# Assessment type (🗹)

|  |  |
| --- | --- |
|  | Questioning (Oral/Written) |
|  | Practical Demonstration |
|  | 3rd Party Report |
|  | Other – Project/Portfolio (*please specify below)* |
|  |  |

# Version Details

|  |  |  |  |
| --- | --- | --- | --- |
| V | Date | Editor | Summary |
| 1 | 2024-07-11 | A Gould | New version using updated template |
|  |  |  |  |

**Note:** On the following page is a table of contents to assist you to navigate this document. You may CTRL+CLICK on an entry to jump to that location.

# Parts of the Document

**Assessment Instructions** These are the instructions that must be followed whilst completing the assessment.

**Assessment Instrument** This is where you may be asked questions, required to supply evidence of your work and other specific information as required.

**Appendices** Further required information that are required guidelines, but are better placed external to the work to be performed.

# Table of Contents

[Assessment type (🗹) 1](#_Toc175145527)

[Version Details 1](#_Toc175145528)

[Parts of the Document 1](#_Toc175145529)

[Table of Contents 2](#_Toc175145530)

[Assessment Resources 4](#_Toc175145531)

[College to Provide 4](#_Toc175145532)

[Student to Provide 4](#_Toc175145533)

[Assessment Instructions 5](#_Toc175145534)

[Date Due 5](#_Toc175145535)

[Scenario 5](#_Toc175145536)

[Information Referencing 6](#_Toc175145537)

[Before Commencing 6](#_Toc175145538)

[Step 1: Build Simulated Circuit 7](#_Toc175145539)

[Step 2: Code Discovery 8](#_Toc175145540)

[Step 3: Write Flasher Code 9](#_Toc175145541)

[Step 4: Write Functions 9](#_Toc175145542)

[Step 5: Code ON/OFF 10](#_Toc175145543)

[Step 6: Code Toggle 10](#_Toc175145544)

[Step 7: Update Circuit & Code 11](#_Toc175145545)

[Step 8: Physical Circuit 12](#_Toc175145546)

[Step 9: Code, Upload and Test on ESP32 13](#_Toc175145547)

[Step 10: Record Video Evidence 13](#_Toc175145548)

[Step 11: Submission 14](#_Toc175145549)

[Assessment Instrument 15](#_Toc175145550)

[Answer 1 Circuit Basics 15](#_Toc175145551)

[Answer 2 Sections 16](#_Toc175145552)

[Answer 3 Flasher 17](#_Toc175145553)

[Answer 4 Functions 17](#_Toc175145554)

[Answer 5 ON/OFF 17](#_Toc175145555)

[Answer 6 Toggle 17](#_Toc175145556)

[Answer 7 Two LED Toggle 18](#_Toc175145557)

[Answer 8 Physical Circuit 18](#_Toc175145558)

[Answer 9 Code, Upload and Test on ESP32 18](#_Toc175145559)

[Answer 10 Record Video Evidence 19](#_Toc175145560)

[Answer 11 Submission Requirements 19](#_Toc175145561)

[Appendix A: Assessment Submission 20](#_Toc175145562)

[Appendix B: Code Style and Commenting 21](#_Toc175145563)

[Appendix C: Referencing 25](#_Toc175145564)

[Appendix D: Video Recordings 26](#_Toc175145565)

|  |
| --- |
| Assessment ResourcesCollege to Provide  * Web server, Python interpreter and database server * IDE or editor for developing Python server/desktop/embedded programs (only PyCharm supported by the college) * IDE or editor for developing C/C++ embedded programs (Arduino IDE v2+ or CLion) * Arduino Uno, ESP32, Raspberry Pi Pico W, or Raspberry Pi with various sensors and actuators * Access to Office 365 & Microsoft Word * TinkerCAD Account (created by assessor/lecturer – if you do not have a classroom account, please use ScreenCraft Helpdesk to make the request)  Student to Provide Students may optionally obtain their own Electronics Kit to use with this and all other assessment items in this cluster.  Details are shown in the Blackboard shell. |

