Shed Check-In System Access Box Build Procedure
Version 1.0

Author: Ian Dennison
Tawa MenzShed
Date: 07/07/2024
Version: 1.0

Table of Contents

Contents

1 Introduction

1.1 Document Purpose

This document describes the physical build and software installation of the Shed Check-In System (SCIS) Sign-In and Access Boxes.

1.2 Intended audience

This document is intended to be read and used by people with relevant technical knowledge (microelectronics, Arduino programming, and computer networking). Sheds wishing to implement this should seek out members or associates who have the required skills. For more details, refer Skills Requirement section in primary Shed Check-In System Manual.

1.3 Architecture

The SCIS Client Images are designed to run on WEMOS D1 Arduino Devices, with a built in WIFI component. Both the Sign-In / Sign Out Box and the Device Access Box are configured to run on these devices.

1.3.1 Sign-In Box

The Sign-in Box is a single box that is used for registering User sign-ins and sign-outs. It is also used to validate Administrators for access to the Admin functions of the System, so should ideally be placed near the Central Server.

1.3.2 Device Access Box

The Device Access Boxes are used for validating a User request and granting or denying access to a specific device (machine).

Each Access Box is identified by a unique identifier called a WWN, which is derived from the network chip on the Arduino device. This is matched up to the configuration inside the Central Server to provide access to the hardware attached to the Access Box.

When building the Device Access Box, your Power Relay will one of two types:

i) High – when the current applied is High, the access is opened

ii) Low – when the current applied is Low, the access will be opened

The current catered for at the moment is 10A, with a control voltage of 3.3V (matching the Arduinos voltages).

1.4 Preparation

1.4.1 Hardware Requirements – Sign-In Box

For the Sign-In/ Sign-Out Box, you will need the components listed in Appendix A. This box can be powered by a longer USB Cable connected to an external power supply (It may be possible to power it from a spare USB port on the Central Server itself).

Photo 1 – External View of Swipe Box

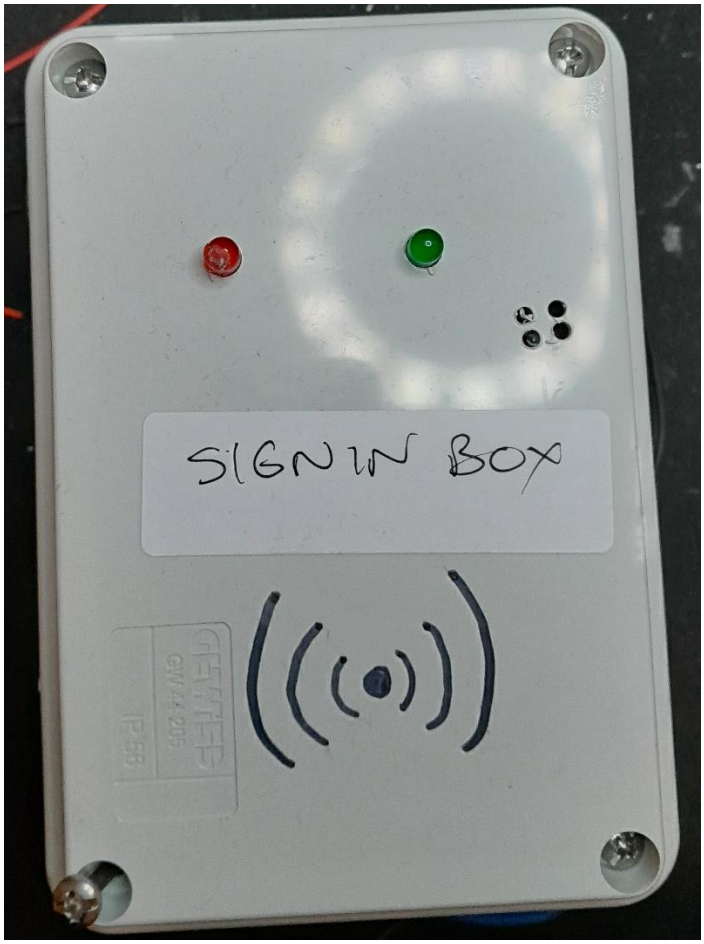
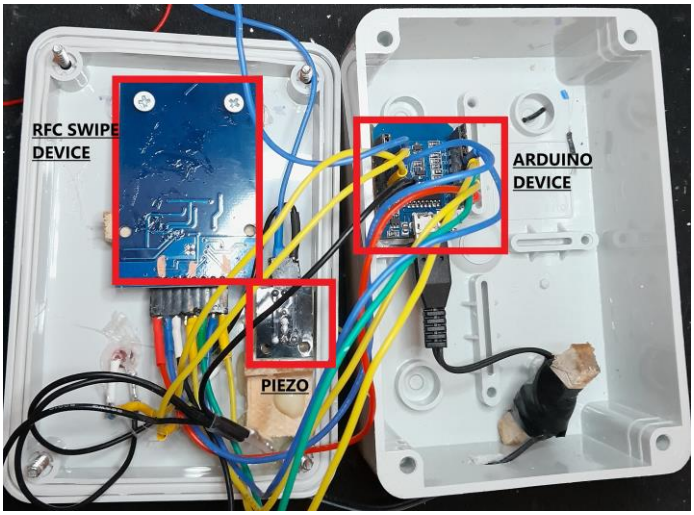


