



TUSIC

**Emily, Kristen, Hannah, Maggie,
Athmika**

shoutkey.com/queasy

Our Goal

A **PAGE TURNER** for musicians that is actuated with a **HANDS-FREE** method.

OUR MVP

1

Can reliably
turn sheet
music in a
binder format

2

Hands-free
trigger
mechanism

3

Professional
Construction

4

User testing
with pianists

5

Integrated
systems

LEARNING GOALS

Our individual goals.
How have we met them
thus far?

EMILY

System
integration,
program
hands-free
inputs

HANNAH

CAD, rapid
prototyping,
mechanisms,
system
integration

KRISTEN

System
integration,
Open-source
programming,
Prototyping

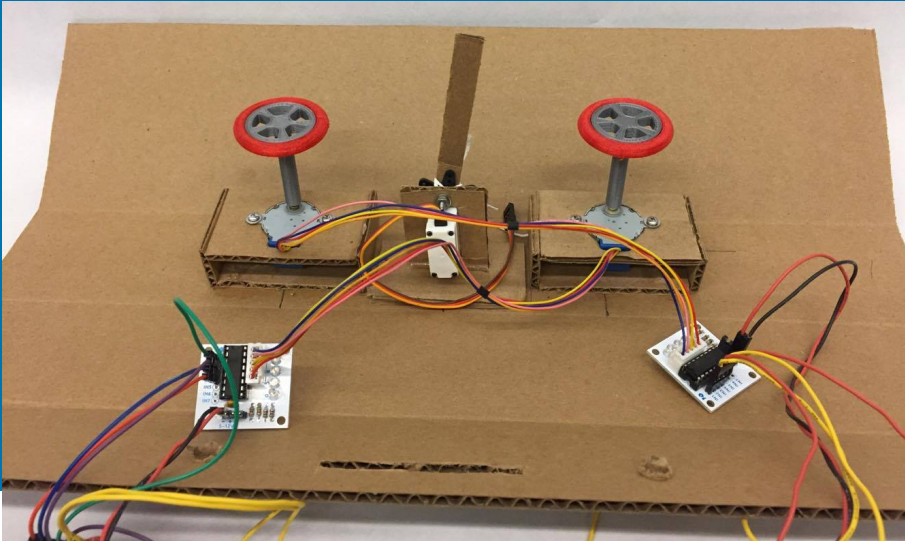
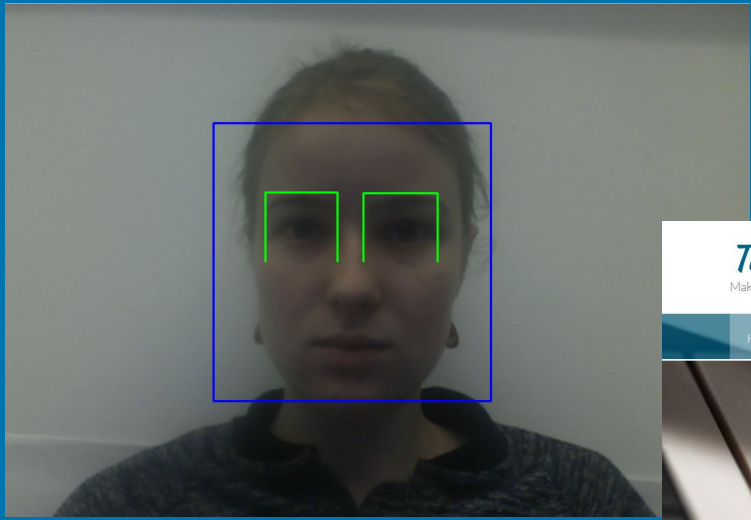
ATHMIKA

Writing more
readable code

MAGNOLIA

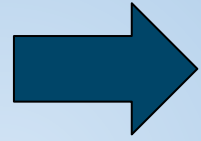
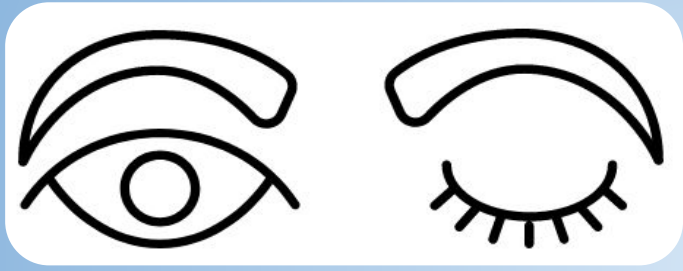
Tidier circuits,
Electrical
troubleshooting

