# Assessment type (🗹)

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

# Version Details

|  |  |  |  |
| --- | --- | --- | --- |
| V | Date | Editor | Summary |
| 1 | 2024-07-12 | 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](#_Toc174533792)

[Version Details 1](#_Toc174533793)

[Parts of the Document 1](#_Toc174533794)

[Table of Contents 2](#_Toc174533795)

[Assessment Resources 4](#_Toc174533796)

[College to Provide 4](#_Toc174533797)

[Student to Provide 4](#_Toc174533798)

[Assessment Instructions 5](#_Toc174533799)

[Date Due 5](#_Toc174533800)

[Scenario 5](#_Toc174533801)

[Information Referencing 6](#_Toc174533802)

[Before Commencing 6](#_Toc174533803)

[Step 1: Wi-Fi Connectivity 7](#_Toc174533804)

[Step 2: MQTT Connectivity 10](#_Toc174533805)

[Step 3: Timeouts and Sleep 11](#_Toc174533806)

[Step 4: Sensors 12](#_Toc174533807)

[Step 5: Actuators 13](#_Toc174533808)

[Step 6: Record Video Evidence 14](#_Toc174533809)

[Step 7: Submission 14](#_Toc174533810)

[Assessment Instrument 15](#_Toc174533811)

[Answer 1 Circuit Basics 15](#_Toc174533812)

[Answer 2 Sections 16](#_Toc174533813)

[Answer 3 Flasher 17](#_Toc174533814)

[Answer 4 Functions 17](#_Toc174533815)

[Answer 5 ON/OFF 17](#_Toc174533816)

[Answer 6 Toggle 17](#_Toc174533817)

[Answer 7 Two LED Toggle 18](#_Toc174533818)

[Answer 8 Physical Circuit 18](#_Toc174533819)

[Answer 9 Code, Upload and Test on ESP32 18](#_Toc174533820)

[Answer 10 Record Video Evidence 19](#_Toc174533821)

[Answer 11 Submission Requirements 19](#_Toc174533822)

[Appendix A: Assessment Submission 20](#_Toc174533823)

[Appendix B: Code Style and Commenting 21](#_Toc174533824)

[Appendix C: Referencing 25](#_Toc174533825)

[Appendix D: Video Recordings 26](#_Toc174533826)

