**Plans and Progress**

*Here you should give as much detail as you can about what your project will do, and how you will do it. This should also include*

*how far you have got with developing any features or outcomes from your project.*

NodeSpiders goal of transforming a standard mailbox, into a *Smart Mailbox,* requires the team to develop both a hardware component, needed to provide feedback and information to a device, and the software component, that will communicate with the device via an app on a mobile phone.

The hardware component is made up of a microcontroller, a temperature and humidity sensor, a switch that is triggered when mail is delivered and a battery pack.

Beginning with the microcontroller, the intention was to source a device that was low in power consumption, in addition to being small enough to fit discretely inside a mailbox, that could turn inputs into a useful action, such as giving notifications to an app.

The team discussed alternative options, such as a Rasberry Pi, however the use of a Rasberry Pi for this application created more complexities, the biggest issue was power consumption. After calculating the required power needed to run the unit, the final choice of going with a microcontroller was made.

Now we have a controller, the controller now needed a process to follow when receiving information from inputs given from the mailbox, inputs of new mail, temperature, and humidity. This required the microcontroller to execute these tasks via a program. To create a suitable and functional program, with limited experience in the team, the strategy was to keep the program simple at first, only taking a single input from a micro switch, that will replicate a mail delivery, get this functional, then implement a temperature and humidity sensor module into the program and controller. The program will be generated using the Arduino language in Arduino IDE, then flashed to the controller via the Arduino IDE.

Now with the plans of the controller and programming taken care of, next is configuring the controller with hardware that will produce and transmit the inputs. For R&D purposes and to save time, the controller and microswitch will be assembled and wired via a breadboard. This use of a breadboard with wire jumpers gave flexibility when adding and testing additional components and made changing pin-in/pinout configuration a straightforward process, as it eliminated the need for soldering. During the early stages of the hardware development, the microcontroller was programmed to take inputs from the switch and sensors, then the data from these was transmitted to a serial monitor. This confirmed the hardware as functional, in addition to the program being operational without any bugs.

Now with the hardware component was deemed to be operational, the data the controller is collecting needed to be sent to a useful destination the app could communicate with, rather than the data simply being displayed via a serial monitor. To achieve this, additional code in the controller sketch was required, data from the sensors was compiled in a Json file, and via a POST request, sent to the web server.

Testing the communication between the controller and web server was broken up into stages. Firstly, the temperature sensor was configured to make a POST request, with the required validation and Json file structure until a successful 201 response was received by the controller. During this process, several issues were encountered with achieving a 201 response. The first response that we were met with was a 401 response. This error was quickly identified to be caused by the layout of the authorisation token in the http header. After resolving the 401 issue we then encountered a 400 response. Investigation of this error found the Json file structure that was being sent, was in a form the webserver did not recognise. To resolve the issue, testing was carried out with a basic Json file, then when this was successful, the temperature sensor Json file was reverse engineered to finally identify the cause. After rectifying these issues, the POST request was met with a successful 201 response. Next, humidity and finally motion was implemented into the controller code to make POST requests of their own.

During the testing stage the decision was made to replace the micro-switch with a motion sensor. Originally, the microswitch was going to detect a mail delivery, however, after testing this in operation, the use of a microswitch to detect a mail delivery was inconsistent. A PIR sensor replaced the microswitch to improve the detection. Additional code to the program was added to allow the use of the PIR sensor.

With all three sensors now operational and successfully posting data to the web server, this concluded the mailbox hardware component of the project.