|  |
| --- |
| Assessment Instructions  * Please read these instructions carefully. * Follow each step as provided. * Questions will be asked in a separate section of this document, with space provided for your answers. * Information in the appendices **MUST** be applied to your assessment submissions. |
| Date Due  * 5PM on day of Session 7 |
| Scenario You are employed as a junior embedded systems developer for RIoT Systems (Robotics & Internet of Things), a Perth based educational and development company who specialise in IoT and Robotics systems.  You have been tasked with the implementation of a basic IoT Circuit and it’s controlling software.  You are provided with a set of steps to accomplish this (this document).  At any stage during this assignment item, you may consult the stakeholder(s) or their representative(s). |
| Information Referencing This is COMPULSORY for all assessments and covers resources that include but is not limited to:   * The Internet; * Books; * Video; * Code; * AI Use; and * Audio.   More details on referencing requirements may be found in Appendix C: Referencing. |
| Before Commencing Familiarise yourself with the content of this assessment by reading the whole document at least once before commencing.  As you progress through the steps contained in this assessment document, any questions relating to a step, or required evidence will be added into the Assessment Instrument section. |
| Step 1: Build Simulated Circuit Open the following link to create a new TinkerCAD account…  **2024/S2 ONLY** <https://www.tinkercad.com/joinclass/6TVBA3V6X>  Once you have created your account and logged in, click on **Classes**, and then click on the **InterRIoT YYYY-SN** link to open the classroom (YYYY is year, e.g. 2024; N is the semester, e.g. 2).  Next click on the **Portfolio Part 1** card to open the portfolio starter kit.  Click the **Copy and Tinker** button to copy the starter kit.  Rename the copy to be “YOUR NAME – Portfolio Part 1”.   * e.g. Adrian Gould – Portfolio Part 1.   You are provided with an Arduino Uno R3, an LED, a Pushbutton, a Resistor and a Breadboard.  Connect these parts up to match the circuit diagram below:    To assist, we are giving you the TinkerCAD component view:    Once the circuit is completed, take a screenshot of the top right of your browser and insert into the Answer 1 section (CTRL CLICK the text to jump to the location).  Answer the questions also in Answer 1 . |
| Step 2: Code Discovery Open the Code tab.  In here you will find the starter code for this portfolio.  Update the provided code to match the Code File Headers shown in Appendix B: Code Style and Commenting.  Study the code carefully, and answer the questions in Answer 2 Sections. |
| Step 3: Write Flasher Code Edit the code to make the LED in your circuit turn one for 1 second, then turn off for 0.5 seconds.  Copy and paste your final code into the provided space at Answer 3 Flasher. |
| Step 4: Write Functions You will now be updating your code to satisfy the following requirements:   * Modify the code so that you now have a function setLed(int ledPin, boolean state) that turns the LED connected to ledPin on or off depending on the value of state. * Modify the code to add two functions, ledOn(int ledPin) and ledOff(ledPin) that use the setLed function to turn the LED on and off respectively. * Make sure that you add comments to your code (See Appendix B: Code Style and Commenting)   Copy the code to the space provided at Answer 4 Functions. |
| Step 5: Code ON/OFF In the main loop, write the code to detect if the pushbutton is pressed down.  The code will perform the following actions:   * The LED will be OFF by default. * When the pushbutton is pressed, the LED should turn ON. * When the pushbutton is released, the LED should be OFF.   Make sure you have documented your code with appropriate comments (see appendix).  Copy your code and paste into the space provided at Answer 5 ON/OFF. |
| Step 6: Code Toggle You will now design and implement an update to your code so that the following requirements are satisfied:   * The LED is OFF by default. * When the pushbutton is pressed the LED will TOGGLE its state:   + If LED on then the LED turns off   + If LED is off then the LED turns on * The release of the pushbutton does NOT affect the state of the LED.   Copy your solution code and paste into the space provided in Answer 6 Toggle. |
| Step 7: Update Circuit & Code You are now to modify the circuit and code to perform the following actions:   * Add a second LED of a different colour, connecting it to PIN x * Add a second resistor for the new LED connecting it to the second pin on the LED and the ground * Write code to Toggle the LEDs, so that:   + LED 1 is OFF by default, LED 2 is ON by default   + When push button is pressed the LED states swap (LED 1 ON, LED2 OFF)   Submit a screenshot and copy and paste your updated code in the space provided at Answer 7 Two LED Toggle. |
| Step 8: Physical Circuit We are now going to transfer the tested and prototyped code and circuit to a physical device.  You will be using an ESP32 Development Kit for this purpose.  As this is a longer step, we will split it into sections. Unpack the ESP Dev Kit Keep the ESP32 plus its breakout board attached to the breadboard.  Carefully remove the chips, buzzers and other items and push into a piece of foam you will be supplied with (in class students).  Locate and put aside:   * Required number of the appropriate resistors * Required number of LEDs * Pushbutton switch. * Required number of jumper-leads. * USB cable.   All other items should remain in the box. Wire the Circuit Connect the components into the breadboard to reflect the layout you have designed in TinkerCAD.  **Before going any further, provide your circuit to the lecturer/assessor to verify wiring is correct.**  Document any issues (if any) that the lecturer/assessor found with your circuit in Answer 8 Physical Circuit. |
| Step 9: Code, Upload and Test on ESP32 You are now ready to power up the circuit and create the code to make it operate as expected.  Do the following steps:   * Plug the ESP Dev Kit circuit into a USB3 port on the PC. * Check the COM Port being connected to using the Device Manger. * Create a new Arduino IDE sketch ready for the code for the circuit. * Configure the Arduino IDE to use the required ESP32 board and COM Port. Make sure the COM port speed is 115,200 baud. * Rename the project to InterRIoT-AT2-POR-Pt1-XXX.ino where XXX are your initials. * Copy the code from the TinkerCAD Ide to the Arduino IDE. * Compile the code and upload to the device. * Verify that the LEDs work in the way expected (as per simulation). |
| Step 10: Record Video Evidence Record a short video of the circuit functioning.  **IMPORTANT:** Details of how to record and what is expected are shown in Appendix E: Video Recordings. |
| Step 11: Submission Make sure you follow the details in Appendix A: Assessment Submission and Answer 11 Submission Requirements when submitting your assessment.  For this assessment we require:   * This document with all questions answered, all required screenshots and any code that has been requested to be copied and pasted into the document. * A compressed copy of the Arduino IDE Project * A copy of the video evidence recording. |

