|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Task:** | | **2** | | |
| **Task Title:** | | **Portfolio** | | |
| **Task Code:** | | **AT2 POR-Task-2** | | |
|  | |  | | |
| Assessment type (): | | | | |
|  | Questioning (Oral/Written) | |  | Portfolio |
|  | Practical Demonstration | |  | Project |
|  | 3rd Party Report | |  | Other – Please Specify |

|  |
| --- |
| The base requirements this assessment task include:   * Web server, Python interpreter and database server * IDE or editor for developing Python programs (only PyCharm supported by the college) * Raspberry Pi with SenseHat or other IoT devices, like Arduino Uno or ESP32 * Access to Office 365 & Microsoft Word * Report Template (Portfolio: Part 1 Document Template) as supplied   Use of some of these items may not occur in this part of the assessment task. |
| Assessment Due This assessment is split into components that have several due dates.  This section is is due:   * Week 09 17:00 (5:00PM) on the day of the scheduled lecture.   Refer to Blackboard for most accurate dates, which may alter due to unforeseen circumstances.  We also will endeavour to update these document(s) at the same time. |
| Instructions Follow the steps listed in this assessment item.  Submission of the documentation, code, and associated items is at the end of each part of the portfolio.  Each part of the portfolio has a deadline for submission.  It is advantageous to you to attempt to meet the deadline provided. |
| Important If you are using a different configuration of tools and equipment for this assessment item, then assistance in this and subsequent parts of the portfolio to ensure the systems work correctly will be limited. |
| Scenario You are currently working for a small Perth-based start-up company called Incredibly Obvious Technologies.  They are looking to create a presence in the home automation and monitoring market.  Download and read the complete “Portfolio-Task-Scenario” from Blackboard. |
| General Instructions Place all your answers directly into this document in the spaces marked Axx, for example A01, A02.  Rename the file as:   * XXX-C-INT-IOT-P-AT2-POR-Task-2.docx   Replacing the XXX with your initials.  For example, Adrian Gould would use AG-C-INT-IOT-P-AT2-POR-Task-2.docx for his submitted filename. |
| Answering Questions 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.  If required, make sure to add any code you’ve written in a separate file to your submission. DO NOT put code in a Word document. |
| Sources of Information In industry, it is good practice to keep track of where information was obtained. This is especially true if it is a written document, or even code.  If you answer any questions using information from web sites, please include the site name and URL (Web site address) after the answer. Likewise, include the title and author for books and magazine articles. For example:   * RS Electronics Ltd: <https://au.rs-online.com/> * Slack API Documentation, Users List Method: <https://api.slack.com/methods/users.list>   It is preferable for you to use APA6 or later for the references. This may be assisted by using MyBib or a similar web site to form the full bibliographic reference.  DO NOT put the references at the end of the document, but where required, include them with each answer. Code Storage We advise that you create a GIT repository on GitHub and use this to store a copy of your work.  You may also use OneDrive within your college Office365 to store a backup of your code or keep a copy on a USB thumb drive. |
| *This space left intentionally blank.* |