Photo 2 – Internal View of Swipe Box



1.4.2 Hardware Requirements – Device Access Box

For the Sign-In/ Sign-Out Box, you will need the components listed at Appendix B.

The Device Access Boxes do involve more components so can compete for internal space. It is recommended to add an intermediate plug for the 2 x LEDs to allow the top of the device to be removed for maintenance. Also leave the bottom left area free, as future releases may use this space.

Photo 3 – Device Access Box (Top Cover)

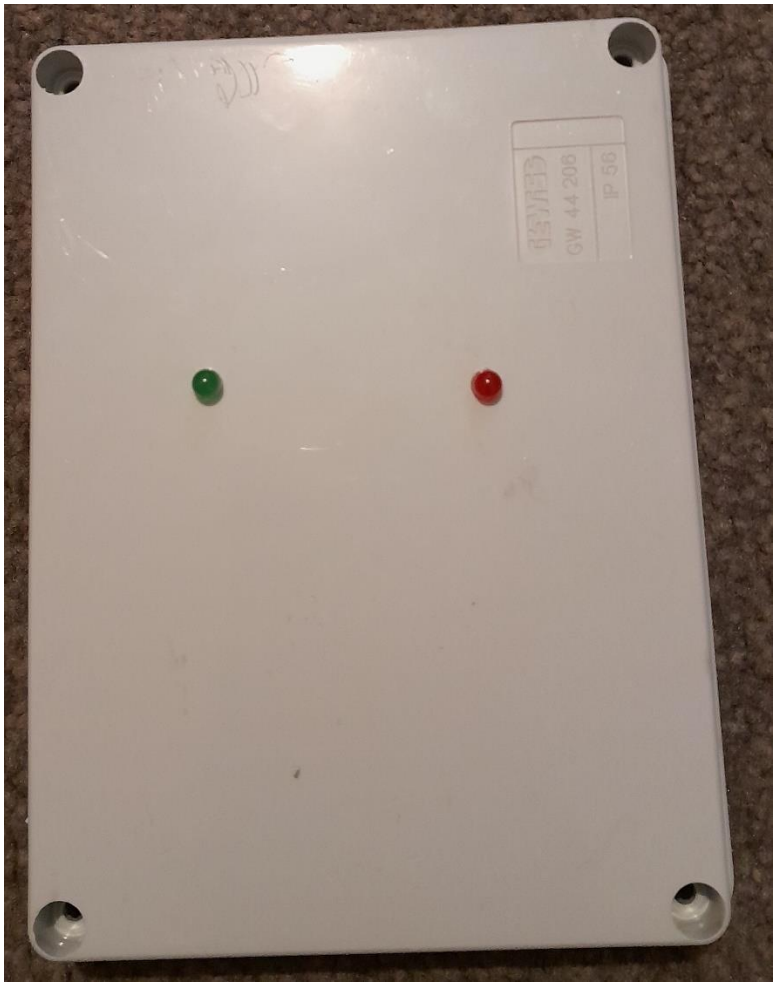


Photo 4 – Device Access Box (Top underneath)

