Energy Team: The PowerPuggle

3A Institute, November 2020

Caddie Brain
Dianna Gaetjens
Ashitha Ganapathy
Victor Crespo Rodriquez

TABLE OF CONTENTS

SUMMARY

Motivation

Climate change is one of the most challenging issues facing humankind today. To make our energy consumption sustainable for future generations and for the survival of the planet, there is broad scientific consensus regarding the need to transition to a zero-emissions global economy[1]. This will require fundamental changes to the supply of energy, which will need to be produced with lower emissions, predominantly through the use of renewable energy sources. Until all our energy generation comes from zero-emissions sources, the other way to reduce emissions is to simultaneously lower the demand for energy by reducing excess energy consumption.

The PowerPuggle is a cyber-physical system (CPS) that aims to do exactly that by encouraging users to reduce their energy consumption, thereby reducing emissions. In doing so, we didn't want to make people feel guilty about their energy habits - we envisioned a system that would engage and excite users about saving energy, and give them a platform to track and share achievements with their friends and broader community; in short, we aimed to design a system that was fun – meet the PowerPuggle!

The PowerPuggle

The PowerPuggle is an interactive energy meter that gamifies energy reduction. About the size of a box of playing cards, the PowerPuggle is a small physical device that you can carry around in your pocket. It's tactile, with buttons and a small colour touch screen.

The screen displays the PowerPuggle, which is a seriously cute baby platypus (the name for a baby platypus is a 'puggle'.) We selected the platypus because of its ability to detect electric fields through its bill, called 'bill sense'[2].

The buttons and touchscreen allow you to perform a number of actions to take care of your PowerPuggle. You need to feed it and give it water every day, or it will eventually get sick. Your PowerPuggle also needs the love and attention of tickles and the occasional party at fairly regular intervals in order to be as healthy and happy as it can possibly be.

So far so good - but where does the energy saving come in? The PowerPuggle device is connected wirelessly to a sensor/s that monitor energy consumption in your home[3]. On set-up, the sensor collects data on the household's energy consumption over seven days, developing a baseline. During this time, the PowerPuggle is an unhatched egg. Based on energy usage during the data collection period, our algorithm recommends an energy saving target. The user can accept this or select their own target. At this point, the PowerPuggle hatches.

The energy consumption sensor now begins to measure daily energy consumption. Daily consumption is displayed in the interface. For each day that the person meets or beats their energy saving target, they are rewarded with a congratulatory message and credits. If they beat their target for seven days in a row, or save a lot of energy, they receive bonus credits.

---

[1] Bruckner (2014), p.346

[2] Keyser (2018)

[3] For our prototype, we connected the PowerPuggle to a single device, a television. We chose a TV because – for most people – it's a non-essential electrical appliance. That means it is possible to reduce energy consumption (by reducing the time we spend watching TV) without a detrimental effect on daily life. Connecting the PowerPuggle to a fridge, on the other hand, would be far less effective as it's much more difficult to reduce the energy consumption of refrigeration. In addition, research we did Indicated that 82.6% of the Australian population watches broadcast TV on In-home TV sets each week. Thus, a device built around reducing TV consumption would be broadly applicable In the Australian market (Nielson 2018)

The credits earned are used to procure food, water and treats for the PowerPuggle. Its overall health is represented by a health bar of between one and seven hearts. Food and water need to be provided every day in order to maintain the PowerPuggle's health. In addition, as the user earns more credits, they can buy the PowerPuggle a party as a treat, or shower their PowerPuggle with love by giving it a tickle – which is free.

If the user fails to take care of their PowerPuggle (either through forgetfulness or because they don't reduce their energy consumption and therefore don't receive credits), the user receives a beep sound and a message, reminders that the PowerPuggle is hungry. If no action is taken, the puggle gets sick, and sustained energy saving is required to make it healthy again.

In order to fulfil our goal of having an impact on the environment by reducing, the PowerPuggle design needed to create change in the physical world (in this case, through changing the user's behaviour). Our design draws heavily on 'nudge theory' from behavioural economics to encourage the maximum possible change in user behaviour without being punitive or mandatory (see Theme for more details). Our idea was that the user would be motivated to take care of their PowerPuggle and reduce their energy consumption in the process because they develop an attachment to their PowerPuggle and want it to be healthy and happy.

For more information on how it works, see Design Requirements.

Intended audience

Our target audience for the PowerPuggle is any household consumer with an interest in saving energy, either in order to save money on bills or due to environmental concerns. Households with children might be particularly well-suited to using the PowerPuggle as parents could use it to teach kids good energy-saving practices and grow behavioural awareness. The PowerPuggle could be used by companies interested in saving energy too. Finally, we wanted to ensure that our audience was not limited to people from a particular background or country. Our scaled design includes many customisable options and accessibility features to ensure the PowerPuggle is culturally appropriate and relevant in contexts beyond Australia (see Scale).

THEME

Our assigned project theme was energy - and our CPS is directly related to this theme via its goal of reducing the energy consumption of users. The theoretical underpinnings of our concept (a fun energy-saving device that maintains user agency and changes behaviour via encouragement rather than directive) comes from behavioural economics, specifically 'nudge theory' and its application via the EAST framework to develop our design.

The first formulation of the term 'nudge' and associated principles was developed in cybernetics by James Wilk before 1995 and described by Brunel University academic D. J. Stewart as "the art of the nudge" (sometimes referred to as micro-nudges)[4]. It drew on methodological influences from clinical psychotherapy tracing back to Gregory Bateson, including contributions from Milton Erickson, Watzlawick, Weakland and Fisch, and Bill O'Hanlon. In this variant, the nudge is a microtargeted design geared towards a specific group of people, irrespective of the scale of intended intervention[5].

In 2008, nudge theory was popularised as part of behavioural economics by the publication of Richard Thaler and Cass Sunstein's book 'Nudge: Improving Decisions About Health, Wealth, and Happiness'[6]. Thaler and Sunstein define a nudge as 'any aspect of the choice architecture that alters people's behaviour in a predictable way without forbidding any options or significantly changing their economic incentives. To count

[4] Wilk, J (1999)
[5] O'Hanlon, B, Wilk, J (1987)
[6] Thaler R and Sunstein C (2008)

as a mere nudge, the intervention must be easy and cheap to avoid. Nudges are not mandates. Putting fruit at eye level counts as a nudge. Banning junk food does not.'[7]

In other words, a nudge is something that makes it more likely that an individual will make a particular choice, or behave in a particular way, by altering the environment so that automatic cognitive processes are triggered to favour the desired outcome. Techniques for nudging include gentle persuasion, changing the framing of choices, resetting default options or harnessing social influence.

Nudges are a good way to change behaviour because they:

- Help people live their values
- Often work better than simple awareness raising
- Are cost-effective
- Move people from motivation to action[8]

In addition, according to the Behavioural Insights Team, behavioural science shows that nudges towards everyday greener decisions are a powerful spur to environmental action[9].

For these reasons, nudge theory was an excellent fit for our prototype. We applied nudge theory to our design to move the user towards behaviour that conserves energy using the EAST framework, which says that if you want to encourage a behaviour, you need to make it Easy, Attractive, Social and Timely (EAST)[10].

- **Easy:** making it easy means harnessing the power of defaults as we have a tendency to go with the default or pre-set option reducing the hassle factor of taking up a service simplifying messages as making the message clear often results In a significant Increase In response rates to communications.
- **Attractive:** making it attractive means attracting attention using images, colour or personalisation designing rewards and sanctions for maximum effect
- **Social:** making it social means showing that most people perform the desired behaviour using the power of networks encouraging people to make a commitment to others
- **Timely:** making it timely means prompt people when they are likely to be most receptive consider the immediate costs and benefits as we are more Influenced by costs and benefits that take effect immediately rather than those delivered later help people to plan their response to events

Our PowerPuggle has been designed very carefully to maximise the chance of change to user behaviour (i.e. to maximise energy saving) by incorporating EAST principles in the following ways (Table 1)

Table 1: Application of the EAST Framework and nudge theory to the PowerPuggle design

EAST aspect	What it means	How the PowerPuggle design applies it
Easy	• Harness the power of defaults • Reduce the 'hassle factor' of taking up a service • Simplify messages	• PowerPuggle included with every smart meter (see Scale) • Setting up the system is simple – requires simply plugging the PowerPuggle in for 7 days to collect initial energy consumption data – no further set-up is required (see Algorithm) • Messages are simple and easy to action e.g. I'm hungry! Feed me! (See GUI)

[7] ibid, p6
[8] Behavioural Insights Team (BIT), *The Little Green Nudge*
[9] ibid
[10] BIT, EAST: Four simple ways to apply behavioural insights

Attractive	• Attract attention • Design rewards and sanctions for maximum effect.	• The PowerPuggle interface is cute and engaging, and uses images, colour and a buzzer to attract attention (See GUI) • The PowerPuggle relies on a reward and sanctions system implemented via credits (see Algorithm)
Social	• Show that most people perform the desired behaviour. • Use the power of networks. • Encourage people to make a commitment to others	• Our plan for scaling includes connecting social networks of users, thus utilising the power of networks to demonstrate that most people perform the desired behaviour, and encouraging people to make a commitment to others (see Scale)
Timely	• Prompt people when they are likely to be most receptive • Consider the immediate costs and benefits • Help people plan their response to events	• Our plan for scaling includes incorporation of smarter AI to push reminders to users when they will be most receptive (see Scale) • The PowerPuggle incorporates immediate costs and benefits to the user. If the user meets their energy target, they gain credits to take care of their PowerPuggle. If they don't, their PowerPuggle will not be cared for, and the user will be encouraged to rectify the situation via messages (I'm hungry! Feed me!) (see Algorithm) • The nudges received by the user include advice on a suitable course of action (I'm hungry! Feed me!) (see Algorithm)

DESIGN REQUIREMENTS

Summary

In creating the design for the PowerPuggle, we drew on the 3Ai research framework which seeks to understand CPS through the lens of Autonomy, Agency, Assurance, Intent, Interface and Indicators[11].

Initially, we carefully considered the *intent* of the system. Our intent was to create a nudge-based system that encourages user to use their *agency* to result in changes to their behaviour, thus influencing the physical, social and environmental worlds.

To translate this into the system intent, we designed three distinct, interacting components (see Figure 1) with specific system goals:

1. **A physical object:** A case with a touch screen and buttons that presents the user with an engaging *interface* encouraging active and ongoing engagement with the system over time. The *interface* consists of both the case and the Graphical User Interface (GUI). The GUI is also the vehicle for presenting the major *indicators* to the user – namely, their energy consumption, whether they are on track to meet their target, and how healthy their PowerPuggle is.
2. **Software:** The software consists of two components, the internet service and the code. The code was implemented in Python based on a design algorithm. The internet service reads the energy consumption every 5 minutes for seven days to produce the baseline energy consumption, which is one of the key *indicators* used by the system. Then, using an AI-powered algorithm, the service *autonomously* recommends an energy-saving target to the user. Once the user has accepted the target and proceeds to normal usage of the PowerPuggle (i.e. caring for it), the internet service reads the energy consumption every 5 minutes, calculates the 24-hour average, stores the data as required,

[11] 3A Institute 2020

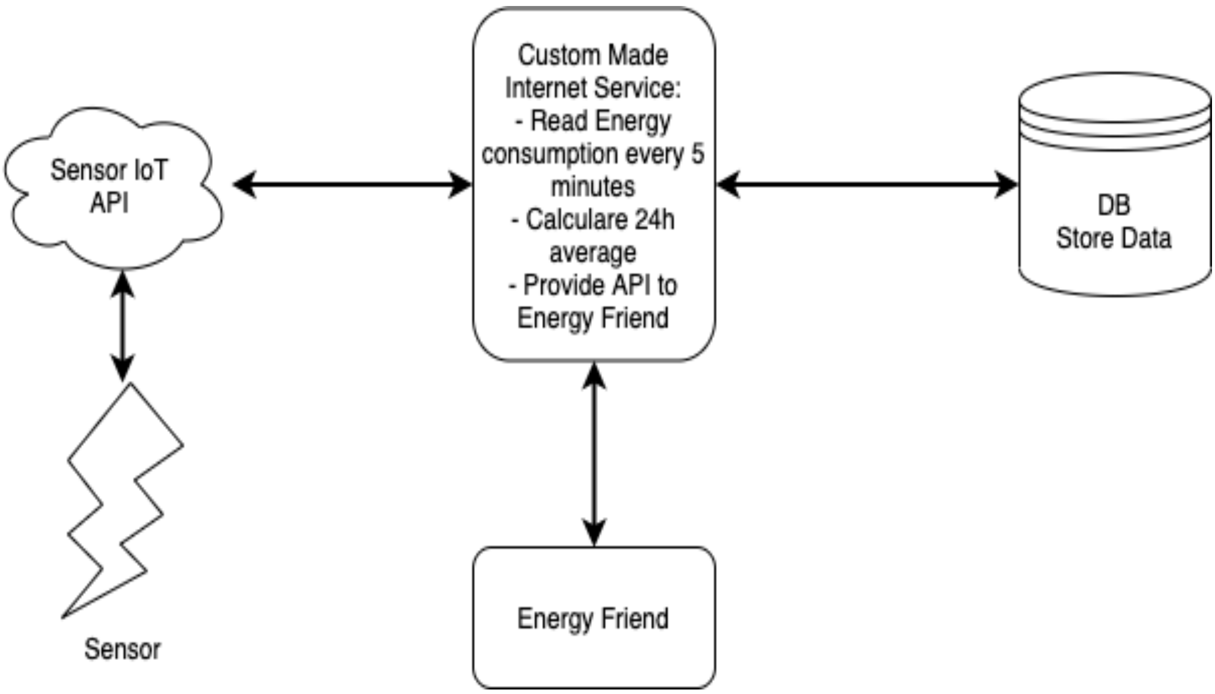
and provides an Application Programming Interface (API) for the PowerPuggle system. This incorporates several *autonomous* functions such as push notifications to the user (e.g. reminders to care for the PowerPuggle; congratulatory messages when targets are met etc).