Sprint 1 Deliverables

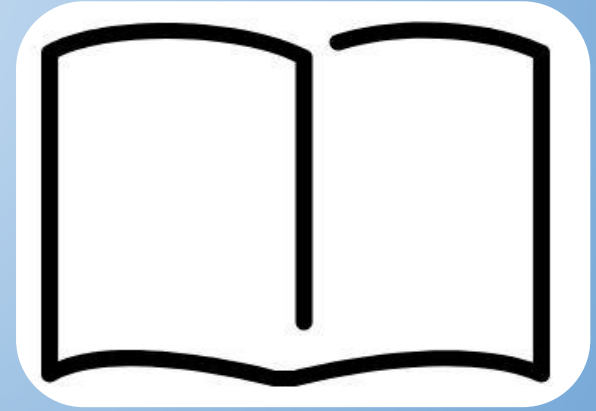
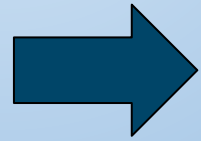
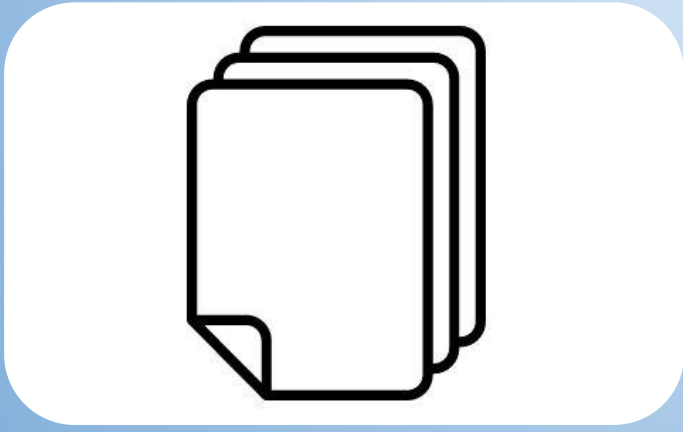