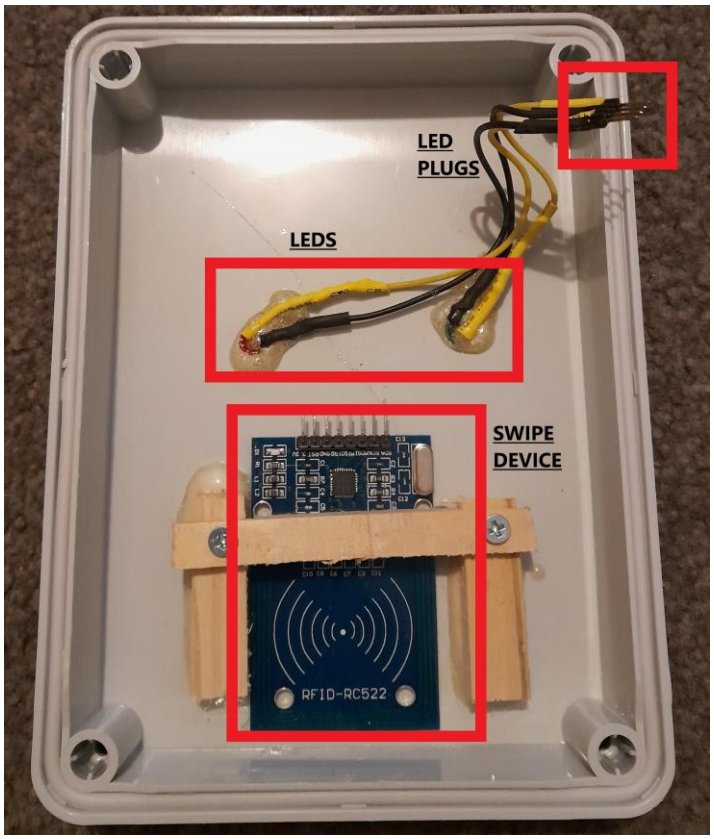
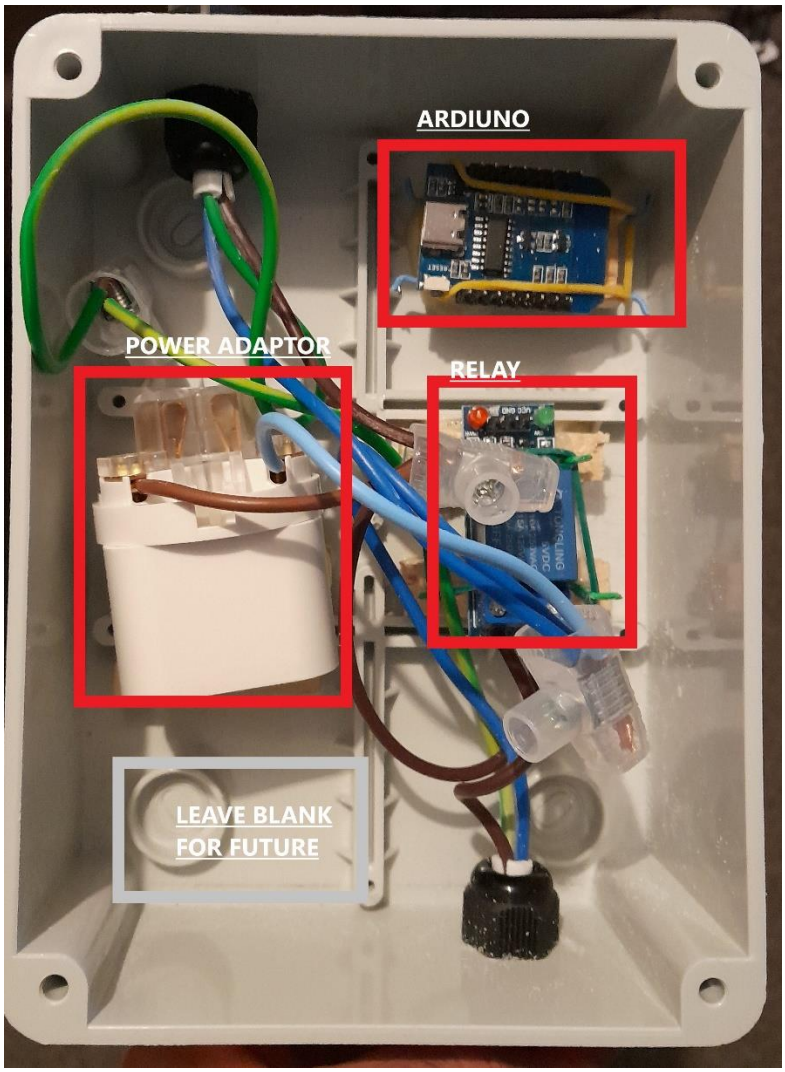