3. **A sensor:** The goal of the sensor is to provide data from the device (or home). We modified an existing energy sensor for use in our prototype by setting up a virtual server and a private API to pull instant consumption data from the sensor, and store that information in a database. This enables us to pull the data from the private API as needed.

With respect to all three components above, and the system as a whole, we wanted to *assure* that everything functioned as intended, especially to make sure the system was safe to use (see CYBERNETIC ENGINEERINGsection for more detail on this aspect).

We'll now look at each of the three components in detail.

Figure 1. A system map showing the interactions of three main components of the PowerPuggle System.



Physical Components

Materials

- Ultimaker 2+ printer
- Printer filament, various colours
- Fusion360 (3D modelling software) (see below for why we selected this software)
- Raspberry Pi 3B+ (we used this as they were distributed to us by 3Ai at the beginning of semester)
- PiTFT Plus Assembled 320x240 2.8 TFT + Resistive Touchscreen (SKU: ADA2298) (We chose this because it was the same size as the Raspberry Pi 3B+. We wanted to keep the PowerPuggle pocket-sized so that people could take it with them, and thus didn't want a larger screen.)
- Faceplate and Buttons Pack for 2.8 PiTFTs - Raspberry Pi B+ / Pi 2 (SKU: ADA2807) (we only used the buttons but you can't buy them separately)
- Polymer Lithium Ion Battery (LiPo) 3.7V 1100mAh (SKU: CE04377) (We chose this battery mainly due to its physical size, which is small enough to fit inside the rest of the components.
- PowerBoost 500 Basic - 5V USB Boost @ 500mA from 1.8V+ (SKU: ADA1903)
- Wires to connect the PowerBoost to the battery

- Solder and soldering station to solder the PowerBoost to the battery
- On/off switch
- Buzzer
- Procreate
- Graphics (as drawn by Caddie)
- Platypus image (licensed from Canstock)

Case

The components of the PowerPuggle are housed within a case. The case itself forms the initial *interface* to the user, featuring buttons and a touchscreen. With that in mind, our design goals for the case included developing something:

- tactile, pleasant to touch and visually pleasing
- small enough to carry around
- robust enough to withstand being dropped
- physically safe and suitable for a child to use (no sharp edges etc)
- the right size to fit all the internal components e.g. cut-outs for the screen and for the buttons.

The first key decision point was deciding what process to use to create the case. We considered both laser cutting and 3D printing, as both are relatively fast and low-cost prototyping tools, and we have access to both on campus. We also considered whether we could purchase a ready-made case. We eliminated purchasing a case quickly because after researching available cases, we realised that none would meet our requirements, specifically the one around being the right size to fit the components in, with cut-outs for the screen and buttons. That left us with laser cutting and 3D printing as options.

Initially, we favoured laser cutting because we could use a sustainable material (i.e. wood) for the prototype (this was desirable in terms of our sustainability goals and for the decommissioning of the systems, see Decommissioning). However, after consulting the MakerSpace staff on the possibility of laser cutting wood, we discovered that the only laser cutter on campus that can cut wood is in the School of Art and Design, which remains closed due to COVID-19. With no guarantee that it would open in time, we decided to 3D print the case instead. The trade-off here was that 3D printing uses PLA filament, which is far less sustainable than wood, and less in keeping with our environmental conservation goals for the project. However, we decided this was an acceptable compromise for the prototype, and that for production at scale, we would investigate sustainable options.

After deciding on 3D printing, the next key decision we had to make was about how to get a 3D model for the case. We employed a systematic approach to do so consisting of the following steps:

1. Research and selection of tools
2. Finding resources and information to help
3. Following a clear set of instructions
4. Prototyping
5. Trouble-shooting failure
6. Final product

With our design goals in mind, we researched various options, including using pre-existing case models on *Thingiverse* and adapting a faceplate we purchased (see Materials list). However, none of these options would have satisfied the unique requirements we had, specifically with respect to integration with all internal components (cut-outs for a screen and buttons).

Based on that, we decided to model and print a case from scratch. The trade-off here was that while we knew that modelling the case from scratch would allow us to meet all our design specifications, we also knew that it would take a lot longer, and would be a significant challenge as the lead for this element, Dianna, had only basic experience with 3D modelling. However, given our sound project planning, we came to this decision early in the process, which left plenty of time for Dianna to research and develop the required proficiency. Thus, we decided that this was an acceptable trade-off to make.

The first step in modelling the case was selecting modelling software to use. Dianna had previously used *Tinkercad* to model a simple object, but had found it difficult to use. Thus, we decided to look for a different modelling software. Using an article on the strengths and weaknesses of different modelling software[12], and also Victor's advice, we selected *Fusion360* as an appropriate tool. The article ranks *Fusion360* as the best all-round 3D modelling software because of its power, the quality of its simulation and the control the user has over shape designs. Victor had also used it before and recommended it.

With that decision made, we looked for a tutorial to help with modelling the case. We found a particularly clear video tutorial on building a Raspberry Pi case in *Fusion360*[13]. This contained step-by-step instructions for modelling a case. Comprehensive instructions can be obtained by watching the video, however a summary of steps involved follows.

To help model the case, the tutorial suggested first importing an existing model of a Raspberry Pi from *GrabCAD*[14]. Having the Raspberry Pi model as a base to model the case around ensured the dimensions of the case and positioning of cut-outs etc. would be correct.

The model started as a rectangle. This was then extruded into a 3D rectangular prism and the inside shelled. The edges and corners were filleted (which means rounded – this is aesthetically pleasing and also safe, as it removes sharp edges). Pegs were added to the inside of the bottom half to hold the internal components in place. Filleted cut-outs were added for the connectors we needed (the micro-SD slot and charging cable slot) using the slot creation tool. Then the case was split in half using construction tools, including the split body tool. Then the cantilever joints were designed, paying close attention to fitting them in around the other components that would be inside the case. This involved positioning the joints (using construction lines), using the sketch tool to draw the joint shape, extruding the joints, adding chamfers (a type of bevel), and filleting all edges (in this case, filleting helped to increase the strength of the joint). Finally, we edited the join between the top and bottom halves of the case, adding a groove to help the case fit smoothly together.

At this point, the tutorial ended, but we still had to create cut-outs for the screen and buttons. We also had to edit the dimensions of the case to make sure it would fit around all of the components we planned on having inside. With the skills already learned in the tutorial, and by measuring the components we planned to put inside the case (i.e. the screen, the Pi, the battery and the Power Boost board), we were able to complete these steps[15].

[12] https://www.adamenfroy.com/3d-printing-software

[13] Product Design Online 2019 https://www.youtube.com/watch?v=E0NVC8xhf3I

[14] https://grabcad.com/library/raspberry-pi-4-model-b-1

[15] Note that the process above included employing other features of Fusion360 including: creating and saving files; inserting objects into an existing file; changing orientation of objects within Fusion360; positioning objects on the workplane; selecting origin planes; choosing dimensions; choosing opacity settings to enable multiple components sitting inside each other can be viewed at the same time; using the section analysis tool to cut away sections and check whether selected dimensions are appropriate; using the project command to project geometry from on component into another; and considering appropriate tolerance levels.

We then moved to printing the case. Our first print resulted in failure as the print plate was not calibrated correctly (see Photo 1). This resulted in strings of filament coming off the case. Our second print also resulted in failure as one corner warped (which is where part of the print lifts up) (see Photo 2). Overcoming both these challenges involved careful consultation with MakerSpace staff about the reasons for the failures and ways to correct for them. Dianna learnt to calibrate the plate, mitigating the first issue. She also investigated warping as part of her Build Learning Portfolio, and developed strategies to minimise the risk of it re-occurring. The main one was using a build plate adhesive and also printing using a brim.

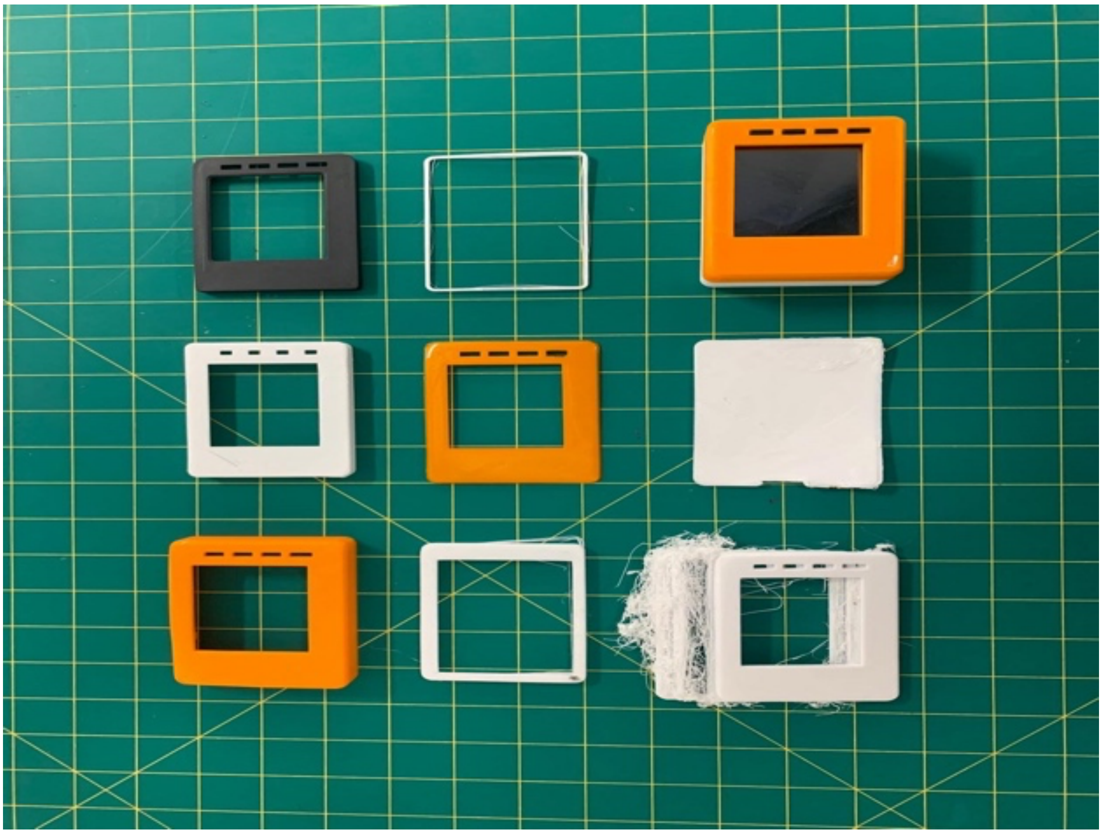


Photo 1: Our various failed prints. The stringy version is the top right, the warped version is the bottom right.

Photo 2. Warping up close - the warp is the bent-up corner in the top right of the photo.

Even though early prints failed, we used them to check dimensions and positions, and were able to tweak the model so that each re-print was improved. The final print was successful – we have a perfect print (i.e. with no warping), with cut-outs in the right places and of the right size.

Components inside the case

The case houses a Raspberry Pi 3B+, a resistive touchscreen, a switch, a buzzer, a battery and a booster board (to provide additional voltage to the Raspberry Pi as the battery alone did not meet the voltage requirements of the Raspberry Pi).

Our second challenge was connecting all the components together within the space available in the case. Caddie took the lead in this process, using a systematic process involving experimentation, prototyping, assembling and iterating.

First of all, she prototyped the placement of the various components to make sure that everything would fit. This involved trial and error. The placement we eventually decided on can be seen in Photos 3 and 4 below. It's important to note here that one trade-off we had already decided on was with respect to the size and power of the battery. In general, a more powerful battery would be better for the PowerPuggle as it lasts for longer (which is important because we expect users to be carrying the PowerPuggle around with them) and requires less charging. However, we also knew that the PowerPuggle needed to be pocket-sized (again, because we wanted to design it to be carried). Thus, we wanted all the components to fit together in a space no larger than the dimensions of the clicked-together screen and the Raspberry Pi. When we went to source the battery, we were therefore limited to choosing one that was physically small enough to fit in the space. This was not the most powerful battery. In other words, we traded off power for size. We are not yet sure how long the battery will last with normal use of the PowerPuggle as we did not have time to test this. An important next step for our design would be making sure that the power trade-off we made is acceptable in terms of how long the device can stay powered up.

