needed)

Test	**Desired Outcome**	**Actual Outcome**	**Execution Time**	**Comments**	**Result (P/F/U)**
Look for a resource's RID when that resource has yet to be added to the MAC Addresses table.	No entry found in table.	No entry found in table.	7ms	Expected behavior	**P**
Add a MAC address to the MAC Addresses table.	Entry added to table.	Entry added to table.	6ms	No RID was added.	**P**
Look for a resource's RID when the resource been added to the MAC address table.	Entry found, but NULL RID.	Entry found, but NULL RID.	6ms	Wait for the webapp to add an RID	**P**
Look for a resource's RID when the resource has been assigned one from the webapp	Entry found, has valid RID.	Entry found, has valid RID.	8ms	This is a preconfigured resource	**P**
Insert resource profile with mocked data	Confirmed existing resource profile in table	No errors thrown on insert	5ms	Confirmed by following test	**U => P**
Get resource profile for existing resource	Get associated location and type	Got associated location, type, and needs training flag	5ms	Previous test confirmed to work	**P**
Attempt access when the user is not authorized	Database confirms user's UID is not authorized	Database confirms user's UID is not authorized	5ms	Critical system functionality	**P**
Log failed access attempt	Log table updated with failed access attempt	No errors thrown on insert	6ms	Confirmed on phpMyAdmin that the entry is there	**P**
Attempt access when the user is authorized	Database confirms user's UID is authorized	Database confirms user's UID is authorized	5ms	Critical system functionality	**P**
Log successful access attempt	Log table updated with successful access attempt	Log table updated with successful access attempt	6ms	Confirmed on phpMyAdmin that the entry is there	**P**

To mock inputs from Ian's side, I created several PHP scripts that would mimic the calls he would be making. I chose PHP because Ian is planning on leveraging several WordPress plugins to create his block. Underneath, WordPress uses PHP on the server side, and it is what the database queries be run in. Again, execution time requirements have been omitted from the table, as we require all queries to execute in 20 milliseconds or less.

Key: P = Pass, F = Fail, U = Undetermined (more in depth test needed)

Test	Desired Outcome	Actual Outcome	Execution Time	Comments	Result (P/F/U)
Add two complete user profile to the database	User profiles exists in database, including all fields	No exceptions thrown on insert	3.8ms	Will confirm with next test	**U => P**
View multiple user profiles on database	Print all UIDs, name, email, year, and department	Print all UIDs, name, email, year, and department	1.9ms	Confirmed previous test	**P**
Remove a single user based on their UID	User profile is removed from database	No exceptions thrown on delete	1.7ms	Will confirm with next test	**U => P**
View single user profile on database given UID	Single profile prints to screen	Single profile prints to screen	1.9ms	Confirmed previous test	**P**
Added two complete resource profiles to the database	Resource profiles exist in database, including all fields	No exceptions thrown on insert	4.0ms	Will confirm with next test	**U => P**
View multiple resource profiles on database	RID, type, location, and needsTraining flag for each resource prints to screen	RID, type, location, and needsTraining flag for each resource prints to screen	2.4 ms	Confirmed previous test	**P**
Remove a single resource based on its RID	Resource with given RID is gone from table	No exceptions thrown on delete	1.9 ms	Will confirm with next test	**U => P**
View single resource profile on database given RID	Single profile prints to screen	Single profile prints to screen	1.9ms	Confirmed previous test	**P**
Add two training request to the database	Training requests appears in training table with isComplete set as false	No exceptions thrown on insert	4.4 ms	Will confirm with next test	**U => P**
View all training requests in training table	UID, resource type, timestamp and is complete all print to screen for each request	UID, resource type, timestamp and is complete all print to screen for each request	2.7ms	Confirmed previous test	**P**
Modify training request to be set as complete	Training request of supplied UID and resource type s set to complete	Training request of supplied UID and resource type s set to complete	28.2ms	Seemed like a fluke with timing on repeat of test	**F**
View an empty log table	Nothing is printed to the screen	Nothing is printed to the screen	1.9ms	Would only happen on system initialization	**P**

View all log entries, successful and unsuccessful	UID, RID, timestamp and was successful flag printed to screen for each log entry	UID, RID, timestamp and was successful flag printed to screen for each log entry	2.7ms		**P**
View a user's authentication based on their UID when they are authorized for nothing	No resource types are returned from query	No resource types are returned from query	1.9ms	New maker behavior	**P**
Add authentication to a specific UID for two resource types	Both resource types are returned from query	Both resource types are returned from query	4.8ms		**P**
View a user's authentication based on their UID when they are authorized for more than one thing	All resource types they are authorized for are returned	All resource types they are authorized for are returned	2.3ms		**P**

Technical Block Diagram

MySQL Database Block:

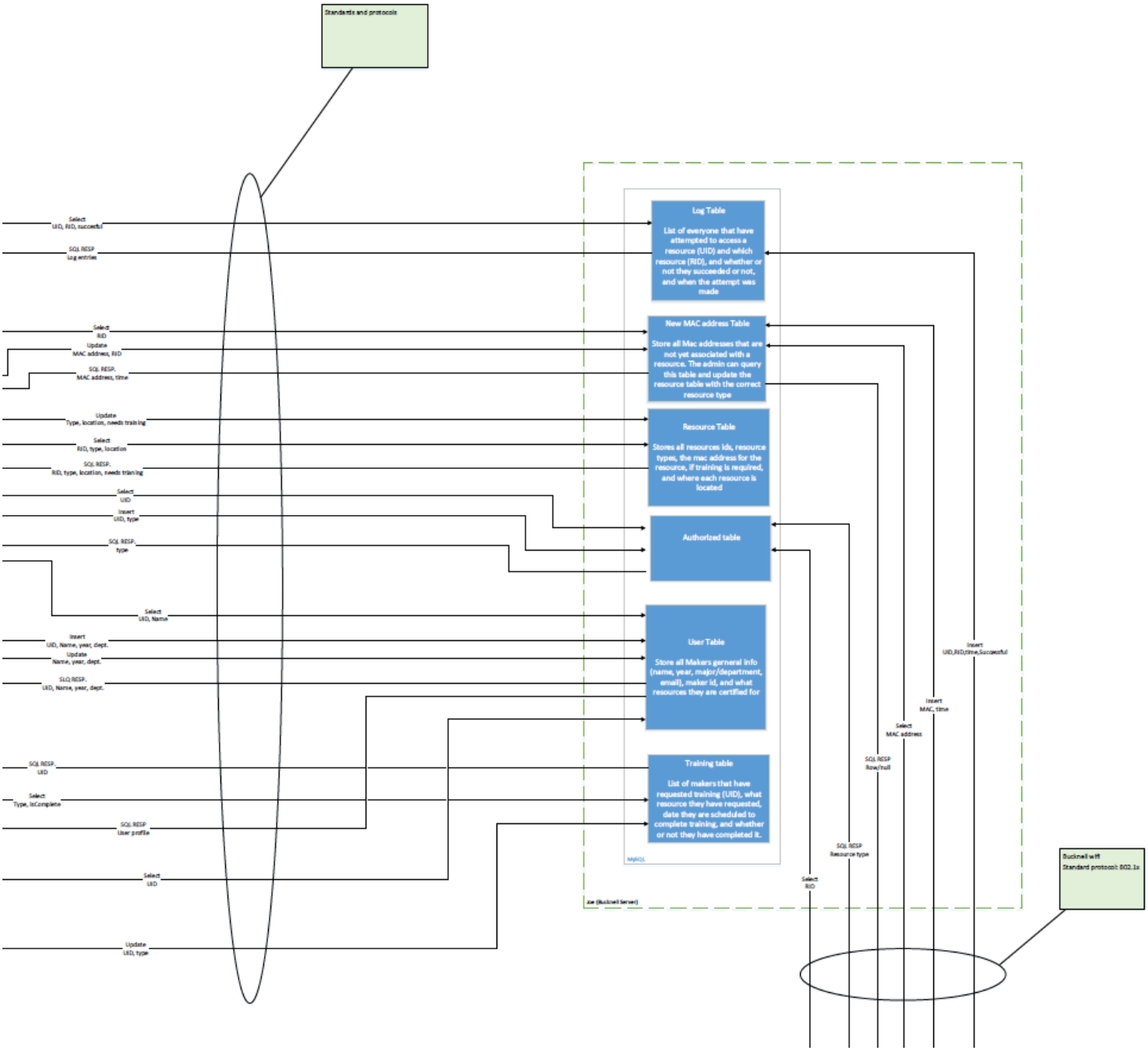


Figure 1: The database block within the system

Detailed Application Information

This section provides instructions to users on how to get your device working (one or two paragraphs for each bullet).

- Provide installation/update instructions including how to connect, program, power, etc.
- Describe in detail how to use the device and how to connect the device to other systems
- Describe common fault conditions and how to correct (debug) them
- Describe any user selectable hardware or software features and their impact on operation
- Describe if necessary how to calibrate the device and or test for proper operation

Installation and Instructions:

The system will be installed by Matt Lamparter in the Maker-E Maker Space. A detailed hardware assembly and update manual will be provided to the client.

Connecting the System:

As mentioned above, the system will be delivered installed in the 7th Street Maker Space. The installation manual will contain a very detailed description of how to set up the system in a new maker space. The hardware is designed to run the software that acts as the brains for the system upon booting up. Connecting the system is to each resource in the maker space is rather simple however. One end of the system will plug straight into any 120VAC or 240VAC wall outlet and the other will provide a female socket for the resource to be plugged into. With these two steps, the system should be active.

The UI and database side of the system is already established and only requires the addition of resources/makers. To be added as an Admin, contact Dan Mancusi in L&IT. Once given a username and generic password, take the following steps to reset the password:

1. Access www.eg.bucknell.edu/makerspace/ using any internet browser.
2. Select the "Admins" tab.
3. Login using the username and password given by L&IT.
4. Select the "Edit Profile" link.
5. Change the password and select "update".

To add a resource, an admin can follow the steps below:

1. Obtain an assembled resource control/communication box (see hardware assembly manual).
2. Power the resource control/communication box by plugging it into the appropriate wall outlet.
3. Plug the resource into the resource control/communication box.
4. Access www.eg.bucknell.edu/makerspace/ using any internet browser.
5. Select the "Admins" tab.
6. Select the "Add New Resource" link
7. Select the new MAC Address discovered at the time the resource control/communication box is powered on.
    a. NOTE: Do not set up another resource control/communication boxes before completing all of the listed steps. If multiple resource control/communication boxes are booted up at the same time, there is no way to distinguish which MAC Address belongs to which box.
8. Type the desired Resource ID (RID) into the text box and select "set RID".
9. The resource is now ready for use by authorized users.

Debugging the System:

Contact L&IT.

*For detailed User Manuals please contact Matt Lamparter

Application Example

- Provides some detailed examples such as a step-by-step guide to operating the project aimed at the end user(s).

Described in the project description

References

Use the IEEE Style. Details can be found on the course web site or in the IEEE Style Manual. References **must** be cited in the body of the text.

Appendix

- Provide any relevant diagrams, schematics, PCB Layouts, Code, or other technical documentation.