|  |
| --- |
| Assessment ResourcesCollege to Provide  * Web server, Python interpreter and database server * Access to the Internet (both Ethernet and Wi-Fi based) * Access to an MQTT broker * 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 6 |
| 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 Wirelessly connected IoT Device and it’s controlling software.  You are provided with a set of requirements 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: Wi-Fi Connectivity Create a new Arduino IDE project, renaming it to InterRIoT-AT2-POR-Pt2-XXX. Remember to replace XXX with your initials.  Download the .gitignore file you are provided with and copy to this new folder (C:\Users\YOURUSERNAME\Documents\Arduino\InterRIoT-AT2-POR-Pt2-XXX). Versioning the Development Open Windows Terminal in BASH mode   * For details on setting up Windows Terminal with the Git Bash CLI see [Add Git Bash to Microsoft Terminal](https://help.screencraft.net.au/hc/2680392001/65/add-git-bash-to-microsoft-terminal?category_id=35) (CTRL+CLICK to open link). * For details on adding useful aliases to the terminal see [Add or Update Bash Command Line Aliases for Git, MailPit and more](https://help.screencraft.net.au/hc/2680392001/66/add-bash-command-line-aliases-for-git?category_id=35) (CTRL+CLICK to open link).   Change into the Documents\Arduino\InterRIoT-AT2-POT-Pt2-XXX folder that was created.  cd /c/Users/YOUR\_USERNAME/Documents/Arduino/InterRIoT-AT2-POR-Pt2-XXX/  Initialise a Git repository  git init .  Add all the current files to the staging area  git add InterRIoT-AT2-POR-Pt2-XXX.ino  git add .gitignore  Commit the added files  git commit -m “init: Start of Portfolio Part 2”  Open your GitHub account.  Create a new repository with the following details:   * Name: InterRIoT-AT2-POR-Pt2-XXX (replace XXX with your initials) * Files: NIL! * Private: YES!     Add the remote to your local repository  git remote add origin https://github.com/USERNANE/InterRIoT-AT2-POR-Pt2-XXX.git  Push the changes to the remote  git push -u origin remote Create Code You will require TWO files, the .ino file and a secrets.h file where the details of the connection will be saved.  Create the code to connect the ESP32 to Wi-Fi with the following requirements:   * The Wi-Fi will attempt to reconnect every 10 seconds. * Indicate each attempt with a dot (.) on a line in the serial monitor * Every 25 dots, a new line must be started.   Make sure BOTH files contain the required documentation comments as per Code File Headers shown in Appendix B: Code Style and Commenting. Adding Updated/New Files to Versioning Add your updated and added code to version control  git add .  Commit the changes  git commit -m “feat: Wi-Fi Connection”  Push to remote  git push  In Answer section 1, add a screenshot of your repository on GitHub with the code uploaded for this step. |
| Step 2: MQTT Connectivity Extend your code to now connect to Adafruit’s Free MQTT service.  Alternatives are also available, but we will use Adafruit.   * [The Best Free Public MQTT Broker & MQTT Client by HiveMQ](https://www.hivemq.com/mqtt/public-mqtt-broker/)  MQTT Connection Requirements The code must complete the following only if a Wi-Fi connection is successful:   * Attempt to connect to the MQTT server * Display error messages as needed to serial monitor * Reattempts connection every 10 seconds. * Display connection success to serial monitor  MQTT Testing Test your device connects to the MQTT service. Adding Updated/New Files to Versioning Add your updated and added code to version control  git add .  Commit the changes  git commit -m “feat: MQTT Connection”  Push to remote  git push |
| Step 3: Timeouts and Sleep We are now going to implement the following features on the device:   * Device automatic sleep * Device automatic wake after 30 minutes * Maximum of 6 Wi-Fi connection attempts before device sleeps * Maximum of 10 MQTT connection attempts before device sleeps  Device Hibernation & Automatic Wake Create the code to sleep the device.  Create the code to wake the device once every 30 minutes. (For testing purposes, have the device wake every 30 seconds.)  Have device announce presence via MQTT.  Have device automatically sleep after announcing presence. MQTT Testing Test your code performs the sleep and wake actions successfully. Adding Updated/New Files to Versioning Add your updated and added code to version control with the commit message of: “feat: Device Sleep and Wake” |
| Step 4: Sensors We are now ready to add the next set of features to our IOT device.  Read a **Photoresistor** (sensor) and publish data onto the MQTT Topic “inter-riot-at2-data.  You must disconnect power before wiring the sensor to the ESP32.  The photoresistor will provide an approximate reading for the amount of light on the device.  Test the device by ensuring it is in total darkness and under bright light (e.g. phone torch).  Connect the device to your Adafruit account, so a graph may be generated by your data feed.  The device will check the sensor and send the reading each time it wakes up. Circuit Wire up the sensor to the ESP32.  Verify with the lecturer/assessor that wiring is correct. Write Code Write the code to read the sensor and send the data to the serial console.  Write the code to send the reading via MQTT message to the designated Topic. MQTT Testing Test your code performs the sleep and wake actions successfully. Adding Updated/New Files to Versioning Add your updated and added code to version control with the commit message of: “feat: Sensor Reading” |
| Step 5: Actuators Next feature for the IoT Device is the addition of actuators in the form of two LEDs.  One led will turn on when the device connects to Wi-Fi successfully.  The second led will flash when data is sent to the MQTT broker. Circuit Wire up the actuators to the ESP32.  Verify with the lecturer/assessor that wiring is correct. Write Code Write the code to turn the first LED on when Wi-Fi connection established.  Write the code to flash the second LED when the MQTT message is successful transmitted. MQTT Testing Test your code performs the sleep and wake actions successfully. Adding Updated/New Files to Versioning Add your updated and added code to version control with the commit message of: “feat: Actuator control” |
| Step 6: 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 7: 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 Setting Up EvidenceScreenshot Delete the example shown below and insert a screenshot of the top left of your browser.   Answer 2 MQTT **Question:** What is your Adafruit account name? Answer 3 Flasher **No evidence required** Answer 4 Sensors Screenshot: Take a screenshot of the sensor’s data being displayed on Adafruit.  Delete the example shown below and insert the new screenshot.     Answer 5 ON/OFF **No direct evidence required.** Answer 6 Video Evidence Record the device functioning as expected and submit as part of the uploads. Answer 7 Submission Requirements You are required to submit FOUR 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 * Link to your GitHUb Repository and a link to allow the lecturer to view your work. |

|  |
| --- |
| 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 singe 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 one more. |