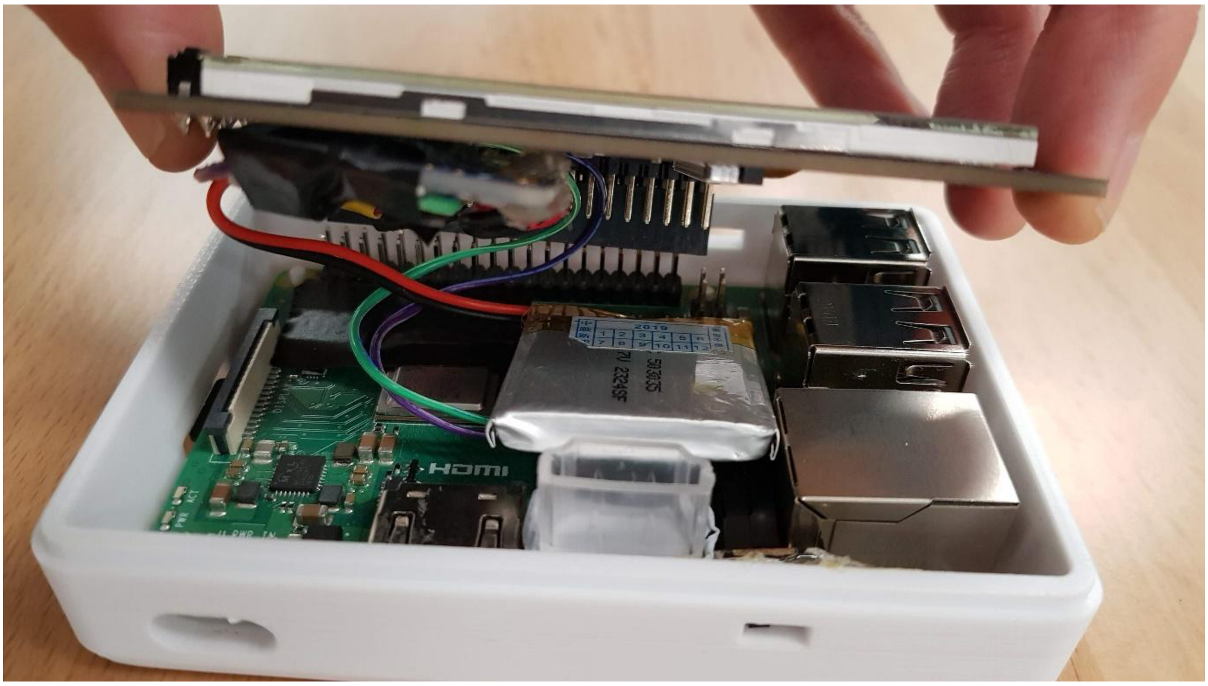


Photo 3. Placement of all the internal components

Photo 4. Internal components of the PowerPuggle

The next part in assembling the components involved fixing all the components together. The screen was designed by Adafruit to click onto the Raspberry Pi 3B+, so we did not have to use any other mode of connection for this part. We connected the battery to the PowerBoost using a JST connector. The next step was connecting the PowerBoost board to the Raspberry Pi. To fix the connection in place, we experimented with using the existing pins on the PowerBoost board and conduit. However, this didn't work (the leads wouldn't stay put,) so we decided to solder them in place. First, Caddie prototyped the circuit to ensure that all the connections were in the right place. Then, under Victor's instruction (as it was the first time she had soldered), she soldered the two components together. After soldering cables to the PowerBoost board, we plugged the battery and the PowerBoost into the Raspberry Pi, and it turned on.

At this point, we believed that the internal components were complete. However, we realised once we started integrating the code and the physical device that we had not included a way to turn the PowerPuggle on and off. We hadn't thought about this when assembling the components because at that point, the code wasn't finished and so we hadn't needed to try and turn it on! We decided to iterate on the component assembly to try and include a switch. We found a suitable switch in the lab at 3Ai, and then used a hot glue gun to attach it to the case. Caddie soldered the pins on the switch to the Enable Pin in the PowerBoost, thus giving us a way to turn the device on and off (admittedly, the switch doesn't actually protrude through the case as it's too short, and so you need tweezers to turn it on or off! We could easily fix this in a future iteration with a slightly bigger switch).

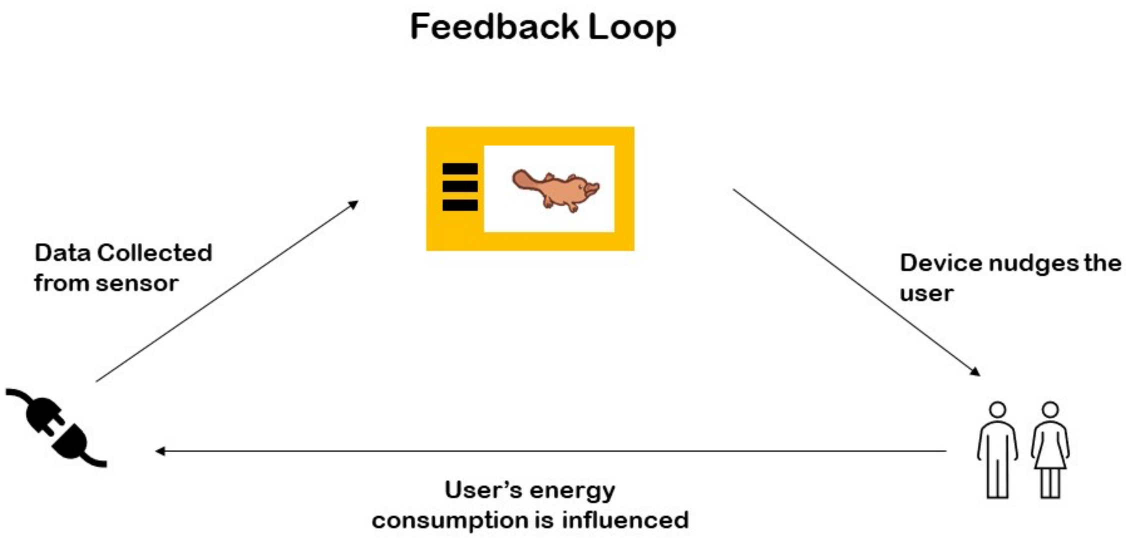
We iterated our design in one other way. We realised once we had drafted the algorithm that we were not embodying best practice with nudge theory as there was no way for the device to remind the user to engage other than through the screen – and if the user is looking at the screen, they are already engaging! So we decided to add another component that could more actively remind the user to engage. We did this by adding a buzzer (in a future iteration, we might replace this with a vibration alert or similar.) The buzzer was integrated with the existing algorithm and is used in conjunction with a message saying *'I'm hungry! Feed me!'* to nudge the user to engage again if they haven't for the last 24 hours.

Graphical User Interface (GUI)

The GUI is the other part of the interface (apart from the case) that presents to the user. As our PowerPuggle uses nudges to change behaviour (rather than directives), the interface is the main vehicle through which our

CPS changes the behaviour of the user in the physical world. In addition, it's the **main feedback loop** in our system (see Figure 2 below). Given this, we knew an accessible and engaging GUI would be critical to the success of our project and so spent considerable effort in designing it.

Figure 2. Feedback Loop: interaction with the physical world



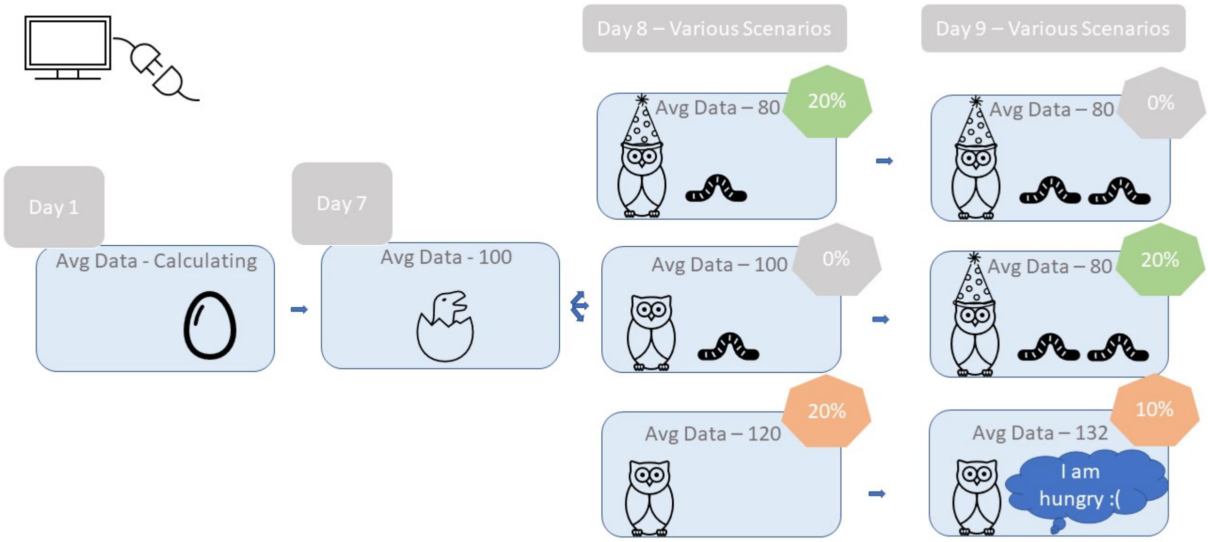
We used the algorithm we had developed for the code (see Software section) to map out elements we would need for the GUI. After several iterations, and keeping in mind the principles of nudge theory and the EAST framework, we decided on a GUI with the following goals:

- simple, easy to read and understand
- engaging for the user
- showing *indicators* including:
    - the user's real-time power consumption
    - the 'condition' of the Puggle (see below)
    - the actions available to the user
    - the credits available to the user to spend
- able to nudge the user to take action
- able to show immediate costs and benefits of action

Through the process of developing the algorithm, we had decided that the PowerPuggle would have ten conditions (egg, hatching, normal and the seven states of health) and four actions (feeding, drinking, tickling and party) (see the Software section for more detail on how these decisions were made). The main challenge for developing the GUI was thus making sure that the interface could represent all these conditions and actions, as well as including the necessary indicators, clearly and easily in a way that encouraged user engagement. We also considered an interface that would appeal to both adults and children. In short, we needed to avoid an interface that was in any way confusing, complicated or hard to use.

Our systematic approach for overcoming this challenge involved experimentation and iteration. The first steps in designing the GUI were deciding how to lay it out on the screen, and how we were going to represent all the indicators we needed to display. Caddie took the lead in this part of the design. The trade-off we faced here was between too much and too little information. We definitely wanted the three key indicators – PowerPuggle health, number of credits, and energy consumption of the user – to be at the top of the GUI so that users had easy access to this information all the time. At the same time, space at the top was limited, and we didn't want the GUI to look crowded. So we decided to use icons and numbers to minimise the space requirements while still facilitating ease of use. We represented the health states (1-7) using a heart bar, in line with how other computer games represent health. Power consumption and number of credits are represented with a number.

Figure 4. Early iteration of the GUI



Data – Live energy consumption data from Sensor
Average Data – 7 days average energy consumption for specific user
Benchmarks by default are set based on the user's previous energy consumption and keep adjusting based on the week that has just passed. However the user has an option to set a target below the default

With respect to the actions available, we knew that these would be accessed through the buttons on the right-hand side of the screen, so placing text with the name of each action along the right-hand side of the screen was a relatively straightforward decision.

That left decisions about how to represent the Puggle itself and the actions the user could choose. Our primary goal was to make sure that we encouraged engagement, so we knew we needed graphics that would keep the user coming back.

With respect to the form of the PowerPuggle itself, we considered many different options. For example, an early front-runner was the slow loris (mainly because they are adorable). We also considered making the PowerPuggle a plant, but dropped this idea as a plant is less engaging. Ultimately, we decided that for the prototype, we wanted an Australian animal that wasn't domesticated (in order to emphasise the project's connection with the environment). We chose the platypus because in addition to fulfilling these criteria, it is on the vulnerable species list. Also, we discovered that its 'bill sense' - a special ability to detect electric fields through its bill. This made it an especially appropriate choice for our project, which is all about electricity!

With that decision made, Caddie led on designing appropriate graphics. We decided that the best way to do this was to use GIFs (we also considered using still images, but decided that a moving picture would engage the user more – see also the trade-off that was caused by using GIFs in terms of programming the GUI using tkinter in the Software section). We researched simple ways of producing GIFs, and discovered Procreate, which creates a GIF from uploaded still images. Using a cute platypus image we found (and licensed) on Canstock as inspiration, Caddie drew between six and twelve stills for each action and then stitched them together into GIFs using Procreate. She also drew stills to create a 'normal' state for the PowerPuggle (which shows it swimming) and a background for all the graphics to sit on top of (water).

The final part of the GUI was incorporating nudges. We did this in the prototype by having reminders for the user if they don't engage for 24 hours, we designed the system to display a message saying 'Feed me! I'm hungry!' This is linked to a buzzer that makes a noise to alert the user. We also incorporated a message congratulating the user for meeting their energy target every day, and a further congratulatory message for the user if they meet their energy targets for 7 days in a row.

With the graphics finished and a plan for how the GUI would look, these elements were then implemented as the UI via the code (see Software).

Software

The software includes two components, namely the internet service for requesting and storing energy consumption data, and the code for driving the system. We started the process of building these components by designing an algorithm incorporating all the important aspects of the PowerPuggle.

Materials

- Python
- Ruby on Rails
- Jupyter Notebooks
- Amazon Web Service free tier virtual server
- Sensor IoT Application Programming Interface (API) access
- Heroku free dyno server
- SQLite (database)
- PyQT5 (GUI framework)

Algorithm

The goals we had for the algorithm were:

- Simple and clear, both to facilitate ease of programming and also to make the user experience simple and easy
- Non-binary: to include various ways for the user to be able to engage with the PowerPuggle
- Realistic energy saving target to encourage saving without deterring users
- Interesting enough to keep user engagement high without introducing too much complexity
- Using the EAST framework, we knew we had to incorporate sanctions and rewards in a way that made users feel immediate benefits and costs and also made users keep coming back (i.e. making sure the barrier to earning rewards was not too high).

We also knew that the algorithm was the vehicle through which key decisions with respect to how the system would work and respond to user engagement. The main decisions related to: the number of conditions we would have for the PowerPuggle (this refers to the different states the PowerPuggle can be In); the number of actions available to the user; the system by which the user earned credits; and how to set the energy saving target.