|  |
| --- |
| Assessment Instrument When a step includes a question, you must attempt to answer it.  There is a minimum and maximum number of words to use for each answer.  If a step has more than one question, these maxima and minima are a total for all the questions in that specific step.  All answers must be in complete sentences unless indicated.  Unless otherwise directed, make sure to add any code you’ve written in a separate file to your submission. Also, unless otherwise directed, DO NOT put code in a Word document. Answer 1 Circuit BasicsScreenshot Delete the example shown below and insert a screenshot of the top left of your browser.   Question (Ohm’s Law) What is Ohm’s Law, and what is the equation it represents?  Ohm’s Law is a fundamental principle in electronics that describes the relationship between voltage, current, and resistance in an electrical circuit. V=I×R Question (Arduino/LED/Resistor) Given that the Arduino outputs 5V from its GPIO pins and an average LED uses 15mA to show at full brightness, what is the resistor value required for the circuit?  Provide your calculations for this answer.  **Given**:   * Voltage from Arduino GPIO pin Vsupply = 5V * Forward voltage of the LED VLED​ = 2V (typical value for a red LED, though this can vary slightly depending on the LED colour) * Desired current through the LED ILED = 15mA (0.015A)   Voltage drop across the resistor (Vresistor):  Vresistor = Vsupply – Vled = 5V – 2V = 3V  Then, use Ohm’s Law to find the resistance (R):  R = Ω Question (ESP32/LED/Resistor) Given that an ESP32 supplies 3.3V from its GPIO pins, and the average LED uses 15mA to show at full brightness, what is the resistor value you would use for the circuit?  **Given**:   * Voltage from ESP32 GPIO pin Vsupply= 3.3V * Forward voltage of the LED VLED= 2V (again, typical value) * Desired current through the LED ILED​ = 15mA (0.015A)   Voltage drop across the resistor (Vresistor):  Vresistor = Vsupply – Vled = 3.3V – 2V = 1.3V  Then, use Ohm’s Law to find the resistance (R):  R = Ω  I would choose a slightly higher resistor value (e.g., 100 ohms) to limit the current a bit more safely.  [LED Resistor Calculator (allmath.com)](https://www.allmath.com/led-resistor-calculator.php)  Allmath.com. (2024). *LED Resistor Calculator*. [online] Available at: https://www.allmath.com/led-resistor-calculator.php [Accessed 26 Aug. 2024]. Answer 2 Sections **Question:** What are the five sections in the example/base code within the starter circuit? (Remove the … from each numbered item)   1. Comment/Documentation Section 2. Include Statements 3. Pin Definitions and Global Variables 4. Setup Function 5. Loop Function   **Question:** In your own words, give a short explanation of the purpose for each section of code.  1: **Comment/Documentation Section**: This section documents the code, providing details about the project, author, and purpose. It helps anyone reading the code to understand its context and intention.  2: **Include Statements**: These lines bring in external libraries that provide additional functionality required by the code. For example, if you need to control a complex sensor, you would include its library here.  3: **Pin Definitions and Global Variables**: Here, the pins connected to components like LEDs and buttons are defined, along with any global variables that the program will use. This makes the code easier to understand and modify.  4: **Setup Function**: This function is crucial for initializing the hardware and setting up the initial conditions for the program. It configures the pin modes and any initial states or settings.  