Sprint 1 Takeaways

SOFTWARE

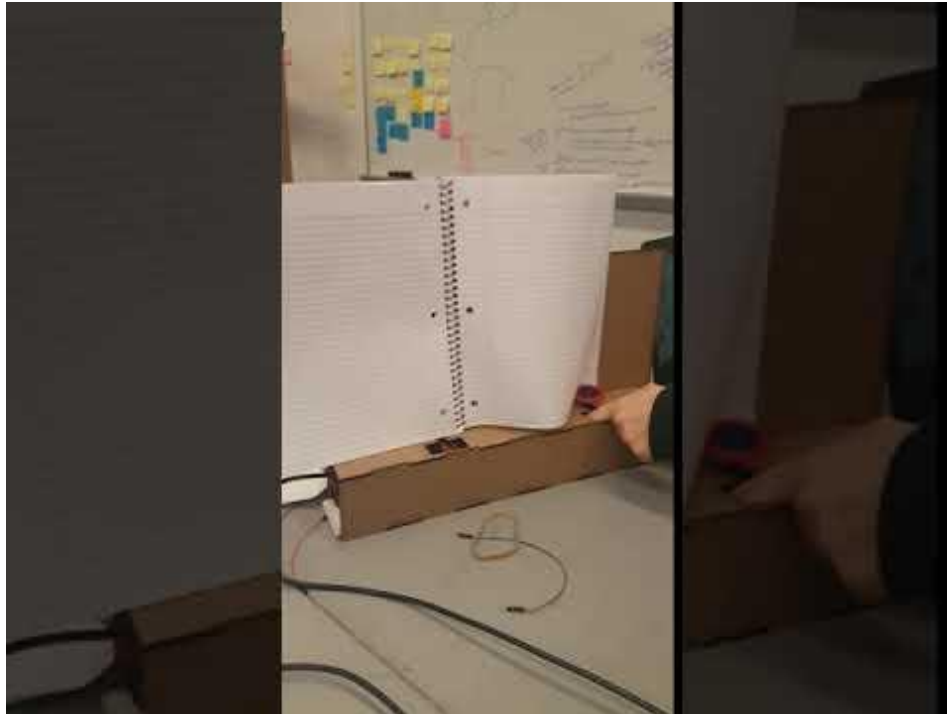


MECHANICAL



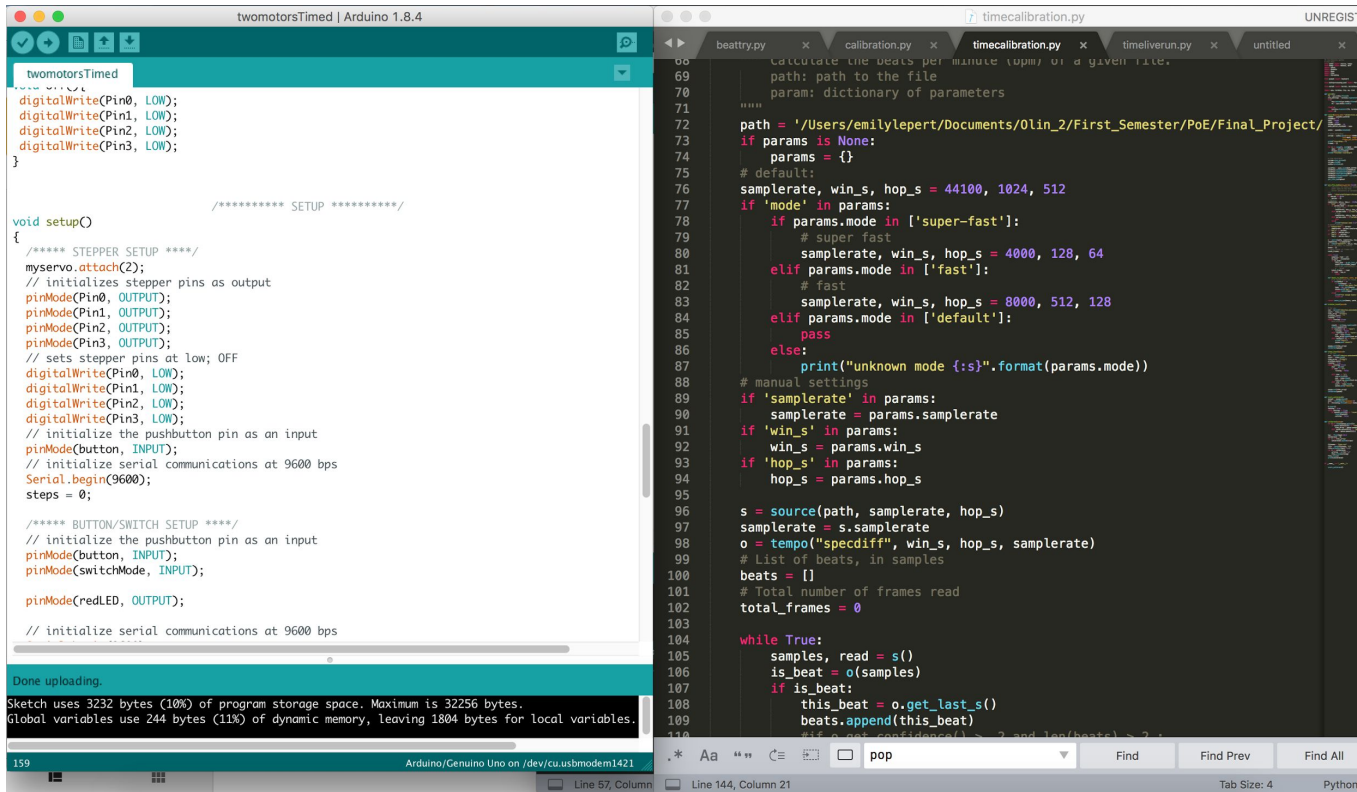
Deliverable 1

Moving one piece of paper



Deliverable 2

Beat Detection



The image shows two IDE windows side-by-side. The left window is the Arduino IDE, titled 'twomotorsTimed | Arduino 1.8.4'. It contains C++ code for an Arduino sketch. The right window is a Python IDE, titled 'timecalibration.py'. It contains Python code for a beat detection algorithm.

```
twomotorsTimed
digitalWrite(Pin0, LOW);
digitalWrite(Pin1, LOW);
digitalWrite(Pin2, LOW);
digitalWrite(Pin3, LOW);
}

void setup()
{
  //***** SETUP *****/
  //***** STEPPER SETUP *****/
  myservo.attach(2);
  // initializes stepper pins as output
  pinMode(Pin0, OUTPUT);
  pinMode(Pin1, OUTPUT);
  pinMode(Pin2, OUTPUT);
  pinMode(Pin3, OUTPUT);
  // sets stepper pins at low; OFF
  digitalWrite(Pin0, LOW);
  digitalWrite(Pin1, LOW);
  digitalWrite(Pin2, LOW);
  digitalWrite(Pin3, LOW);
  // initialize the pushbutton pin as an input
  pinMode(button, INPUT);
  // initialize serial communications at 9600 bps
  Serial.begin(9600);
  steps = 0;

  //***** BUTTON/SWITCH SETUP *****/
  // initialize the pushbutton pin as an input
  pinMode(button, INPUT);
  pinMode(switchMode, INPUT);

  pinMode(redLED, OUTPUT);

  // initialize serial communications at 9600 bps
}

Done uploading.
Sketch uses 3232 bytes (10%) of program storage space. Maximum is 32256 bytes.
Global variables use 244 bytes (11%) of dynamic memory, leaving 1804 bytes for local variables.
```