Ash took the lead in designing the algorithm (see below Figure 3). We did several iterations (unfortunately, we didn't keep photos of the first versions). Each iteration created new questions and forced us to think carefully about how to maximise user engagement given the trade-offs inherent in our goals (for example, between being complicated enough to keep users engaged but simple enough to understand easily).

Our decision points around the number of conditions our PowerPuggle could be in; the actions available to the user; and the credit earning system were interlinked, so we developed these together. Our process involved a lot of trial and error, brainstorming and experimenting. At the end of this, we had come up with one model for number of conditions, number of actions and credit earning. The key feature of this model is that it is logical - for example, food and water together cost less than the daily credits earned by a user. It also incentivises continued user engagement - for example, because it takes five days of continued energy to save enough credits to buy the PowerPuggle a party. However, it's important to note that without user testing, we cannot be sure about how effective the model is i.e. whether these decisions will maximise user engagement. If we were pushing this product towards deployment, we would need to Incorporate a user testing before proceeding to make sure that our assumptions around the right level of complexity etc stand up to real use of the device.

Our model consists of 11 possible conditions for the PowerPuggle to be in and 4 possible actions through which the user can engage with the PowerPuggle. The conditions consist of:

- Egg: the egg condition is for the PowerPuggle during the first seven days when the sensor is collecting data for the baseline
- Hatching: the Puggle hatches after seven days of data collection is complete. This signals the beginning of the user's active engagement with the PowerPuggle.
- Normal state: this is the condition the Puggle is when the user interacts with the Puggle (i.e. before taking any of the actions below). We decided that this state would show the Puggle swimming.
- Health condition of between 1-7. We chose seven conditions for health as it allowed for the PowerPuggle to start at 4 health bars when it hatched, and left users with 3 bars in either direction. This meant the Puggle wouldn't get sick after one day of neglect, and also would take at least a couple of weeks of ideal care and attention to reach full health, thus encouraging users to stay engaged.
- A state for the Puggle being sick. The Puggle loses health if the user does not feed it or give it water every day. When the Puggle is at one health, it becomes sick.

The four actions we decided to use Include:

- Feeding the Puggle
- Giving it water
- Tickling it
- Throwing it a party

The table below shows the system we developed for how many credits certain actions require, and what impact they have on the PowerPuggle's health.

Table 3. Possible actions and credit required

Action	**Credits to purchase action**	**Effect on health**
Feed Puggle	4	No increase, but if Puggle does not receive food and water every day, health diminishes by 1
Give Puggle water	4	No increase, but if Puggle does not receive food and water every day, health diminishes by 1
Tickle Puggle	Free	Nothing (to keep user engagement high)
Throw Puggle a party	10	Increase 1 health bar

Table 4 below shows the credit earning system we developed in tandem with the conditions/actions. In developing this, we needed to find a balance between not making it too easy to increase the Puggle's health (to keep user's coming back) and not making it too hard (which would lead to disengagement). We designed the credit system carefully with this in mind. You can see In Table 3 that feeding and giving the Puggle water every day costs 8 credits. Table 4 shows how many credits users earn for changing their behaviour (or not). The two are closely linked. For example, the user earns 10 credits a day for meeting their target. This means that it takes 5 days of meeting the target before you can throw a party for your Puggle and increase its health.

In the meantime, you can tickle it whenever you want, which incrementally increases health (and allows engagement even when you have no credits).

Table 4. Credit system

User meets / does not meet energy saving target	Impact
Yes, user meets daily target	Earns 10 credits
Yes, user meets and exceeds daily target (reduces their consumption by 75%)	Earns a bonus 10 credits
User meets their daily energy target every day for seven days in a row	Earns a bonus 10 credits
No, user does not meet daily target	No credits earned

The final element we needed to decide on as part of the algorithm was how to calculate the energy saving target. This is the element that incorporates AI into our system, where we define AI in simple terms as an ability to collect and analyse data and make decisions on the basis of that data[16] (where the decision of the algorithm in this case is the energy saving target).

To do this, we researched what the average daily TV consumption was in Australia. According to Nielson, Australians spend on average two hours and 27 minutes watching live TV and playing back recorded TV content through their TV sets. On a monthly basis, this equates to 74 hours and 58 minutes[17].

Based on this, we played around with various figures for an energy reduction target. We settled on using our algorithm to propose a 10% energy reduction target as it seemed achievable (for the average Australian, that equates to a reduction in viewing every day of around 15 minutes, or just under 2 hours per week.) We also incorporated a feature into the algorithm to allow users to select a larger or smaller target depending on their individual circumstances, thus maintaining the principle of user agency that underpins our system.

This figure constitutes an estimate about what seems reasonable in order to build our prototype. However, without market testing, it's hard to tell whether the 10% figure is the right balance between saving enough energy and being too difficult to achieve.

In addition, when designing the energy saving target, we realised that our simple AI algorithm was not sophisticated enough for long-term use. Specifically, the way our algorithm currently works, users are encouraged to reduce their energy by 10% of the previous week's consumption every week. So for example, if a user watches TV for 1 hour per week, the saving target the algorithm suggests is 10%, which is a reduction of 6 minutes per week (=54 minutes). For week 2, the algorithm suggests a 10% reduction from 54 minutes, which is 5.4 minutes, so the user can only watch TV for 48.5 minutes in the second week. As this process continues, the user is eventually asked to watch no TV at all – and once they aren't watching any TV, how can they get credits for saving energy to take care of their PowerPuggle?

Because of this design problem, our plan for scale includes the introduction of more sophisticated AI-powered algorithms that can take into account diminishing returns and optimise the energy-saving target based on the previous behaviour of the user.

The other important aspect to incorporate into the algorithm was the nudges. Our algorithm dictates that the PowerPuggle sends nudges when the user either meets their energy saving target (in which case, it's a congratulatory message), or doesn't meet their target (in which case It's a message telling them to save energy because their PowerPuggle is hungry). In accordance with the EAST framework, which emphasises that nudges should be sent out at the time the user will be most receptive, our design for scaling also incorporates a more

[16] Schroff, 2019
[17] Nielson, 2018

sophisticated AI-powered learning algorithm to monitor engagement habits of the user and send these nudges at the time of day when the user is most likely to want to engage.

With all those decisions made, we were able to map out the final form of the algorithm, including all the conditions, possible actions, credit system and energy-saving target. In the final version (Figure 3), you can also see multiple feedback loops within the algorithm, which constitute a secondary feedback loop in our system (where the primary loop is between the user and the PowerPuggle – see GUI).

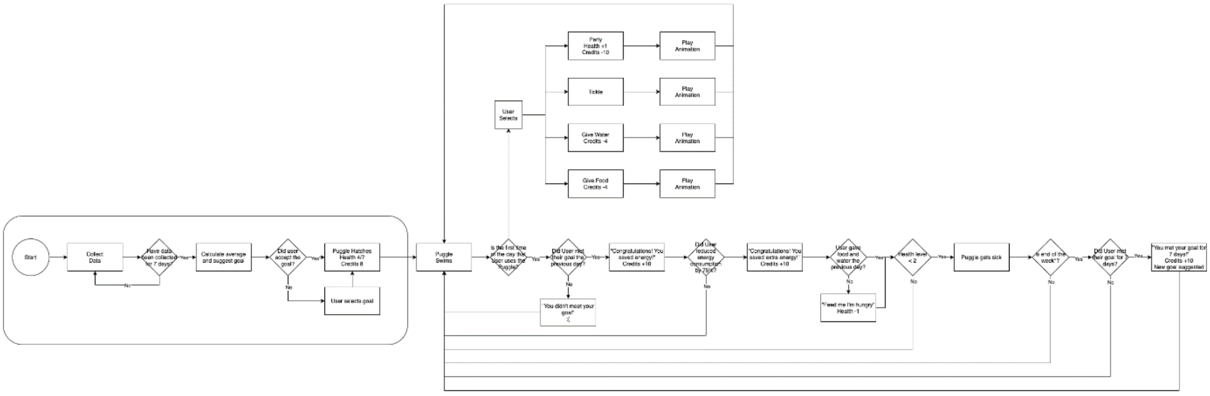


Figure 3. Algorithm for PowerPuggle operation

Code

With the algorithm completed, we then needed to implement it in code. A key decision here was choosing the coding language. We had two options - Ruby and Python. Both had advantages and disadvantages and making our selection involved weighing the trade-off between them. The advantages of Ruby were that it is very easy-to-use for web development and Victor (who led the code development) had experience with Ruby. However, no-one else in the team did (and we all wanted to help with the code). In addition, none of us (not even Victor) had developed a GUI using Ruby before, but we knew (thanks to Victor's existing knowledge) that it was not a preferred language for GUI development.

Python, on the other hand on the other hand, has a lot of support and community knowledge. All members of the team had some experience using it. Also, we knew that the code was going to be shipped in a Raspberry Pi and that Python supports physical interactions and interfaces through the Raspberry Pi GPIO. However, we didn't know at the outset whether Python was suitable for use with web development as none of us had used it in that way before. In order to make a decision, Victor investigated using various online forums and knowledge bases. At the end of this process, he decided that Python would be suitable for our purposes and so that's what we chose.

To prototype the code, we decided to use Jupyter Notebooks because it's easy to share and collaborate on. Steps required were:

1. Creating an https request to get data from our private API (see Internet service)
2. Creating a local database to store the data
3. Creating a graphical user interface (GUI)
4. Finding libraries to use Raspberry Pi physical buttons embedded in the screen
5. Finding Python support for the touchscreen

With respect to the second step, we needed a local database to store the data coming back from interactions between the user and the physical component of the Power Puggle (e.g. somewhere to store information about whether the user had fed/given water/thrown a party/tickled the PowerPuggle). In deciding which database to use, we considered several options, including MySQL, Postgres and SQLite. The trade-off we faced here was between security (important for the assurance of the system - see SYSTEMS ANALYSIS

Table 5. Summary of Systems Analysis

System map	Description
1. Component map	This system map allowed us to visualise the different components of the system and how they would work together. It made us think through the requirements of the system carefully and prompted sus to ask questions like - how will the data get from the sensor to the physical device with the user? How will the user engage with the PowerPuggle? See summary in Design Requirements (Figure 1).
2. Algorithm	This system map sets out the steps we needed to implement into code and prompted us to make decisions with respect to how many actions we wanted to be available to our user, and how many conditions we needed for the PowerPuggle. We learned a lot from designing the algorithm, and it was the most complicated system map we used. Principally, it made us think carefully about the user pathway and how we wanted users to be able to engage with the system. It prompted us to make decisions about the number of conditions and actions we wanted to have available. It also required us to design the credit earning system (including deciding how many credits a user would earn and how many credits each action would cost). Part of this was also making a decision to include a 'free' action (the tickle), as we realised that our design might only allow the user to engage with the system via spending credits every 24 hours (because the user only earns credits every 24 hours). See Algorithm
3. User experience journey map	We used aspects of user journey mapping to guide us in creating a rich user experience. Our user map marks the major points of Interaction that the user will have with the PowerPuggle at the initial stage and beyond. Visualising the user process helped us identify pain points where a user might feel disengaged (delayed feedback loops) or where a user might feel helpless (e.g. PowerPuggle falls sick too quickly). This influenced the decisions we took in designing our algorithm. See Attached PPT
4. Scale system map	We used this system map to develop and conceptualise our plan for scaling the prototype. Creating this map allowed us to come up with a logical set of steps for scaling the product and helped us to determine which steps should come first (i.e. we wanted to have accessibility and customisable options available before incorporating the social networks). See Scale

SCALE

Definition of scale: in the context of our project, we defined scale in several stages:

1. **Prototype 1 (complete)**: proof of concept
2. **Prototype 2 (3 years)**: this would involve scaling our original prototype to **encompass electricity consumption data from the whole house, for example via connection to the home's smart meter**. To achieve this scale, an agreement would be needed either with a) electricity distributors who own the smart meters or b) agreements with individual households who have installed their own smart meter. In the absence of a mechanism by which the PowerPuggle could be connected to home smart meters, an alternative method to collect aggregate data on home energy consumption would be required. This would be less neat but still achievable. It would likely require the installation of sensors on each electrical appliance the household wished to include in energy monitoring. Those sensors would be linked in a network to aggregate energy consumption (effectively taking the place of a smart meter). This level of scale would also include connection to energy generation infrastructure within the house i.e. solar panels (note: if a smart meter is used, this step would be automatic as a smart meter also measures energy flowing back into the grid).

As part of this step, we would also incorporate **more sophisticated AI algorithms**, specifically to generate weekly energy targets that take into account previous energy targets. This is necessary because at present, the algorithm calculates a new energy saving target based on the previous 7 days energy consumption. However, if a user continually reduces their energy consumption, there will be a point when further reductions are no longer feasible. In this case, if the algorithm continues to recommend further reductions, the user will disengage from the system. Therefore, part of our plan for scaling the system involves the creation of a smarter AI-powered algorithm to optimise the energy-saving targets, creating a balance between encouraging further reductions in energy use and keeping the user engaged in the system.

A final part of this step would include incorporating a **strong data security plan** to protect user's electricity consumption data. We would also Investigate what sort of access to the data might be useful for the user - some users may wish to be able to access their electricity data in a usable format separate to the PowerPuggle interface.