Photo 5 – Device Access Box (Body) – cables removed for clarity



1.4.3 Software Requirements

There are 2 main images for the Device Access Arduino, depending on the type of Relay being deployed. At present, it is trial-and-error on which image is required, but an incorrect image should not damage the device (be sure to test before installing the Device Access Box).

Field	Rationale
machine_box_WWN_High.ino	Works on the WWN based addressing and activates Relay when current is High
machine_box_WWN_Low.ino	Works on the WWN based addressing and activates Relay when current is Low

In order to read the Wiring Diagrams for the Boxes, you will need the following software:

i) Adobe Acrobat – This is the free reader application for PDF Files. It can be downloaded from https://get.adobe.com/reader/. It also comes bundled in newer web browsers.

ii) Fritzing – This circuit layout software is required if you wish to modify the .fzz sketches that are included as part of the Software Bundle and also included in PDF form for easy viewing. It can be downloaded from https://fritzing.org/.

To manage the Arduino devices and download the images and drivers, you will need the following software on a windows-based device.

iii) Arduino IDE – This is the software needed to manage the Arduino Devices and deploy the images to the Arduinos. It can be downloaded from https://www.arduino.cc/en/software.

1.5 Build

The attached PDF plans (taken from the Fritzing Images) should detail the layouts for cabling and connections, between the various components. Included are plans for Sign-in Boxes and Device Access Boxes.

This document assumes that you already know how to solder, cable and mount components to build an electronic project. A list of recommended tools and ancillaries are listed in Appendix D but are not exhaustive.

1.6 Good Practice Recommendations (Device Access Boxes)

- Power Cables – put these entry points offset from centre, to allow space for other electronics to fit in. Typically leave 60 cms of cable in for the plug end and 30 cms of cable to the socket end.
- Swipe Sensors – Place these hard up against the top cover of the box, even a gap of 5-8mm will make it difficult to read. Also mark the external location of the sensor to direct the user where to swipe.
- Wemos D1 – always purchase the variant with the requirement to solder the connectors, and chose the shorter as these fit the connectors better (longer pre-made connectors can push the connectors out of the surround).
- Relays – always connect incoming Phase Power Wire to “COM” (Common) and outgoing Phase Wire to “NO” (Normally Open).
- Relays – Always tin the wires that go into the control sockets of the relay for screw-down attachments.
- Relays – Option “H” should be selected on the Relay Jumpers.
- Mounting brackets – Wooden standoffs for securing components are cheap and easy to make. Future releases of this tool may include 3d printing patterns for custom mounting brackets.
- Connections – Use a hobby glue gun for components you may wish to remove and replace. For more permanent connections, it is recommended to use Tarzan’s Grip glue.

1.7 Software Deploy

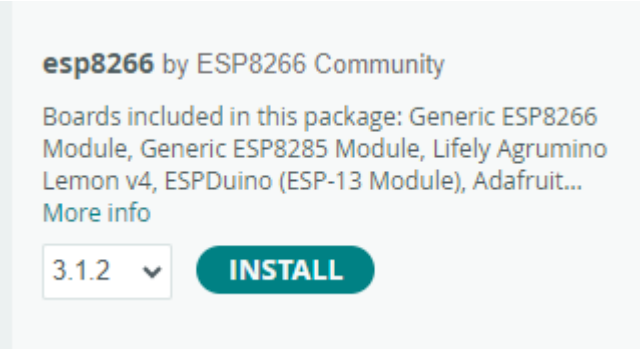
1.7.1 Arduino IDE Software and Drivers

Download and install the Arduino IDE.