| STEP | Task to perform |
| --- | --- |
| 00 | Preparing for Work Make sure you have followed the instructions on creating the answer document, as given in the General Instructions.  Familiarise yourself with the content and document your progress in this assessment.  Make sure that you complete the title page of the document.  At any stage during this assignment, you may consult the stakeholder(s) or their representative(s). Note on Hardware Option for Task: Please note that this assessment may also be done on actual Arduino hardware provided you have the following components:   * Arduino Uno (with USB cable and access to the Arduino IDE) * LED (red) * Resistor (value T.B.D.) * Push button (NO – Normal Open) * Wires * Breadboard (optional – but makes life easier) * PIR Sensor (for the last question)   In this assessment, where Arduino Uno is shown, you may replace it with an equivalent device that is programmable using the Arduino version of C/C++.  For example, an ESP32, ESP8266 or similar. |
| 01 | Create a new TinkerCAD circuit In this Portfolio task, we’ll be creating a simple electronic circuit around the Arduino Uno.  Within TinkerCAD, you can place various electronic components, including complex components like the Arduino Uno, to create an electronic circuit. You can write code for the Arduino that will then be simulated.  We will be building a simple sensor board, starting with the basics, and extending it as we go along.  Create a new circuit in TinkerCAD.  Please invite your lecturer into the project.  Place an Arduino Uno in the circuit.  Add a push button, which we will use as a proxy for a simple open/close sensor.  Connect the push button to GND and to Pin 2 on the Arduino.  We will extend the circuit in following questions, so you may consider using a breadboard to place the components on.  You might have to rotate components for a snug fit. |
| A01 | Screenshot of Circuit Add a screenshot of the circuit that you have created. |
| 02 | Reading a sensor Having a (simulated) sensor is only useful if you can do something with it.  Download the file main.ino from Blackboard. It contains the following code:  void setup()  {  Serial.begin(9600);  pinMode(2, INPUT\_PULLUP);  }  void loop()  {  int sensorVal = digitalRead(2);  Serial.println(sensorVal);  delay(250); // Delay in ms  }  It configures the serial port and Pin 2 as input. The loop simply prints the value of the digital input (0 or 1).  Program the Arduino in your circuit with this code and run it (‘Start Simulation’). Open the Serial Monitor and observe the result. Press the button a few times and see what happens.  Put a screenshot of the code from your project and the output of the Serial Monitor in the spaces provided below. |
| A02a | Screenshot of Circuit Add a screenshot of the circuit that you have created. |
| A02b | Screenshot of Code |
| A02c | Screenshot of Output from Serial Monitor |
| A02d | Q: What happens if you press the button very briefly? When the button is not pressed, serial monitor says ‘1’ repeatedly,  When keeping the button pressed, serial monitor says ‘0’ repeatedly,  The more frequently I press the button, the more frequently serial monitor shows ‘0’, default value is still ‘1’. However, I found that the serial monitor output isn’t reflecting the speed at which I press the button. |
| A02e | Q: Explain why this happens. This occurs because I’ve set up digital pin 2 with the internal pull-up resistor. So when the button is not pressed, the pull-up resistor pulls the pin’s voltage up to the logic level HIGH(1) however, when the button is pressed, it connects the pin to ground which is logic level LOW or 0. In related to the delay code, it could be due to the way the code structured. The delay of 250 milliseconds(‘delay(250)’) between readings might not accurately capture the changes in button state. |
| A02f | Q: Explain how you may solve the problem you have identified. To solve this problem, I need to ensure that the serial monitor output reflects the button presses accurately.  I can put lower number for delay time or just delete it |
| 03 | Adding a simple output signal In this question, you’ll be adding an additional component to the circuit to show the state of the input sensor.  You should also add your solution from Question A02f to make the program a bit more responsive.  Add an LED to your circuit.  To limit the current drawn by the LED, you need to add a resistor.  Calculate the value of the resistor so the LED turns on but doesn’t get damaged. (You can change the value of the resistor by selecting it.)  Connect the LED and the resistor *in series* in your circuit:   * one end gets connected to GND, * the other end to Pin 13. |
| A03a | Screenshot of Circuit Add a screenshot of the circuit that you have created. |
| A03b | Q: What is the formula used to calculate the resistor value? V=IR R = V/I |
| A03c | Q: Show the calculation you performed to work out a suitable resistor value. Include your presumptions before showing the calculation.  Arduino supplies 3.3 volts and LED limits the current to 20mA  R = 3.3V/0.02A = 165 Ω |
| A03d | Q: Why does it matter how you connect the LED in the circuit? LEDs are polarized components which have two leads, an anode(positive) and a cathode(negative). It’s essential to connect the LED with the correct polarity to ensure it lights up. LEDS are current-driven devices, meaning they require a controlled current to operate. |
| 04 | Drive the output signal To drive the LED, we must update the code. We need to add two additional steps to the code:   1. Initialise Pin 13 as output. We only need to do this once. 2. Write a suitable value to Pin 13 to drive the LED. We must do this every time.   Using the existing code, add the correct statements at the correct places in the code to achieve the requirements that we just wrote down. Use pinMode(13, OUTPUT) to change Pin 13 to output mode and digitalWrite(13, X) to drive the output, where X equals either HIGH or LOW.  Simulate the code and ensure that the LED is ON when you press the button and OFF otherwise.  Add the current version of your code in a separate file: main-v4.ino. |