3. **Customisation (4 years)**: the EAST framework says that in order for nudges to be effective, they need to be attractive. A key way to make a product attractive is to allow for customisation. In the context of the PowerPuggle, customisation also serves another purpose, which is to **make the PowerPuggle relevant and appropriate for diverse audiences**. Thus, this part of our scaling plan would include – at a minimum – creating options for the user to select the animal used (i.e. instead of a platypus – which is highly relevant in an Australian context, but not necessarily elsewhere – the user could select other animals more relevant to their own cultural setting. We also wanted to include the possibility of the PowerPuggle taking the form of a plant.) Other customisation options could include selecting the actions the user can apply to the Puggle – i.e. instead of a tickle, which might not always be culturally appropriate, perhaps a cuddle or a kiss might be possible.

   We are also conscious that at present, the Power Puggle is not suitable for use by vision-impaired people. As part of this stage, therefore, we would also seek to add accessibility features like voice-control, high contrast, text-to-speech and a vibration alert (in addition to the buzzer).

4. **Social networks (6 years)**: another key part of the EAST framework is that nudges should be social because this incentivises behaviour change. We plan to incorporate this as the PowerPuggle scales by **embedding social networks into the function of the PowerPuggle**. Our idea is that users could create a 'friend' network with the PowerPuggle, allowing them to compare energy saving across their network. This could even include extra credits or the potential of unlocking new actions for your energy friend as a reward for being the best energy saver in your group. This feature is also important as another aspect of scale we considered was whether our **PowerPuggle could be used in a corporate setting** to encourage employees at work to reduce their energy consumption. The network feature would allow workplaces to connect teams or divisions to the network to engage in friendly competition on their energy saving targets. A final aspect of the social scaling aspect is that we thought about the potential for community interfaces. The idea is that in addition to the individual energy friends that people carry around, there could be a screen in a prominent place showing different people's energy saving and regarding the top few energy savers. This would be especially appropriate in a corporate setting (where the best energy savers could be rewarded by the company and the worst might have to staff the annual Christmas party etc) but could also be useful in homes as a way to compare individual users' energy saving (and potentially highlight the achievements of children).

   At the same time as we incorporate social networks, we plan to **send the PowerPuggle virtual** via an app and possibly AR. Carrying around a physical energy friend does have its advantages, as the physical object serves as a very tangible reminder. However, carrying something around can also be a burden when we have so many things to carry and remember. So our energy friend could be virtual, similar to Pokemon GO. In this case, the system would still function the same (i.e. credits, reminders to take action and rewards/sanctions for achieving or not achieving targets), but it would all be located on a mobile phone, and the energy friend would be visible through AR. In this case, certain customisation options could be monetised, with the proceeds going towards sustainable energy initiatives. A virtual version would make expanding the design to incorporate social networks much easier.

5. **Decommissioning**: part of thinking about scaling a system includes developing plans to decommission it. As a physical object, we acknowledge that when the PowerPuggle reaches the end of its life, it produces waste. Given the energy conservation focus of our CPS, a key goal for us was to ensure the decommissioning process was as sustainable as possible. Our design to do this includes:
    a. design: **the PowerPuggle is easy to repair**. Its build ensures that it can be easily disassembled and individual components can be accessed. It is also made from parts that are easy to find and don't cost a lot. This means that if something breaks, the PowerPuggle can be fixed rather than thrown out.
    b. **lifetime warranty**: we plan for the PowerPuggle to come with a lifetime warranty. This complements our 'easy repair' design – if the Puggle breaks, send it back and we'll either fix it or provide a replacement. The replacement won't necessarily be new, but is more likely to come from our recycling program – see below – thus creating a virtuous cycle of re-use.
    c. planning for **a recycling program**. If the PowerPuggle is broken or the user no longer wishes to use, it they can return it. If it's beyond repair, we would harvest any recyclable parts and re-use them. If it's repairable, we would repair it and either send it to existing customers whose Puggle has broken or put it back into circulation as a refurbished model.
    d. considering ways to **make the production of our PowerPuggle more sustainable**. When planning production at scale, we would seek to use recyclable and sustainable materials to produce our energy friend case. This could include wood (rather than the plastic we used for our prototype), but we would need to investigate further the environmental impact of various materials and production processes before making a final decision. Our plan for scale also includes ways to minimise waste. For example, making the PowerPuggle virtual would remove physical waste entirely.
    e. providing a mechanism **to delete accounts and data**: user data and privacy is of paramount concern for decommissioning the system. Users should have the right to have all data and account information deleted from the PowerPuggle system quickly and easily.

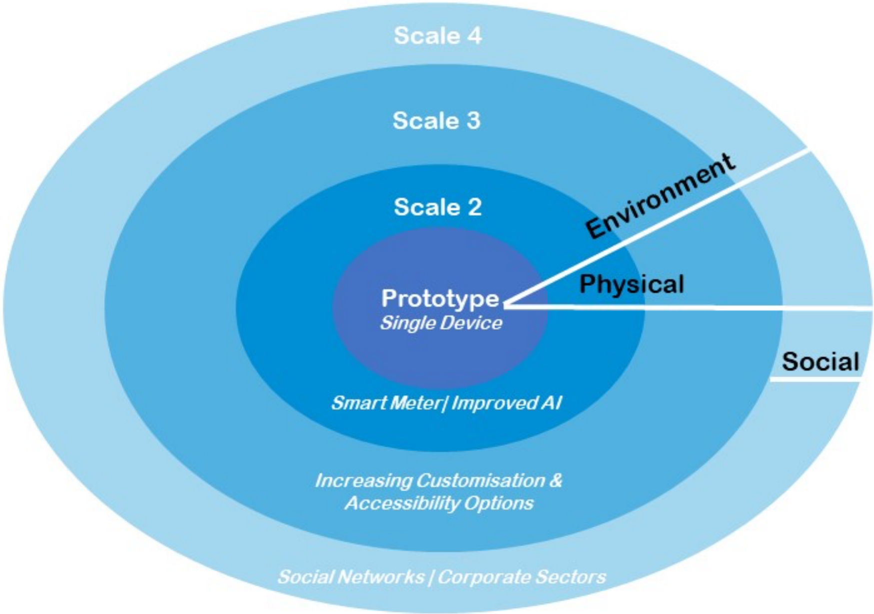


Figure 3. Scaling the PowerPuggle

CYBERNETIC ENGINEERING), size and ease of use. Postgres and MySQL are both comparatively secure and make it easy to access the data, but they are very large (around 50-100 times bigger than SQLite.) SQLite, on the other hand, makes it even easier to access the data and is also much smaller (a smaller footprint was desirable in our case because the storage capacity of our Raspberry Pi was limited) - but it's less secure. As we knew that no-one (except us) would have access to the data through the physical device for the prototyping stage, we decided to use SQLite. We would have to revisit this option at scale in order to provide appropriate data protection for users.

One challenge we faced was Implementing the GUI. Initially, we started to use tkinter, as it's the most common Python framework for GUI design. But working with GIFs (which we had decided to use for the interface - see GUI) in tkinter is very difficult because all the image processing has to be done by code. So we changed to QT, which has a native animated image processing capability. Another advantage of QT is that it has a capability to work with embedded labels, which allows nesting of labels.

For steps 4 and 5, minimal research revealed that the Raspberry Pi OS provides native support for both using physical buttons embedded in the screen and a touchscreen.

The code Is accessible here: https://gitlab.cecs.anu.edu.au/u7091150/powerpuggle

Internet service

The internet service pulls data from the sensor every 5 minutes and pushes that data to a server database (not the local one mentioned above) that is accessible to us via a private API (Application Programming Interface) that we designed.

We needed an internet service due to a key decision we made while implementing the code about whether to use on-device storage or cloud-based storage for retrieving and storing the electricity consumption data. The trade-off was between security, internet connectivity and time required for development. The advantage of on-device storage is that the user's data is stored on the device, thus enhancing security and privacy. However, for this to work, the device needs to be connected to the internet 24/7. Relying on a cloud-based retrieval and storage process doesn't require 24/7 internet connection. However, it adds a layer of vulnerability in terms of security and also requires a new internet service to do the job (which takes time to develop). Ultimately, we decided that assuming the user would be connected to the internet 24/7 was not reasonable, and so went with the cloud-based data storage and developed our own internet service to do the job.

To do this, we created a Ruby algorithm that is stored in an Amazon web service (AWS) and is run autonomously every five minutes. This process pulls the data from the sensor API. Once this process has pulled the data, it pushes it to our private API that is stored in Heroku. The private API stores the data in the database.

We created the API using Ruby on Rails to store the energy consumption data that is pushed every 5 minutes. It has one accessible endpoint with two http verbs enabled - POST and GET. When a POST http request is sent to the endpoint, a new record is created. When a GET request is sent to the endpoint, it provides the existing records in the database in a JSON object.

Victor took the lead in setting this part up as he was the only one in the team with Ruby experience.

Sensor

Materials

- Powertech Smart Plug with Energy Monitor
- Connection to internet
- Sensor API
- API Keys

The sensor is connected to a device in the user's home and pulls energy consumption data from it. When deciding on which sensor to choose, our main design requirements:

- Easy access to data
- Cheap (within our budget)
- Accurate (to minimise the chance of false positives and negatives)
- Located locally (so we didn't have to wait for shipping – there have been COVID delays)

Based on these requirements, Victor investigated options. The sensor we chose was called the Powertech Smart Plug with Energy Monitor. It was the cheapest available locally. It is acceptably accurate (considering the IEEE ANSI C12.1 standard is +/- 2%[18]). At the time, we were not sure whether the sensor would enable us to access the data easily (i.e. through its API); however, none of the sensors we investigated were clear about this, and so we decided to order one early, experiment with it and then replace it with something else if necessary (you may recall that one of the risks we covered in our Design Brief was the possibility of needing to hack the sensor to solve this problem).

After the sensor arrived and we could investigate it physically, we discovered there was an accompanying app that allowed users to get their energy consumption data in their smartphones. This meant that even if we didn't have access to an API to pull data from the sensor, we could reverse-engineer the app and gain access to the sensor's data. Ultimately, however, we didn't even need to use that method, as the sensor manufacturer had a public facing API for developers to implement third-party applications and services for/through the sensors. We registered as developers and obtained API Keys with which we were able to code our implementations.

The other major challenge we faced was that the power sensor only provides instant power consumption readings and doesn't have a way of providing energy consumption over a period of time. This was not suitable for our use, as a key part of the PowerPuggle design is the ability to record data over seven days in order to generate an energy reduction target. To solve this, we set up a virtual server and a private API to pull instant consumption data from the sensor, and store that information in a database. We could then pull this data from the private API as needed.

[18] https://www.pjm.com/-/media/committees-groups/task-forces/mtf/20151113/20151113-item-08-ansi-and-ieee-standards.ashx

A note on why we selected the TV for our prototype. We chose a TV because – for most people – it's a non-essential electrical appliance. That means it is possible to reduce energy consumption (by reducing the time we spend watching TV) without a detrimental effect on daily life. Connecting the PowerPuggle to a fridge, on the other hand, would be far less effective as it's much more difficult to reduce the energy consumption of refrigeration. In addition, research we did Indicated that 82.6% of the Australian population watches broadcast TV on In-home TV sets each week[19]. Thus, a device built around reducing TV consumption would be broadly applicable In the Australian market.

Design requirements marking criteria

Table 2 is a summary of marking criteria listed under Design requirements in the marking rubric and where we have addressed them.

Table 2. Design Requirements Summary

Criterion	Demonstration
Your team has clearly incorporated artificial intelligence into the design of your cyber-physical system.	The algorithm the prototype PowerPuggle uses to generate an energy saving target can be thought of as simple AI, in that it classifies and analyses data, and makes decision based on that data (i.e. to recommend an energy saving target to the user)[20]. However, our design for the Puggle includes the incorporation of more intelligent AI as the device scales. This is necessary because at present, the algorithm calculates a new energy saving target based on the previous 7 days energy consumption. However, if a user continually reduces their energy consumption, there will be a point when further reductions are no longer feasible. In this case, if the algorithm continues to recommend further reductions, the user will disengage from the system. Therefore, part of our plan for scaling the system involves the creation of a smarter AI-powered algorithm to optimise the energy-saving targets, creating a balance between encouraging further reductions in energy use and keeping the user engaged in the system. The second way our design incorporates smarter AI at scale to Is to monitor user engagement and time nudges so that they arrive when the user is likely to be most receptive to them. This Is a principle under the EAST framework. See Algorithm
Your team has clearly shown your CPS has at least one feedback loop that includes a physical sensor and a means of generating action in digital / physical worlds.	There is one main feedback loop at work in the PowerPuggle, between the device and the human user. Based on data collected from the physical sensor, the device nudges the user to engage with the system. The user responds to that nudge by changing their energy consumption (I.e. the PowerPuggle generates action in the real world), thus earning credits (or not). They then interact with the physical part of the PowerPuggle (through the buttons and screen) to use the credits. See Graphical User Interface (GUI) There are also a number of secondary feedback loops in the algorithm itself, which also work to prompt changes to user behaviour, thus generating action in the physical world. See Algorithm
Your team has demonstrated that the CPS you have developed makes decisions that have the capacity to influence the physical, social, and/or environmental world in some way.	The algorithm in the PowerPuggle decides on an energy reduction target, proposes it to users and then nudges them to take action. In this way, it has the capacity to influence the *physical* world. See Software Our vision for scale includes expansion of the PowerPuggle to incorporate social networks, which (according to the EAST framework) are an excellent leverage tool to change behaviour. Thus, the PowerPuggle has the capacity to influence the *social* world. See Scale Even if only a single person reduces their energy as a result of using the PowerPuggle, decisions made by the PowerPuggle can be said to influence the environmental world.