5: **Loop Function**: The loop function contains the core logic of the program, running repeatedly to check inputs (like button presses) and update outputs (like turning the LED on or off) based on those inputs. Answer 3 Flasher void setup() {  // Initialize the LED pin as an output  pinMode(LED\_BUILTIN, OUTPUT);  }  void loop() {  // Turn the LED on (HIGH is the voltage level)  digitalWrite(LED\_BUILTIN, HIGH);    // Wait for 1 second (1000 milliseconds)  delay(1000);    // Turn the LED off by making the voltage LOW  digitalWrite(LED\_BUILTIN, LOW);    // Wait for 0.5 seconds (500 milliseconds)  delay(500);  }   * pinMode(LED\_BUILTIN, OUTPUT) - This line in the setup() function configures the LED pin (often built-in on many Arduino boards) as an output pin. * digitalWrite(LED\_BUILTIN, HIGH) - This command sends a HIGH signal to the LED pin, turning the LED on. * delay(1000) - The program pauses here for 1000 milliseconds (1 second) while the LED is on. * digitalWrite(LED\_BUILTIN, LOW) - This command sends a LOW signal to the LED pin, turning the LED off. * delay(500) - The program pauses here for 500 milliseconds (0.5 seconds) while the LED is off.  Answer 4 Functions Copy your solution code and paste below this line.  void setup() {  // Initialize the LED pin as an output  pinMode(LED\_BUILTIN, OUTPUT);  }  // Function to set the LED state  void setLed(int ledPin, bool state) {  digitalWrite(ledPin, state ? HIGH : LOW);  }  // Function to turn the LED on  void ledOn(int ledPin) {  setLed(ledPin, true);  }  // Function to turn the LED off  void ledOff(int ledPin) {  setLed(ledPin, false);  }  void loop() {  // Turn the LED on  ledOn(LED\_BUILTIN);    // Wait for 1 second  delay(1000);    // Turn the LED off  ledOff(LED\_BUILTIN);    // Wait for 0.5 seconds  delay(500);  }   * setLed(int ledPin, bool state): This function takes two parameters: ledPin (the pin number where the LED is connected) and state (a boolean value). If state is true, the LED turns on; if false, it turns off. * ledOn(int ledPin) and ledOff(int ledPin): These functions make it easier to turn the LED on and off by calling setLed() with true or false, respectively. * Updated loop() Function: The loop() function now uses ledOn() and ledOff() to control the LED, making the code more modular and easier to read.  Answer 5 ON/OFF Copy your solution code and paste below this line.  const int ledPin = LED\_BUILTIN; // Built-in LED on pin 13  const int buttonPin = 2; // Pushbutton connected to digital pin 2  int buttonState = 0; // Variable to store the button state  void setup() {  // Initialize the LED pin as an output  pinMode(ledPin, OUTPUT);    // Initialize the pushbutton pin as an input with internal pull-up resistor  pinMode(buttonPin, INPUT\_PULLUP);  }  void loop() {  // Read the state of the pushbutton  buttonState = digitalRead(buttonPin);  // Check if the button is pressed  if (buttonState == LOW) { // Button pressed (LOW due to pull-up)  // Turn the LED on  digitalWrite(ledPin, HIGH);  } else {  // Turn the LED off  digitalWrite(ledPin, LOW);  }  }  **Setup Section:**   * pinMode(ledPin, OUTPUT) - This line tells the Arduino that the ledPin (which is connected to the LED) will be used to send signals out (to turn the LED on or off). * pinMode(buttonPin, INPUT\_PULLUP) - This line tells the Arduino that the buttonPin (which is connected to the pushbutton) will be used to read signals. The "pull-up" part keeps the button pin normally HIGH unless the button is pressed.   **Loop Section:**   * if (digitalRead(buttonPin) == LOW) - This checks if the button is pressed. When the button is pressed, the buttonPin reads LOW because it’s connected to ground (GND). * digitalWrite(ledPin, HIGH) - If the button is pressed (LOW), this line turns the LED on by sending a HIGH signal to ledPin. * digitalWrite(ledPin, LOW) - If the button is not pressed (HIGH), this line turns the LED off by sending a LOW signal to ledPin.  