| A04a | Screenshot of Circuit (LED ON) Add a screenshot of the circuit in simulation mode with LED On. |
| A04b | Screenshot of Circuit (LED OFF) Add a screenshot of the circuit in simulation mode with LED Off. |
| 05 | A simple optimisation Your program keeps reporting the value of the sensor in the Serial Monitor no matter what. It would be useful to only have it report the value if it has changed.  For this purpose, add a global variable called previousSensorValue, which should be of type int.  Add code to your program the code that:   1. Initially sets previousSensorValue to a sensible value. Think about where in the code this should happen. 2. Checks the current sensor value against the previous sensor value and prints the current value if it has changed and updates previousSensorValue.   Simulate the code and ensure only updates are shown in the Serial Monitor.  Add a screenshot showing how you tested this and explain how the Serial Monitor’s output proves your implementation is correct.  Add a version of your code in a separate file: main-v5.ino. |
| A05a | Screenshot of Code |
| A05b | Screenshot(s) from Testing Code/Circuit |
| 06 | Make the LED show events only In your program, the LED just shows the state of the sensor: ON if the sensor is in the closed position, OFF otherwise.  We want to make the LED only light up if a change occurs on the sensor’s input, so analogous to the Serial port output, which only updates when the sensor value changes.  Create the code that will turn ON the LED for a brief period (but at least 300ms) if the sensor’s input value changes.  Ensure your code remains responsive to changes. In other words: you may not block the loop just to keep the LED ON.  Add one or more screenshots as evidence to show your program behaves as intended.  Add a version of your code in a separate file: main-v6.ino. |
| A06a | Screenshot of Code |
| A06b | Screenshot(s) from Testing Code/Circuit   Button pressed (LED ON)    Button pressed (LED OFF in 0.3 seconds)    Button not pressed (LED ON)    Button not pressed (LED OFF in 0.3 seconds) |
| 07 | Using an actual sensor Instead of using a simulated sensor, we will now use an actual sensor.  For this question, you will need to update your circuit to use a PIR Sensor.  A picture containing text, envelope, vector graphics, businesscard  Description automatically generated  Remove the push button from your circuit and add the PIR Sensor. Make sure to connect the power (Power, Ground) and the signal line (Signal). The latter can be connected to Pin 2.  The breadboard would come in handy now!  The PIR Sensor is powered whereas the previous simulated sensor was not. You can “activate” the PIR Sensor by moving the green dot inside the green area. (You have the select the PIR Sensor first.) |
| A07a | Q: Describe what the output signal of the PIR Sensor looks like. (Feel free to experiment to determine this.)  When the PIR sensor detects any motion in the green field, it turns on the red light in the internal display for 1 second and turn back off. When there is no motion detected, the light just stay off up until any motion detected. |
| A07b | Q: Explain how and why we need to change the input mode of Pin 2 to accommodate for the PIR Sensor.  PIR sensors provide a digital output signal indicating the presence or absence of motion. The sensor actively drives its output pin to either ‘HIGH’ or ‘LOW’ depending on whether motion is detected. Unlike a push button, a PIR sensor’s output pin actively drives its state, so there’s no need for a pull-up resistor to stabilize the pin’s state when the sensor is not actively driving it. I’ve switch from using ‘INPUT\_PULLUP’ mode for the push button to ‘INPUT’ mode for the PIR sensor. |
| A07c | Screenshot(s) of Code Update your code so that the Serial Monitor shows “Motion detected”, but only when motion is detected. You need to change the logic to achieve this.  Just like a real PIR motion sensor, you may update the LED to reflect the state of the PIR Sensor (optional).  Add a version of your code in a separate file: main-v7.ino.  Include a screenshot here. |
| End | Submission of Portfolio Work To submit the portfolio, do the following:   * Save the document with your answers as a MS Word file (.docx). * Compress your code into a single ZIP file and attach to your submission. * Open Blackboard, and locate the AT2 Portfolio Task 1 assessment. * Open the assessment and upload the original word-processed document. * Click submit.   Whilst there is no need to use any other word processing software as you have access to Office 365 for free as a student, if you use Apple Pages, or Open Office, we will then require you to upload the original file **AND** a PDF version. |

# Appendix A: Code Style Guidelines

The following guidelines should be applied to your code as it is developed.

Many may be applied via the use of PhpStorm, PyCharm or similar plugins and code formatting.

### PHP Code (General)

Please refer to the PHP PSRs:

* <https://www.php-fig.org/psr/psr-1/>
* <https://www.php-fig.org/psr/psr-12/>
* <https://www.php-fig.org/psr/psr-4/>

### Applications Built with Laravel

Please refer to the following articles:

* <https://dev.to/lathindu1/laravel-best-practice-coding-standards-part-01-304l>
* <https://dev.to/lathindu1/laravel-best-practice-coding-standards-part-02-a40>

### HTML Code

Please refer to the Google Style guide:

* <https://google.github.io/styleguide/htmlcssguide.html>

### JavaScript

Refer to the Google JS Style guide:

* <https://google.github.io/styleguide/jsguide.html>

### Python Code

Your code will follow the PEP 8 standard.

Other code standards available in the Presentation, “Python Coding Standards for North Metropolitan TAFE”.

### JSON Code

JSON should be formatted in an appropriate manner.