[19] Nielson 2018
[20] https://medium.com/mytake/artificial-intelligence-explained-in-simple-english-part-1-2-1b28c1f762cf

	At scale, this effect would be amplified – the more people who use decisions on energy saving targets generated by the PowerPuggle, the larger the effect on the environmental world will be. Thus, the PowerPuggle also has the capacity to influence the *environmental* world. See Scale
Your team has demonstrated that the CPS you have developed has a means of interacting or interfacing with humans.	The PowerPuggle has an interface consisting of the physical case/components and also the GUI. Human users interact with the PowerPuggle via the interface, which allows the user to select from a number of actions. See Physical components

SYSTEMS ANALYSIS

Table 5. Summary of Systems Analysis

System map	Description
1. Component map	This system map allowed us to visualise the different components of the system and how they would work together. It made us think through the requirements of the system carefully and prompted sus to ask questions like - how will the data get from the sensor to the physical device with the user? How will the user engage with the PowerPuggle? See summary in Design Requirements (Figure 1).
2. Algorithm	This system map sets out the steps we needed to implement into code and prompted us to make decisions with respect to how many actions we wanted to be available to our user, and how many conditions we needed for the PowerPuggle. We learned a lot from designing the algorithm, and it was the most complicated system map we used. Principally, it made us think carefully about the user pathway and how we wanted users to be able to engage with the system. It prompted us to make decisions about the number of conditions and actions we wanted to have available. It also required us to design the credit earning system (including deciding how many credits a user would earn and how many credits each action would cost). Part of this was also making a decision to include a 'free' action (the tickle), as we realised that our design might only allow the user to engage with the system via spending credits every 24 hours (because the user only earns credits every 24 hours). See Algorithm
3. User experience journey map	We used aspects of user journey mapping to guide us in creating a rich user experience. Our user map marks the major points of Interaction that the user will have with the PowerPuggle at the initial stage and beyond. Visualising the user process helped us identify pain points where a user might feel disengaged (delayed feedback loops) or where a user might feel helpless (e.g. PowerPuggle falls sick too quickly). This influenced the decisions we took in designing our algorithm. See Attached PPT
4. Scale system map	We used this system map to develop and conceptualise our plan for scaling the prototype. Creating this map allowed us to come up with a logical set of steps for scaling the product and helped us to determine which steps should come first (i.e. we wanted to have accessibility and customisable options available before incorporating the social networks). See Scale

SCALE

Definition of scale: in the context of our project, we defined scale in several stages:

6. **Prototype 1 (complete)**: proof of concept

7. **Prototype 2 (3 years)**: this would involve scaling our original prototype to **encompass electricity consumption data from the whole house, for example via connection to the home's smart meter**. To achieve this scale, an agreement would be needed either with a) electricity distributors who own the smart meters or b) agreements with individual households who have installed their own smart meter. In the absence of a mechanism by which the PowerPuggle could be connected to home smart meters, an alternative method to collect aggregate data on home energy consumption would be required. This would be less neat but still achievable. It would likely require the installation of sensors on each electrical appliance the household wished to include in energy monitoring. Those sensors would be linked in a network to aggregate energy consumption (effectively taking the place of a smart meter). This level of scale would also include connection to energy generation infrastructure within the house i.e. solar panels (note: if a smart meter is used, this step would be automatic as a smart meter also measures energy flowing back into the grid).

   As part of this step, we would also incorporate **more sophisticated AI algorithms**, specifically to generate weekly energy targets that take into account previous energy targets. This is necessary because at present, the algorithm calculates a new energy saving target based on the previous 7 days energy consumption. However, if a user continually reduces their energy consumption, there will be a point when further reductions are no longer feasible. In this case, if the algorithm continues to recommend further reductions, the user will disengage from the system. Therefore, part of our plan for scaling the system involves the creation of a smarter AI-powered algorithm to optimise the energy-saving targets, creating a balance between encouraging further reductions in energy use and keeping the user engaged in the system.

   A final part of this step would include incorporating a **strong data security plan** to protect user's electricity consumption data. We would also Investigate what sort of access to the data might be useful for the user - some users may wish to be able to access their electricity data in a usable format separate to the PowerPuggle interface.

8. **Customisation (4 years)**: the EAST framework says that in order for nudges to be effective, they need to be attractive[21]. A key way to make a product attractive is to allow for customisation. In the context of the PowerPuggle, customisation also serves another purpose, which is to **make the PowerPuggle relevant and appropriate for diverse audiences**. Thus, this part of our scaling plan would include – at a minimum – creating options for the user to select the animal used (i.e. instead of a platypus – which is highly relevant in an Australian context, but not necessarily elsewhere – the user could select other animals more relevant to their own cultural setting. We also wanted to include the possibility of the PowerPuggle taking the form of a plant.) Other customisation options could include selecting the actions the user can apply to the Puggle – i.e. instead of a tickle, which might not always be culturally appropriate, perhaps a cuddle or a kiss might be possible.

   We are also conscious that at present, the Power Puggle is not suitable for use by vision-impaired people. As part of this stage, therefore, we would also seek to add accessibility features like voice-control, high contrast, text-to-speech and a vibration alert (in addition to the buzzer).

9. **Social networks (6 years)**: another key part of the EAST framework is that nudges should be social because this incentivises behaviour change[22]. We plan to incorporate this as the PowerPuggle scales by **embedding social networks into the function of the PowerPuggle**. Our

[21] EAST: Four simple ways to apply behavioural insights https://www.bi.team/wp-content/uploads/2015/07/BIT-Publication-EAST_FA_WEB.pdf

[22] EAST: Four simple ways to apply behavioural insights https://www.bi.team/wp-content/uploads/2015/07/BIT-Publication-EAST_FA_WEB.pdf

idea is that users could create a 'friend' network with the PowerPuggle, allowing them to compare energy saving across their network. This could even include extra credits or the potential of unlocking new actions for your energy friend as a reward for being the best energy saver in your group. This feature is also important as another aspect of scale we considered was whether our **PowerPuggle could be used in a corporate setting** to encourage employees at work to reduce their energy consumption. The network feature would allow workplaces to connect teams or divisions to the network to engage in friendly competition on their energy saving targets. A final aspect of the social scaling aspect is that we thought about the potential for community interfaces. The idea is that in addition to the individual energy friends that people carry around, there could be a screen in a prominent place showing different people's energy saving and regarding the top few energy savers. This would be especially appropriate in a corporate setting (where the best energy savers could be rewarded by the company and the worst might have to staff the annual Christmas party etc) but could also be useful in homes as a way to compare individual users' energy saving (and potentially highlight the achievements of children).

At the same time as we incorporate social networks, we plan to **send the PowerPuggle virtual** via an app and possibly AR. Carrying around a physical energy friend does have its advantages, as the physical object serves as a very tangible reminder. However, carrying something around can also be a burden when we have so many things to carry and remember. So our energy friend could be virtual, similar to Pokemon GO. In this case, the system would still function the same (i.e. credits, reminders to take action and rewards/sanctions for achieving or not achieving targets), but it would all be located on a mobile phone, and the energy friend would be visible through AR. In this case, certain customisation options could be monetised, with the proceeds going towards sustainable energy initiatives. A virtual version would make expanding the design to incorporate social networks much easier.

10. **Decommissioning**: part of thinking about scaling a system includes developing plans to decommission it. As a physical object, we acknowledge that when the PowerPuggle reaches the end of its life, it produces waste. Given the energy conservation focus of our CPS, a key goal for us was to ensure the decommissioning process was as sustainable as possible. Our design to do this includes:
    a. design: **the PowerPuggle is easy to repair**. Its build ensures that it can be easily disassembled and individual components can be accessed. It is also made from parts that are easy to find and don't cost a lot. This means that if something breaks, the PowerPuggle can be fixed rather than thrown out.
    b. **lifetime warranty**: we plan for the PowerPuggle to come with a lifetime warranty. This complements our 'easy repair' design – if the Puggle breaks, send it back and we'll either fix it or provide a replacement. The replacement won't necessarily be new, but is more likely to come from our recycling program – see below – thus creating a virtuous cycle of re-use.
    c. planning for **a recycling program**. If the PowerPuggle is broken or the user no longer wishes to use, it they can return it. If it's beyond repair, we would harvest any recyclable parts and re-use them. If it's repairable, we would repair it and either send it to existing customers whose Puggle has broken or put it back into circulation as a refurbished model.
    d. considering ways to **make the production of our PowerPuggle more sustainable**. When planning production at scale, we would seek to use recyclable and sustainable materials to produce our energy friend case. This could include wood (rather than the plastic we used for our prototype[23]), but we would need to investigate further the environmental

[23] Initially, we planned to make the casing for our PowerPuggle out of wood using the laser cutter, but the only laser cutter at ANU that is capable of cutting wood is located in the School of Art and Design, which has been closed for the duration of the semester due to COVID 19.

impact of various materials and production processes before making a final decision. Our plan for scale also includes ways to minimise waste. For example, making the PowerPuggle virtual would remove physical waste entirely.

e. providing a mechanism **to delete accounts and data**: user data and privacy is of paramount concern for decommissioning the system. Users should have the right to have all data and account information deleted from the PowerPuggle system quickly and easily.

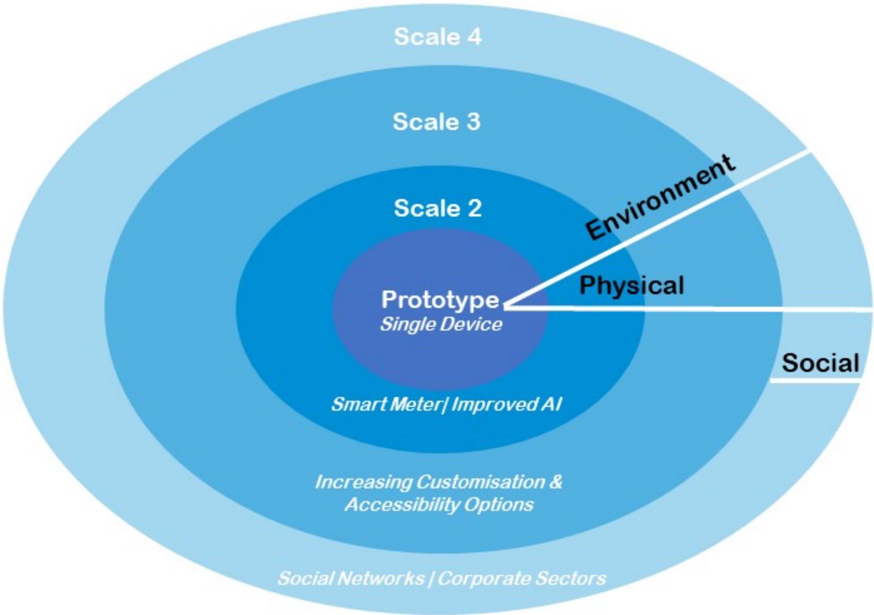


Figure 3. Scaling the PowerPuggle

CYBERNETIC ENGINEERING

In designing our PowerPuggle, we've taken into account NBE concepts and perspectives, primarily by considering how our system will scale safely, responsibly and sustainably. We also considered the A's and I's in two places: firstly as part of the system design (for more information see Design Requirements), and secondly in this section (note that while all six are covered in the Design Requirements section, not all of them are covered here again.)[24] Consideration of the system through these two frameworks allowed us to visualise the success and limitations of our project and incorporate mitigating features Into our design.

Safely

Making sure a system is safe is closely connected with the idea of **assurance**. Assurance is the extent to which a system achieves a desired level of confidence that it will perform according to designed behaviour under any set of conditions[25].

When complex systems do not perform according to designed behaviours, the consequences can be insignificant, or even beneficial. However, system designers should be particularly concerned when the

[24] In particular, consideration of our system's intent and autonomy does not fit well under analysis of our PowerPuggle using the safe, responsible and sustainable framework. Both are considered in the section on Design requirements.

[25] Cyber-Physical Systems Public Working Group 2016

unintended consequences are unsafe. Thus, in looking at how to *assure* our PowerPuggle, we have primarily considered how to make our PowerPuggle *safe*.

In examining the safety of our PowerPuggle, we have examined both virtual safety, which relates to data and privacy, and physical safety:

1. Our primary concern with regard to safety in scaling our PowerPuggle safety relates to the **safety of consumer data and privacy**. At scale, the PowerPuggle will collect and store energy consumption and energy saving data from everyone who uses the system. While we were not too concerned with data security while prototyping, we would need to develop a robust security plan before scaling, for example by revisiting our choice of local database to ensure it did not create vulnerabilities (see Internet service). This Is particularly important as our scaled design Involves using smart meters as the source of energy consumption data. Smart meters are a much more sensitive data source than Individual sensors, as they allow access to data from the whole household and are also used for billing and so need to be protected from misuse. We were also careful to include a plan for allowing users to delete accounts and data in our design for scale, which Is a key aspect of data safety.
2. Our design for the Puggle case also took into account the **physical safety** of users. There are no sharp edges on the case, and even in its prototype form, it is quite solid, robust and hard to break. There are no external small or moving parts that a child could choke on (see Case).

Responsibly

Scaling a system *responsibly* means designing and producing a system that is inclusive and relevant in multiple diverse contexts. We have considered how to responsibly scale our PowerPuggle and incorporated this into our PowerPuggle design by carefully designing the PowerPuggle **interface** to be culturally appropriate and relevant for diverse audiences. Key ways we have done this include:

- making the Puggle customisable in terms of both animal and actions, thus making it culturally relevant and appropriate
- thinking about the systems as being applicable beyond the boundaries of individual homes
- adding accessibility options

Scaling a system responsibly also means giving people choices about whether and how they engage, in other words making sure that users have **agency** vis a vis their engagement with the system. Our PowerPuggle embodies the idea of agency through its design foundation in nudge theory and EAST. Nudge theory is clear about the need for nudges to influence behaviour without forbidding options, significantly changing economic incentives or mandating certain behaviour, thus maintaining the user's agency. The PowerPuggle does not force users to engage, nor are there real-world consequences should a user fail to save energy (beyond possible embarrassment within a circle of friends!) The PowerPuggle relies on gentle persuasion and social networks to effect change. Nothing prevents a user from stopping use of the system. In this way, the PowerPuggle aims to change behaviour while keeping agency in the hands of the user.