For Board Drivers, start the Arduino IDE on an internet connected computer and navigate to “File” – “Preferences. In the field “Additional Board Manager URLs:” add the entry http://arduino.esp8266.com/stable/package_esp8266com_index.json and close with OK.

Navigate to “Tools” – “Board” - “Boards Manager” and search for “esp8266” in the text field. Select the “Install” button.

Photo 6 – ESP8266 Package



For Reader Libraries, select “Tools” – “Manage Libraries”. Enter the text “MFRC522” and select “Install” button.

Photo 7 – MFRC522 Package

1.7.2 Customising the Arduino Image

In order to connect to your system, you will need to specify some of your local parameters.

Start the Arduino IDE and open the Image you wish to deploy using “File” - “Open” and selecting the relevant “.ino” file. You will need to update the following parameters…

Field	Rationale
ssid[]	The identifier of your Wifi Router
password[]	The password of your Wifi Router
port	Should stay at 80

Photo 8 – network parameters

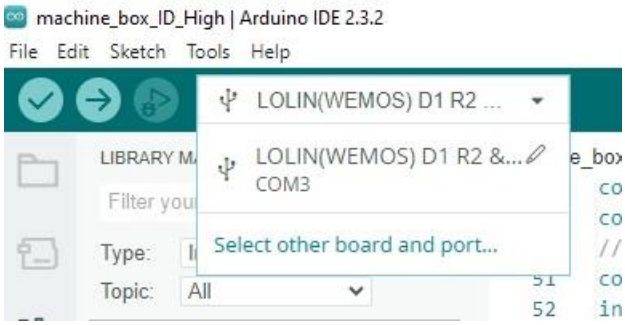
```
/* WiFi variables */
const char ssid[]     = "insert_ssid_here";
const char password[] = "password";
const uint16_t port = 80;
```

1.7.3 Arduino Device Setup and Image Deploying

Start the Arduino IDE and connect the Arduino Device via USB.

Drop down the Board Box and you should see a COM Port assigned to the Device.

Photo 9 – Board Box Selection



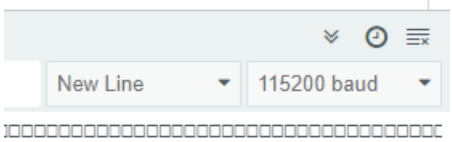
If it says “UNKNOWN”, click on the device and select Board Type of “LOLIN(WEMOS) D1 R2 & mini”.

To deploy the image to the Arduino, first open the image using “File” – “Open” and select the “.ino” file. Select the right arrow symbol on the toolbar to deploy the image (see above)

To monitor the output, the Serial Monitor needs to be open. Select the Magnifying Glass on the top right of the screen to open this. 

You will need to select the baud rate of the communications channel inside the Arduino, we recommend you use 115200. Otherwise you may see the response below.

Photo 10 – unprintable characters



You may need to perform a reset on the Arduino to see traffic (button next to USB connection).

1.7.4 Arduino Messages

The Arduino produces some messages not visible using normal operation but which can be viewed when connected to a Windows host with Arduino IDE installed.

Initial Setup

When starting up, the Arduino will alternately flash the LEDs until it can login to the Wifi. The messages below are a normal startup.

Photo 11 – normal startup

```
Connecting to [system]
.........
WiFi connected
IP address:
192.168.0.3
str_machine_wwn:083a8dd40959

Closing off machine 083a8dd40959
Setup complete for machine 083a8dd40959
```

Access Denied

When a swipe is detected, the Arduino will send a web call to the Central Server with the Tag ID being swiped and the WWN of the Arduino, and the server will return a code of 0 for access denied or 1 for access approved. If no signal is received in 2-3 seconds, it will be assumed that access is denied.