Answer 6 Toggle Copy your solution code and paste below this line.  const int ledPin = LED\_BUILTIN; // Built-in LED on pin 13  const int buttonPin = 2; // Pushbutton connected to digital pin 2  int ledState = LOW; // Variable to store the current state of the LED (starts OFF)  int buttonState; // Variable to store the current state of the button  int lastButtonState = HIGH; // Variable to store the previous state of the button  void setup() {  pinMode(ledPin, OUTPUT); // Set LED pin as output  pinMode(buttonPin, INPUT\_PULLUP); // Set button pin as input with internal pull-up resistor  digitalWrite(ledPin, ledState); // Ensure LED starts off  }  void loop() {  buttonState = digitalRead(buttonPin); // Read the current state of the button  // Check if the button has been pressed (state changed from HIGH to LOW)  if (buttonState == LOW && lastButtonState == HIGH) {  ledState = !ledState; // Toggle the LED state (HIGH to LOW or LOW to HIGH)  digitalWrite(ledPin, ledState); // Set the LED to the new state  delay(50); // Short delay for debouncing  }  lastButtonState = buttonState; // Update the previous button state  }  **Initial State:**   * ledState = LOW; sets the LED to be off by default when the Arduino starts. * The digitalWrite(ledPin, ledState); in setup() ensures that the LED starts in the off state.   **Toggling Mechanism:**   * The code reads the button's state continuously. * If the button state changes from not pressed (HIGH) to pressed (LOW), it toggles the LED state. * The LED state is flipped (on becomes off, and off becomes on) using ledState = !ledState;. * The new LED state is then applied using digitalWrite(ledPin, ledState);.  Answer 7 Two LED Toggle Screenshot your circuit showing the newly added LED, plus resistor.  Paste the image in the space below: Completed in session 2 (in class)    Copy and paste your code into this space provided:  int RED\_LED\_1 = 13;  int GREEN\_LED\_1 = 12;  int PB\_1 = 2;  bool redFlashing = true; // Start with the red LED flashing  bool buttonPressed = false; // To keep track of button state  void setup() {  pinMode(RED\_LED\_1, OUTPUT);  pinMode(GREEN\_LED\_1, OUTPUT);  pinMode(PB\_1, INPUT\_PULLUP); // initialize the button pin as an input with a pull-up resistor  }  void loop() {  // Read the state of the button  int buttonState = digitalRead(PB\_1);  // Check if the button is pressed  if (buttonState == LOW && !buttonPressed) {  buttonPressed = true; // Record that the button was pressed  redFlashing = !redFlashing; // Toggle the LED flashing state  } else if (buttonState == HIGH) {  buttonPressed = false; // Reset the button press record when the button is released  }  // Flash the appropriate LED  if (redFlashing) {  digitalWrite(RED\_LED\_1, HIGH);  digitalWrite(GREEN\_LED\_1, LOW);  } else {  digitalWrite(RED\_LED\_1, LOW);  digitalWrite(GREEN\_LED\_1, HIGH);  }    // Did 200 mill instead for faster switch - better effect    delay(200); // Wait for 200 milliseconds  digitalWrite(RED\_LED\_1, LOW);  digitalWrite(GREEN\_LED\_1, LOW);  delay(200); // Wait for 200 milliseconds  } Answer 8 Physical Circuit What issues did the assessor/lecturer find with your circuit?   * …   Insert photograph of your physical circuit below: Answer 9 Code, Upload and Test on ESP32 Copy your solution code and paste below this line. Answer 10 Record Video Evidence DO NOT PASTE ANYTHING INTO THIS SPACE Answer 11 Submission Requirements You are required to submit THREE items as separate items to your submission attempt.   * This document, renamed as outlined in APPENDIX… * A compressed file with your code for the ESP32 * Video of your demonstration |

