Component choice	
LDR	A resistor effected by light level; this was used to vary the voltage to an ADC pin on the Pico to allow me to convert changes in light into a integer value.
Transistor	This was chosen because a transistor would allow me to turn off and on the flow of current using an external voltage. This was used by the Pico as I would set a pin high, and the transistor would then open to allow the motor to start spinning.
Resistor	LDR resistor: this was used to change the sensitivity and range for the light level input this resistor needed to be big enough to not just max out the pins reading but also low enough that I can see some change in the light level, 3.3v with LDR(max) = 1000ohms would mean that with aw even split of voltage I would use a 1k to split it evenly when there is no light and the light level would go up with more light with 1.65v going to Pico when LDR is at max resistance Transistor Resistor: this component was used to stop the transistors over heating by limiting the current and voltage to the base making it more stable, 3.3v at pin with 2.6v to resistor and 0.7v to transistor base thus a current at base of 0.007A with a current gain of 100 from the 337 transistor 0.007/0.01 = 0.07A and the motor only needs about 0.5A so it will be enough of a damper.
Raspberry PI Pico W	This micro controller was chosen as it was a very powerful micro controller, capable of running python scripts but also being powered by USB and/or battery power of 3.3v-5v. with python scripts I'm able to get very in depth with what the robot can do, and python makes it easy to develop and debug. It also gave me a good amount of GPIO pins and ADC-GPIO pins this would be used with the Motor and LDR subsystems
Diode	A diode was chosen to be put before the VSYS pin on the Pico as when connected to USB power the VSYS would output 5v this would short the USB connection, so a diode was used to only allow voltage in not out. A power diode was used because it was able to handle the 5v load and not breakdown.
DC Motor	A DC motor was used because it gave me a way to convert electric energy to rotational kinetic energy to move the robot. As the Pico deals with DC current, I would not be able to use a AC motor thus a DC motor was chosen. It was a small DC motor to be able to fit in the robot body and was able to be driven by the 4.5/5v supply from the battery/USB power inputs

Test Method (Describe how you tested the circuit)	**Test Results (Describe what happened when you tested)**	**Debugging (Describe what you did to fix any problems)**
LDR subsystem building test: to test whether the LDR subsystem works, had one LDR and one 10k resistor and wired them up as my diagram asked	The light levels were detecting very low and not sensitive, with the levels in the 100 and not changing a lot with different floors	I decided to retest with a 1k resistor to see if it would give me a higher and more sensitive reading

LDR subsystem building retest: LDR light level detection was retested with a 1k resistor instead of the 10k	This gave me better results of the light being in the 3000-6000 range and was very sensitive with floor changes	No error was needed to be fixed
LDR subsystem test: only plugged in LDR subsystem to test light levels in certain places by hand	In natural light in class the light level read about 6000, with it dipping down to about 4000 when moved over something black, thus a difference of 2000 I would need to detect	No error was needed to be fixed
Motor subsystem building test: to test whether the motors could be controlled by the Pico and with a transistor I plugged one motor in with a BC547 NPN transistor as a switch and with 5v power from the Pico and with the base connected to a GPIO pin on the Pico	This worked to make the motor turn on but, sometimes the transistor would heat up and this didn't allow me to have the motor go backwards as I could flip the polarity of the motor	To fix the overheating issue I added a 330 resistor to the base of the transistor to limit the current and voltage to it. This fixed the issue with the motor still working too
Motor subsystem building test: to test the motor to go backwards I first needed to test whether I could have multiple transistors with one motor, I did this by having a transistor on either side of the motors connection with each transistor having a 330 resistor to the base and running to separate pins on the Pico	This worked and the motor turned on but sometimes the transistor would overheat and not conduct	A fix I found for the BC547 NPN transistor was to swap them for BC337 NPN transistors, these transistors didn't overheat and conducted very well with the small amount of current gain
Motor subsystem building test: to enable a polarity flip on the motors I would need to have 4 transistors per motor to test whether this would work I connected two transistors to one side of the motor with each being in a different orientation and connected to power and ground respectively, I did the same on the other side of the motor, with each transistor paired with a 330 resistor and each connected to a GPIO pin on the Pico	With the correct coding of each pin, the motor was able to flip polarity and go backwards, the only problem was that it was moving very slow, this could be from a voltage drop from the 2 active transistors or could be a gear ratio error	I was able to fix the speed issue by swapping to the 60:1 gear ratio instead of the 288:1 (Motor spin : Wheel spin)
Code test: to test the code I had written I just wired up both the LDR and Motor subsystem	The code worked perfectly with the only thing needing to be improved is the light level it checks for. Also, even though the code works It would	The light level checks were fixed by just changing them until I was happy, and the code got

	be better as functionally type code instead of object oriented	rewritten in a more functional way instead of object oriented