Design
The payload was built into the shape of a 6” X 6” X 6” box of foam core with half inch thick foam insulation on all faces inside the box. This gave an effective working area of 5” X 5” X 5” on the inside, minus the flight, minus the flight tube which ran through the center of the top and bottom faces of the box.

Hardware
The box was heated by 3 4.7kohm ceramic resistors wired up in series powered by 2 alkaline 9V batteries wired in parallel. This setup kept the payload warm for the duration of the flight. The Raspberry pi was powered by 5V 1A 4400mAh battery pack from adafruit.com. A USB A to USB micro cord was used to connect the battery pack and the pi. The cord was spliced in order to insert a switch. The pi was secured to a piece of foam core which was then hot glued to a piece of Velcro to secure the pi to the inside of the box without damaging the pi. By cutting a slot for the ribbon cable and one for the lens, the camera module was able to fit inside a standard SD card case with the lens protruding from the front of the case. This kept the electronics relatively safe. The case was then secured to the inside of the box with tape so that the camera lined up with a hole in the box.

Software
The software that runs the camera is a simple minimalistic python script that runs when the Pi is powered on. In order to make the Pi runs the script on boot up, the crontab was modified to run the script when power is received. The crontab essentially is a program in the OS level that is in charge of running software at predetermined times since boot-up.

See link [1] for the full tutorial for crontab, but essentially type into the command line on the Pi:

```shell
sudo crontab -e
```

This will open the crontab, and then inside the crontab type the following:

```bash
@reboot sh /home/pi/.../nameofscript.py >/home/pi/logs/cronlog 2>&1
```

Save the edits to the crontab with `ctrl+x` once you are done. Remember to replace the `...` after the `/home/pi/` with the directory path of your script and replace `nameofscript.py` with the name of your script. For this project the file was named `test1.py` and was located in `/home/pi/Desktop/imageteststuff.py` so the crontab has the following line:

```bash
@reboot sh /home/pi/Desktop/imageteststuff.py >/home/pi/logs/cronlog 2>&1
```

The actual script is very easy and simple to write, remember that in python comments are denoted by a #. In essence all you need is to import `picamera` and then add the rest of the code you would like to run.

Document is continued on the next page, text moved for continuity reasons.
This code first calculates how many images to take given the interval between pictures and the time of flight in minutes. Then the resolution is set to maximum available and the camera is told to wait for a second, this is done just to give the camera sometime after turning the Pi on and before it starts to take pictures.

```python
import time
import picamera

## please note that the current image res (2592,1944) makes images
## about 2.5 MB in size, the current setup has 3.2 GB of space

camera = picamera.PiCamera() #we define camera as a PiCamera, necessary step
pic_interval = 10 #seconds between pictures
time_of_flight = 120 #please put flight time in minutes
image_count = ((120*60)/pic_interval) #counts the images to be taken
camera.resolution = (2592,1944) #sets the camera resolution
time.sleep(1)
```

We know make the code go through a for-loop that goes from x = 0 to x = image_count. We then make a string `timestr` that is composed of the time and year in the format of `year_month_days` – `hour_minute_seconds`. The camera capture line saves the image with `timestr` as the name in the directory `directory-path` and with the file extension `.jpg`. When retrieving the images of the flight, they will be in the `directory-path` used in the code.

```python
camera.capture('directory-path' + timestr + '.jpg')
```

The `time.sleep(pic_interval)` line makes the program stop for `pic_interval` seconds which was determined by the user in the previous chunk of code.

```python
try:
    for x in range(image_count):
        timestr = time.strftime("%Y_%m_%d-%H_%M_%S") #sets naming convention
camera.capture('/home/pi/Desktop/imageteststuff/images/' + timestr + '.jpg') #the line above determines where the image will be saved
time.sleep(pic_interval)
finally:
    camera.close()
```

The only current downside, is that a way to stop the running python once the Pi is powered is yet to be found. For extra information, updates on the code, and more details about the code visit link [2].

**What We Have Learned (the hard way)/Improvements**

- The pi is able to last 2 hours without being heated.
- After connecting the pi, the indicator button on the battery pack must be pressed in order to start powering the pi. The battery pack does have a current sensing circuit inside it.
- Though the camera module is more protected on the inside of the box, it is easy for it to get unaligned with the hole causing the box to obscure part of the picture. We recommend attaching the camera module to the outside of the box.
- Both the 9V batteries and the battery pack come loose very easily even with multiple pieces of Velcro.
Links