|  |
| --- |
| Appendix A: Assessment Submission These assessment submission guidelines are common for all submissions in this cluster.  **DO NOT COMPRESS** any of the following when submitting:   * MS Office Documents (including Word, Excel and other files) * PDF Documents * Images (if less than three) * Video Recordings   **COMPRESS** the following:   * Project Code * Images if more than 3   Any single submission must contain all required components unless stated.  Submissions must be completed BEFORE 5PM on the date specified at the beginning of the assessment, unless otherwise indicated in this document. |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Appendix B: Code Style and CommentingCode File Headers At the start of EVERY file written in a C-style language (C, C++, C#, PHP, JavaScript, et al), the following block of comment is required, and must be completed with the appropriate information:  /\*\*  \* Assessment Title: Portfolio Part X  \* Cluster: Intermediate RIoT  \* Qualification: ICT50220 Diploma of Information Technology (Advanced Programming)  \* Name: YOUR NAME  \* Student ID: xxxxxxxxx  \* Year/Semester: 2024/S2  \*  \* YOUR SUMMARY OF PORTFOLIO ACTIVITY  \* GOES HERE  \*  \*  \* Components & Identifiers:  \* - Arduino Uno R3 UNO\_1  \* - Breadboard BBOARD\_1  \* - LED LED\_R\_1 [Red]  \* - Resistor RES\_1 [xxxxΩ]  \* - Pushbutton Switch PUSH\_1  \*  \*/ Code Style (Naming Conventions) For code written in a C-style language (C, C++, C#, PHP, JavaScript, et al), the following will be required… Case (Upper/Lower/Mixed)  |  |  |  | | --- | --- | --- | | Case | Use For… | Example | | Camel Case | Variables  Methods  Functions | ledState  toggleSwitch()  toggleLed() | | Pascal Case | Class names | class Led() { …} | | Snake case |  | bonus\_value | | Shouty/Angry Snake Case | Constants | LED\_1 |  Length  |  |  |  | | --- | --- | --- | | Use | Requirements | Example | | Variables, Constants, Methods, Functions, Class Names… | Minimum one word  NO abbreviations | ledState  LED\_1\_PIN  taxRate | |
| Code Style (Formatting) Code must be formatted consistently to facilitate ease of reading, debugging and collaboration. The following will be used as basic requirements:   |  |  |  | | --- | --- | --- | | Rule | Requirements | Example | | Indenting | Multiples of 2 or 4 spaces  Do not mix indent sizes | if (switchState == LOW) {  setLedOn(LED\_1);  } |  Code Documentation (All Comments) Commenting of code will depend on the requirements of the assessment or project. The following are good guidelines for good code documentation for use in your work:   | Rule | Requirements | Example | | --- | --- | --- | | Value | Comments must add value to the code | // Calculate the power using Ohm’s law | | Length | Lines should be less than 96 characters including prefixing symbols | // Determine taxation rate  /\*\*  \* Determine taxation rate  \*/ |  Code Documentation (Doc Blocks) Doc Blocks are used for commenting of methods, function, and classes. They are required to explain what the purpose is and how to use the item being described.   | Rule | Requirements | Example | | --- | --- | --- | | Doc Block | Provide summary details  Used for functions, methods and classes  Defines inputs and types  Defines output and types  Start with /\*\*  Each line starts with: \*  Last line: \*/  General Structure:   * First line after /\*\* is a one sentence short description * One blank line * Optional longer explanation with example usages * One blank line * Inputs * One blank line * Outputs   Inputs and Outputs are optional, so if the function/method does not contain these then the detail may be omitted. | /\*\*  \* LED On  \*  \* @input int ledPin  \*/ | |

|  |
| --- |
| Appendix C: Referencing You will be expected to use MyBib (https://mybib.com) to collate and create your references.  We DO NOT expect a university style references section with in text citations.  We DO expect to see any references to use APA 6 or APA 7 style  We DO expect to see references added after answers to questions.  For example:  Imagine that you're working on a project locally and bump into an exception. You try to figure out the problem, but you're unable to find a solution. In that case, you might want to ask a colleague for help.  *Introducing Laravel Error Share - Blog*. (2024, June 6). Flare. https://flareapp.io/blog/introducing-laravel-error-share |

|  |
| --- |
| Appendix D: Video Recordings Video recording may be a required part of many of your submissions.  You may be required to record whilst demonstrating components of the assessment.  The following list is a set of basic requirements for video recordings:   * Recording MUST be done in LANDSCAPE mode only (image is WIDE not tall). * The video MUST be recorded in a SINGLE take. * No editing permitted. * At the start you must:   + Show your face   + Verbally state the Cluster name   + Verbally state the Assessment title   + Verbally stating your name   + Verbally state your student number * When demonstrating you are expected to explain what you are showing. * At the end of demonstrating you are expected to state your name once more. |