Sustainably

We have thought about how to make our system scale sustainable by:

- thinking carefully about ways to make the **production** of our PowerPuggle more sustainable. When planning production at scale, we would seek to use recyclable and sustainable materials to produce our energy friend case. This could include wood (rather than the plastic we used for our prototype[26]),

[26] Initially, we planned to make the casing for our PowerPuggle out of wood using the laser cutter, but the only laser cutter at ANU that is capable of cutting wood is located in the School of Art and Design, which has been closed for the duration of the semester due to COVID 19.

but we would need to investigate further the environmental impact of various materials and production processes before making a final decision.

- thinking carefully about how the system will be **decommissioned**
    - part of our ideas for scaling our system included plans for making the decommissioning of the system as sustainable as possible. As a physical object, we acknowledge that when the PowerPuggle reaches the end of its life, it produces waste. We have tried to mitigate this through:
        - design: the PowerPuggle is easy to repair. Its build ensures that it can be easily disassembled and individual components can be accessed. It is also made from parts that are easy to find and don't cost a lot. This means that if something breaks, the PowerPuggle can be fixed rather than thrown out.
        - lifetime warranty: we plan for the PowerPuggle to come with a lifetime warranty. This complements our 'easy repair' design – if the Puggle breaks, send it back and we'll either fix it or provide a replacement. The replacement won't necessarily be new, but is more likely to come from our recycling program – see below – thus creating a virtuous cycle of re-use.
        - planning for a recycling program. If the PowerPuggle is broken or the user no longer wishes to use, it they can return it. If it's beyond repair, we would harvest any recyclable parts and re-use them. If it's repairable, we would repair it and either send it to existing customers whose Puggle has broken or put it back into circulation as a refurbished model.

CONTRIBUTION

Table 9. Summary of contribution

	Victor	**Caddie**	**Ash**	**Dianna**
Design	Victor took the lead in designing and writing the code	Caddie researched and sourced all the materials for the physical build and drew the graphics for the GUI.	Ash designed and iterated the algorithm	Dianna led on the physical build of the prototype (modelling, 3D printing)
Project planning	All team members participated in weekly project planning meetings on Wednesday. Group discussions resulted in setting a vision for the project; chunking into discrete tasks and allocating responsibility for tasks. Evidence of the output of some of these sessions is in the Project Management section below.			
Stakeholder management	Victor led in engaging stakeholders on with respect to the sensor and gaining access to the real-time data from the sensor	Caddie led on discussing the PowerPuggle concept with stakeholders from our Practice CPS because there is a close aligent between the two projects. The PP has the potential to offer an interface for an energy system such as the community battery. These interactions were covered under our	Ash led in engaging stakeholders on behavioural economics	Dianna led in engaging stakeholders on the physical build, particularly MakerSpace staff.

		existing ethics protocols.		
Budget management	All group members contributed to managing the budget. Before any purchases, we discussed cost and necessity as a group before agreeing to the purchase together. We set up a shared Google Docs page to keep track of expenses and ensure we remained within budget. This document is summarised in the Budget section.			
Topic/skillset outside previous experience	Victor developed a GUI/UX for the first time using the Nokia QT Framework.	Caddie soldered for the first time, connecting the boost board to the Raspberry Pi and also designed GIFs for the first time.	Ash developed expertise in nudge theory and the EAST framework	Dianna developed intermediate 3D modelling skills in Fusion360 (building on her experience with TinkerCAD as part of her Build learning portfolio)

Project Management

Building all the PowerPuggle components within the timeframe allowed and amidst a very heavy course load required significant project planning. All team members contributed to project planning throughout the semester. We met as a team every Wednesday for several hours. The first chunk of these sessions was always devoted to project planning and management, including tasks such as:

- Brainstorming ideas (in the early phases - see Photo 5)
- Developing the concept for PowerPuggle (see Photo 6)
- Developing timelines (see Figure 4. GANTT Chart)
- Allocating tasks
- Checking in on progress (see Figure 4. GANTT Chart)
- Helping each problem solve

Other techniques we employed to help our project management included a team trello board that showed all the deadlines across both Build and Practice; a Teams chat for the group, where team members added thoughts or questions at any time; and regular Zoom meetings (see Photo 7).

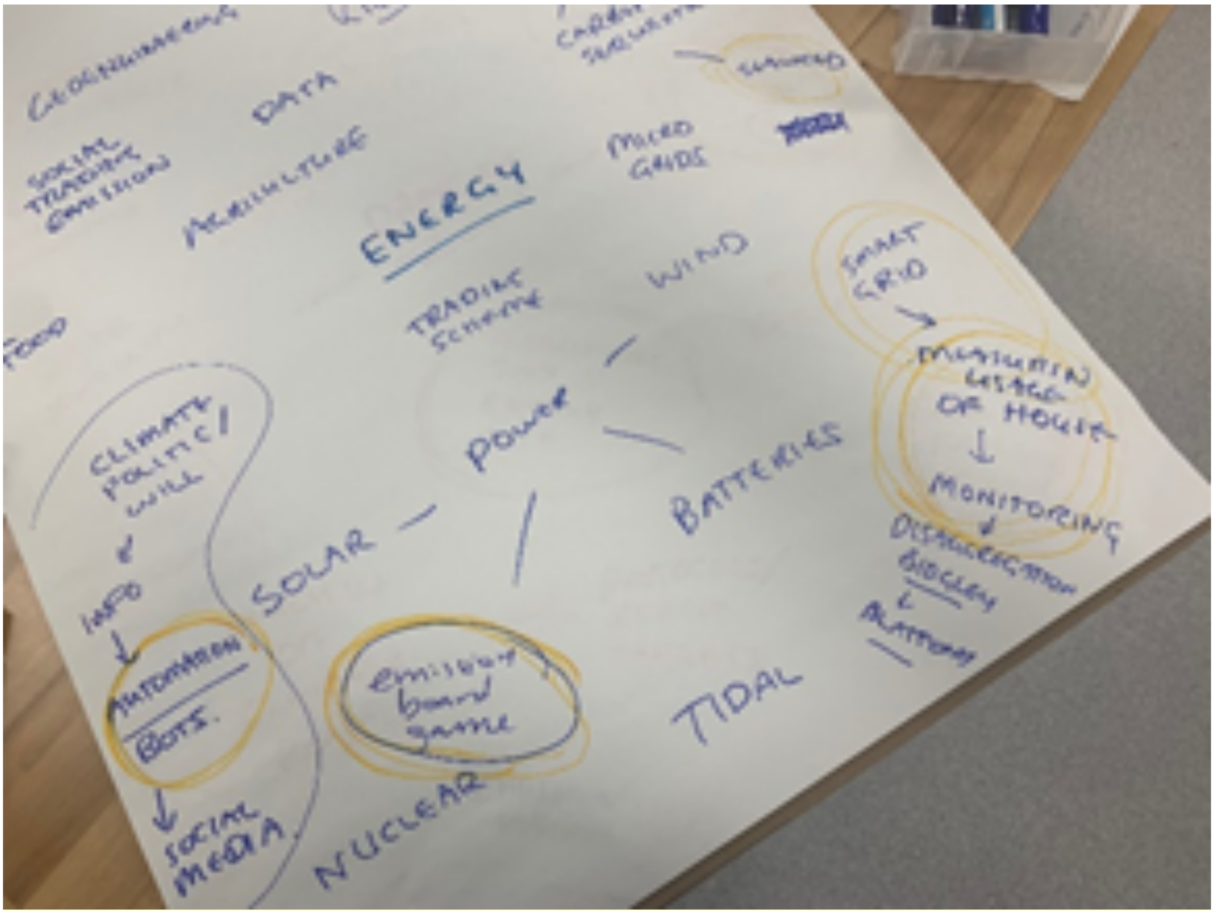


Photo 5. We didn't start off with the PowerPuggle - we investigated many different options for making a CPS. This photo is from one of our early brainstorming sessions.

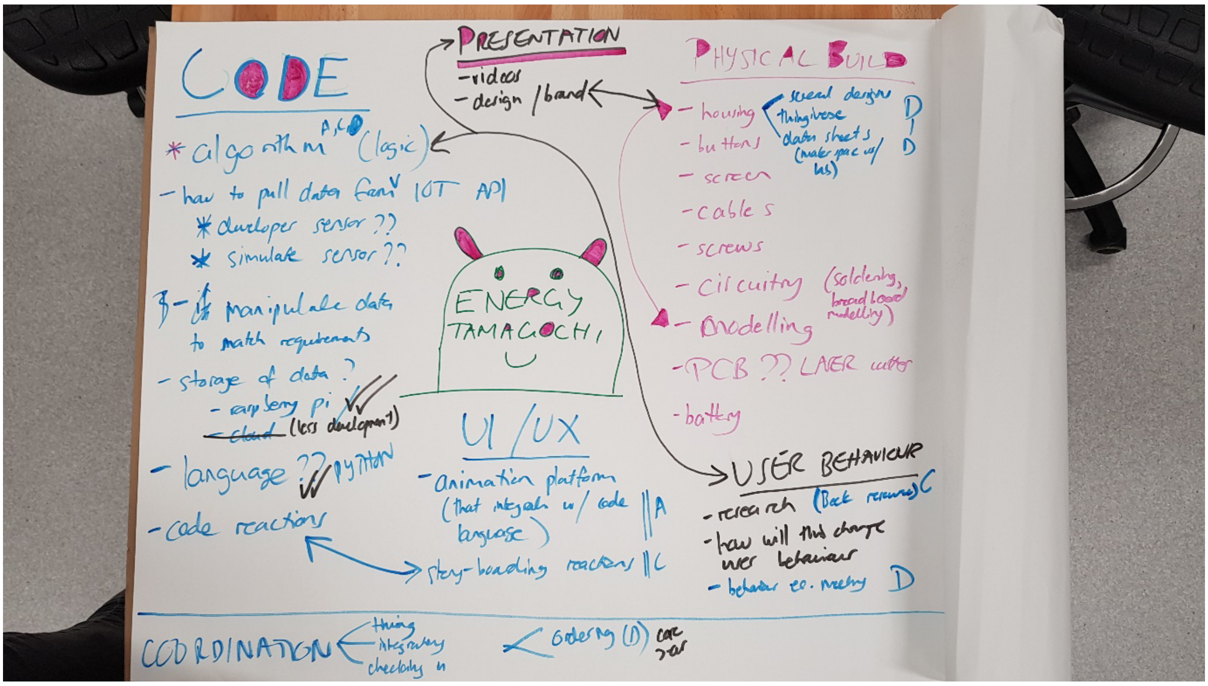


Photo 6. This is our initial concept map of the PowerPuggle including a break-down of all the components we thought we would need. It includes many items that ultimately became our key decision points (e.g. how to pull data, how to store data, how to make the GUI, how to build the case etc.)

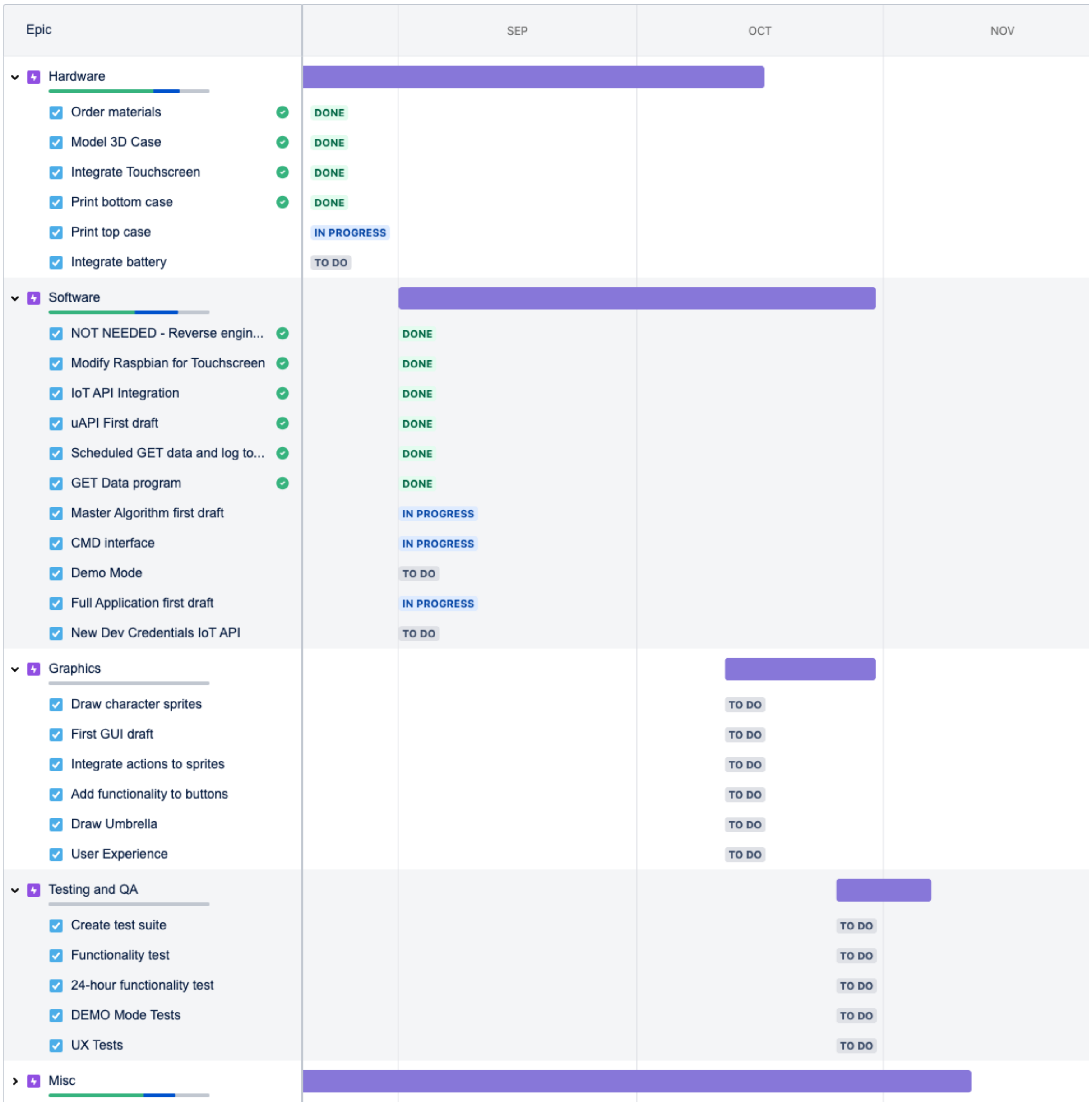


Figure 4. GANTT Chart (shows tasks and progress on PowerPuggle as of October)