Photo 12 – Access Denied

```
PICC type: MIFARE 1KB
**Card Detected:**

Card UID: 53 39 2F 16
connecting to 192.168.0.105:80
sending data to server
/menzshed/api/machine/check_long.php?tag=53+39+2F+16&machine_wwn=8cce4ec37191
receiving from remote server
0string received:0:

closing connection

Machine Access Denied
```

Access Accepted

When access is granted, the response from the Server will look similar to the text as follows…

Photo 13 - Access Granted

```
PICC type: MIFARE 1KB
**Card Detected:**

Card UID: 53 39 2F 16
connecting to 192.168.0.105:80
sending data to server
/menzshed/api/machine/check_long.php?tag=53+39+2F+16&machine_wwn=8cce4ec37191
receiving from remote server
1string received:1:

closing connection

Access allowed
connecting to 192.168.0.105:80
sending data to server
/menzshed/api/machine/log_machine_long.php?tag=53+39+2F+16&machine_wwn=8cce4ec37191
receiving from remote server
1string received:1:

closing connection

Opening Relay....

Logged on
```

Appendix A – Sign-in / Sign-out Box Hardware Components Recommendation and Costs

Component	Description	Indicative Cost
Arduino Device	Wemos D1 Mini with ESP8266 chip	$9 (Trademe NZ)
Swipe Device	RFID RC522 sensor kit	$20 (Trademe NZ)
USB Cable*	2.0m Cable (check if Micro USB or USB-C, depends on model of Arduino)	$20 (most stores)
Enclosure	155mm x 115mm x 75mm	$40 (J.A. Russell NZ)
Resistors	2 x 150 Ohm resistor	$2 (Jaycar NZ)
LEDs	1 x Red, 1 x Green (5mm, 20mA)	$4 (Jaycar nz)
USB Power	5V 2A Mains Power Adaptor	$15 (PB Tech)

Appendix B – Device Access Box Hardware Components Recommendation and Costs

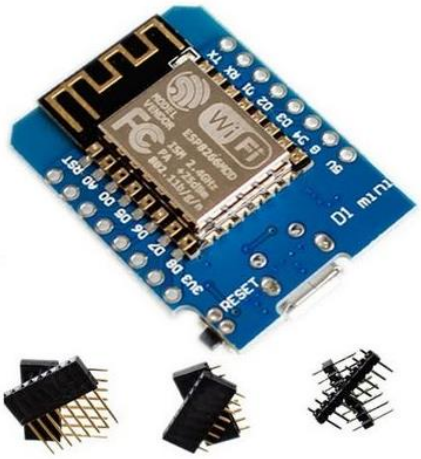
Component	Description	Indicative Cost
Arduino Device	Wemos D1 Mini with ESP8266 chip	$9 (Trademe NZ)
Swipe Device	RFID RC522 sensor kit	$20 (Trademe NZ)
USB Cable*	0.5m Cable (check if Micro USB or USB-C, depends on model of Arduino) recommend braided with small bend raduis	$20 (most stores)
Relay (3.3V 10A)	Capable of 240V AC, model BESTEP JQC3F-03VDC-C	$5 (epartners.co.nz)
Enclosure	155mm x 115mm x 75mm	$40 (J.A. Russell NZ)
Resistors	2 x 150ohm resistor	$2 (Jaycar NZ)
LEDs	1 x Red, 1 x Green (5mm, 20mA)	$4 (Jaycar nz)
Power Cable	2m extension cable	$4 (Bunnings NZ)
Enclosure (Power)	Arlec IP44 Outdoor Cord Safety Box	$4 (Bunnings NZ)
USB Power	5V 2A Mains Power Adaptor	$15 (PB Tech)
AC Plug Socket	3 pin 240V Female, For USB Power	$10 (Trademe NZ)

Appendix C – Ancillary Tools / Supplies Required (not complete)

Component	Description
Soldering Iron	
Soldering Station	With magnifying glass
Electrical Cable	
Heat Shrink	
Hookup Cable (24AWG)	
240V insulated connectors	
Wire Strippers	

Appendix D – Pictures of Common Parts

In order to prevent confusion when ordering online, the required images are shown for the required components.

Component	Image
WEMOS D1 Arduino Device	
3.3v 10A Relay	
RFID RC522	

Shorter Male Prongs (recommended)	
240V electrical connector