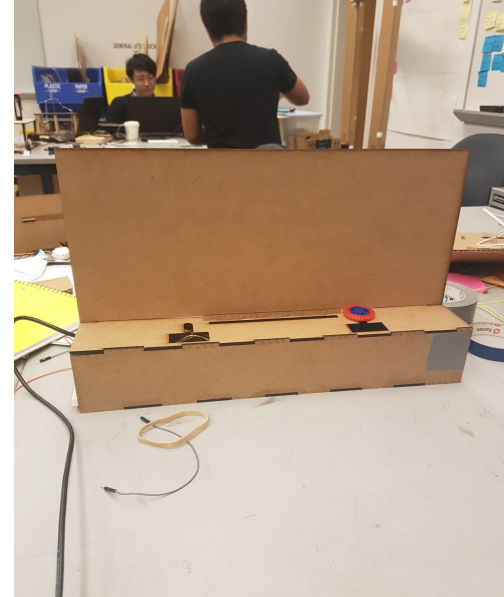
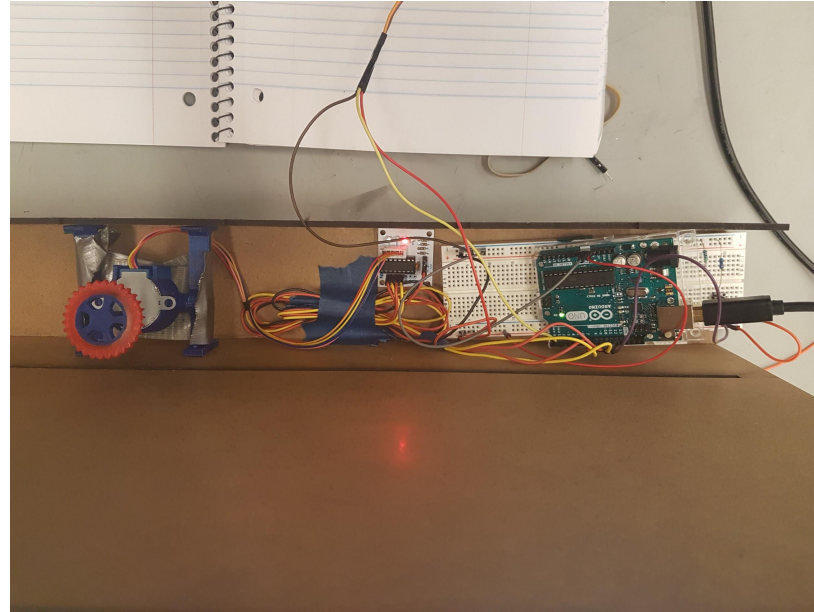
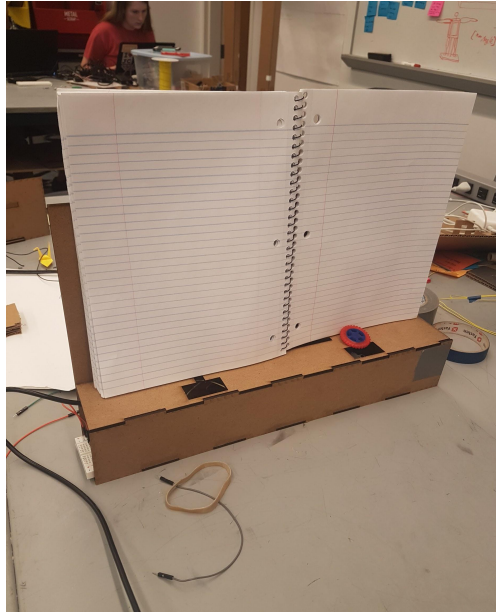
```
timecalibration.py
def calculate_beats_per_minute(tempo) of a given file.
    path: path to the file
    param: dictionary of parameters
    """
    path = '/Users/emilylepert/Documents/Olin_2/First_Semester/PoE/Final_Project/
    if params is None:
        params = {}
    # defaults:
    samplerate, win_s, hop_s = 44100, 1024, 512
    if 'mode' in params:
        if params.mode in ['super-fast']:
            # super fast
            samplerate, win_s, hop_s = 4000, 128, 64
        elif params.mode in ['fast']:
            # fast
            samplerate, win_s, hop_s = 8000, 512, 128
        elif params.mode in ['default']:
            pass
        else:
            print("unknown mode {}".format(params.mode))
    # manual settings
    if 'samplerate' in params:
        samplerate = params.samplerate
    if 'win_s' in params:
        win_s = params.win_s