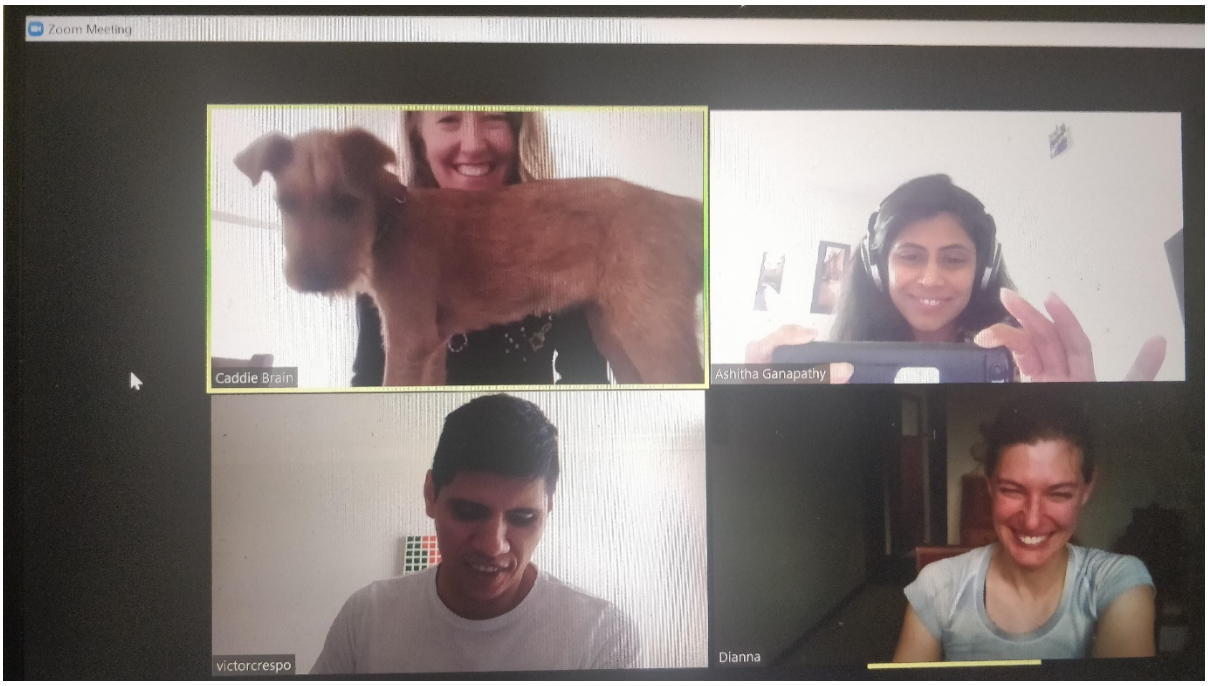


Photo 7. Team meetings on Zoom (with special invited guests…we thought about making her the Puggle too!)

EVIDENCE

Evidence of meeting the criteria under this section of the marking rubric (Evidence) is provided throughout this document. Table 6 indicates where.

Table 6. Summary of Evidence

Criterion	Demonstration
You have provided evidence of completion of design elements	At the Demo Day we provided evidence of: • A completed physical object, consisting of a case and internal components • A completed GUI • Completed software including an algorithm and code in Python, and an internet service • A completed system incorporating a physical sensor to measure energy consumption, the physical object and the virtual internet service Code can be found here: https://gitlab.cecs.anu.edu.au/u7091150/powerpuggle
You have provided evidence of having undertaken a systematic process to the design and build of your prototype	Evidence of systematic processes we used included: • Design of the system incorporating the application of nudge theory through the EAST framework - see Theme • Our approach to building the case - See Case • Our approach to assembling the components - See Physical components • Our approach to developing the GUI – see GUI • Our approach to developing the code - see Code • Our approach to deciding on and implementing the decision to use cloud-based storage – see Internet service

	• All of these components were developed in the context of a systematic approach to project management - See Contribution
You have provided evidence of information gathering that informed key decision-making points.	We provided evidence of information gathering that informed key decision-making with respect to the following decisions: • How to design the system to maximise changes to user behaviour and impact on the physical world (using nudge theory) - see Theme. • Decisions related to the physical components, including: deciding on a process to make the case, including deciding to 3D print it, use Fusion360 to model it - see Case • Decisions related to the GUI including: the design and lay-out, using the platypus, and using GIFs – See GUI • Decisions about the Software including: why we selected 10% as the energy reduction target; decision to use Python as the programming language; decision on the local database; decision on server vs on-device storage - see Software

CRITICAL THINKING

Evidence of meeting the criteria under this section of the marking rubric (Critical thinking) is provided throughout this document. Table 7 indicates where.

Table 7. Summary of Critical Thinking

Criterion	Demonstration
Have you identified challenges your team encountered during the development of the prototype? Have you articulated how you overcame these challenges? Have you articulated the resources and contacts you sourced?	Articulation of the main challenges we faced, how we overcame them and what resources we used are found in the following sections: • Modelling and printing the case - see Case • Assembling the physical components Inside the case - see Physical components • Creating an engaging and useful GUI - see GUI • Designing the algorithm - see Algorithm • Implementing the GUI In Python - see Code • Creating a virutal sever and a private API to pull Instant consumption data from the sensor and store It In a database. • Managing time and workload - see Contribution
Have you articulated the key decision-making points of your project?	Articulation of key decision-making points can be found in the following sections: • Deciding to design a nudge-based system - see Theme • Deciding to use 3D printing to build the case - see Case • Deciding how the GUI should look to maximise user engagement - see GUI • Deciding on the PowerPuggle conditions and actions; the credit system and the energy saving target - see Algorithm • Deciding on the coding language for the software section - see Code • Deciding on the local database - see Code
Have you clearly justified trade-offs in your design?	Trade-offs we have justified in this document include: • Using 3D printing to create the case - see Case • Using a less powerful battery - See Physical components • Displaying too much and too little information on the GUI – see GUI • Choosing Python as the main coding language - see Code • Choosing Virtual vs. on-device storage - see Internet service • Using SQLite as the local database - see Internet service

BUDGET

Table 8. Budget

Item purchased	Cost
Core Electronic order (PowerBoost 500 Basic inc. shipping)	29.40
Platypus graphic licence	3.45
Core Electronics order (faceplate and buttons, battery, battery holders and switches inc. shipping)	40.90
Core Electronics order (PiTFT 2.8inch screen inc shipping)	85.83
MakerSpace, PLA 3D printing filament	30.00
Printing	19.47
Printing	34.76
Acrylic	40.00
TOTAL	**238.82**

CONTENT OF PRESENTATION

Table 10. Summary of content of presentation

Criterion	Where to find it
What your CPS is	The PowerPuggle and How it works
Intended audience	Summary – Intended audience
Motivation	Summary - Motivation
Scale	Scale section
Design requirements	Design requirements section
Systems analysis	Design requirements section and Systems analysis
Evidence	Design requirements section
Critical thinking	Design requirements section
Budget	Budget section
Contribution	Contribution section

CLARITY

Table 11. Summary of content of presentation

Criterion	How addressed

Is it appropriate for the audience?	Our stall design was developed with our audience in mind. Noting that people will be walking through and that Demo Day is designed to be fun, we have made sure that our stall is full of tactile elements that people can see and feel – for example, our big posters of the PowerPuggle graphics, the various iterations of the physical components and the PowerPuggle itself, which users can tickle as much as they like! In addition, in recognition of the fact that the objectives of target audience on Demo Day is quite different to the objectives of the Build team marking our prototype, we have prepared this supporting documentation, organised clearly against the marking rubric, to ensure the markers can also gain the information they need to mark our project.
Does it have a clear narrative and structure?	This document is organised according to the criteria set out in the marking rubric, thus its narrative and structure should be clear. Noting that different visitors to our stall would have different objectives, we made sure the slides we prepared and displayed on Demo Day clearly conveyed how our system worked, thus ensuring our narrative was as clear as possible.
Are the figures, diagrams, or slide layouts clear and easy to read?	We displayed three of our system maps at our stall as part of our slides. Our other main presentation device In this document Is tables. We have sought to make these as clear and easy to read as possible.
Do you define key terms?	Key terms are defined throughout this document including: ● Nudge theory (p4) ● EAST framework (p5) ● Safe (p25) ● Sustainable (p26) ● Responsible (p26)
Did you cite all relevant resources, be it literary or through interviews?	Relevant resources are cited throughout this document and collected in the references section. Interviews were not conducted as part of this project.

REFERENCES

3A Institute, Australian National University 2020, *Research,* viewed 12 October 2020, https://3ainstitute.org/research

Cyber-Physical Systems Public Working Group 2016, Framework for Cyber-Physical Systems, Release 1.0. National Institute of Standards and Technology (NIST). Accessed at https://s3.amazonaws.com/nist-sgcps/cpspwg/files/pwgglobal/CPS_PWG_Framework_for_Cyber_Physical_Systems_Release_1_0Final.pdf.

Bruckner, T, Bashmakov, Igor A, Mulugetta, Y, Chum, H, de la Vega Navarro, A, Edmonds, J, Faaij, A, Fungtammasan, B, Garg, A, Hertwich, E, Honnery, D, Infield, D, Kainuma, M, Khennas, S, Kim, S, Nimir, H B, Riahi, K, Strachan, N, Wiser, R and Zhang, X 2014, 'Energy Systems', in OR Edenhofer, Y Pichs-Madruga, E Sokona, S Farahani, K Kadner, A Seyboth, I Adler, S Baum, P Brunner, B Eickemeier, J Kriemann, S Savolainen, C Schlömer, T Stechow, Zwickel and JC Minx (eds.) *Climate Change 2014: Mitigation of Climate Change*, Contribution of Working Group III to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change, Cambridge University Press, Cambridge and New York.

Keyser, H 2018 *'10 curious facts about the platypus*', MentalFloss, accessed 7 November 2020, https://www.mentalfloss.com/article/63062/10-curious-and-quirky-platypus-facts

Product Design Online 2019, 'Fusion 360 Snap Fit Cases: 3D-Printable Raspberry Pi Case', accessed on 10 September 2020, https://www.youtube.com/watch?v=E0NVC8xhf3I

Low J (2020) '*How to design snap-fit joints for 3D printing'*, 3D Hubs, accessed on 1 September 2020, https://www.3dhubs.com/knowledge-base/how-design-snap-fit-joints-3d-printing/

Thaler, R.H. & Sunstein, C.R. 2008, Nudge: improving decisions about health, wealth, and happiness, Yale University Press, New Haven.

Wilk, J. (1999), "Mind, nature and the emerging science of change: An introduction to metamorphology.", in G. Cornelis; S. Smets; J. Van Bendegem (eds.), EINSTEIN MEETS MAGRITTE: An Interdisciplinary Reflection on Science, Nature, Art, Human Action and Society: Metadebates on science 6, Springer Netherlands, pp. 71–87, doi:10.1007/978-94-017-2245-2_6, ISBN 978-90-481-5242-1

United Nations Environment Programme, GRIDArendal and Behavioural Insights Team (2020) '*The Little Book of Green Nudges: 40 Nudges to Spark Sustainable Behaviour on Campus'*. Nairobi and Arendal: UNEP and GRID-Arendal. Accessed at https://www.bi.team/wp-content/uploads/2020/09/LBGN-2.pdf

O'Hanlon, B.; Wilk, J. (1987), Shifting contexts : The generation of effective psychotherapy., New York, N.Y.: Guilford Press.

Behavioural Insights Team, '*EAST: Four simple ways to apply behavioural insights'* https://www.bi.team/wp-content/uploads/2015/07/BIT-Publication-EAST_FA_WEB.pdf

Shroff R (2019) 'Artifical Intellignce explained in simple terms), Medium, accessed at https://medium.com/mytake/artificial-intelligence-explained-in-simple-english-part-1-2-1b28c1f762cf

Nielson 2018, Screen time still an Australian pastime, accessed at https://www.nielsen.com/au/en/insights/report/2018/screen-time-still-an-australian-pastime