    if 'hop_s' in params:
        hop_s = params.hop_s

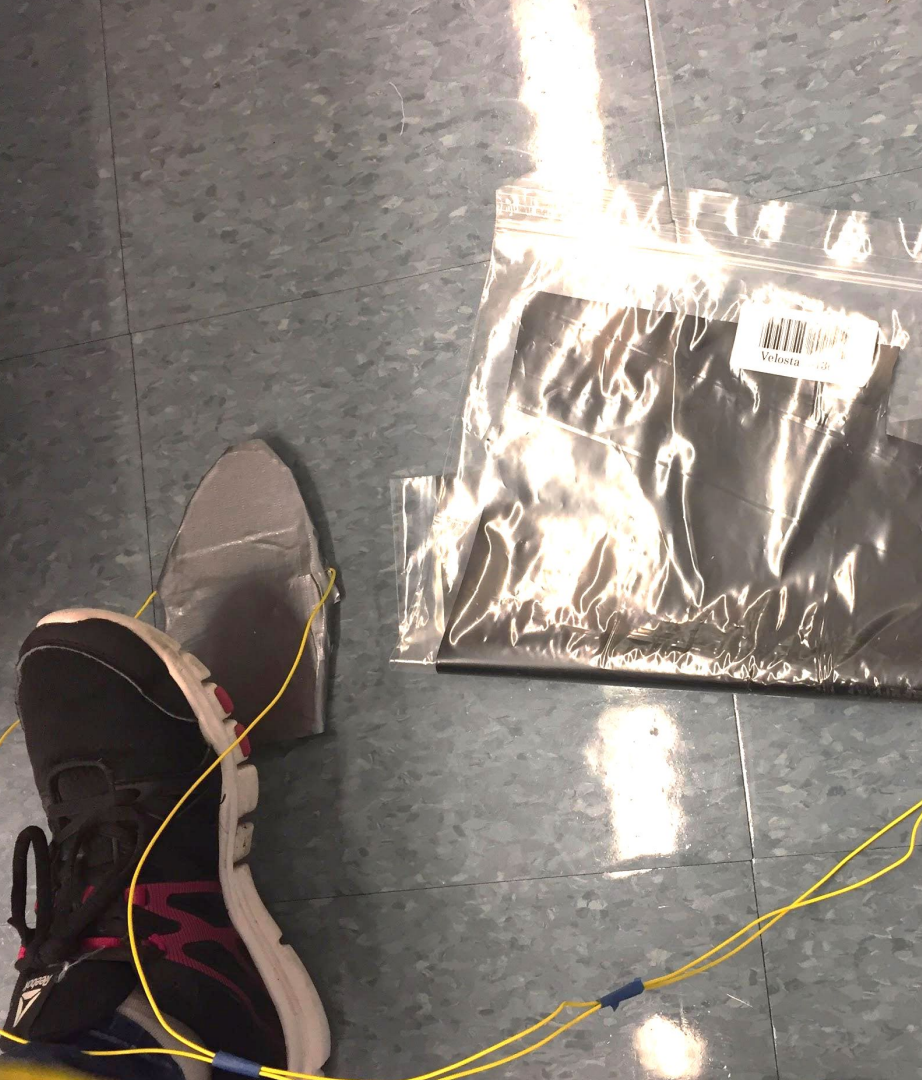
    s = source(path, samplerate, hop_s)
    samplerate = s.samplerate
    o = tempo("specdiff", win_s, hop_s, samplerate)
    # List of beats, in samples
    beats = []
    # Total number of frames read
    total_frames = 0

    while True:
        samples, read = s()
        is_beat = o(samples)
        if is_beat:
            this_beat = o.get_last_s()
            beats.append(this_beat)
            #if o.get_confidence() > 2 and log(len(beats)) > 2:
```


Deliverable 3

Integrate Beat Detection with motors





Deliverable 4

First-cut pedal using
pressure sensing paper

Deliverable 5

Continued blogging and updates

turning as the biggest risk, our next spring goals did not reflect those goals with the biggest risk to tackle those head on.

[Attached slides]

Sprint 1 Recap:

Our first sprint consisted of us ideating on a project and coming up with initial prototypes. On the mechanical side we were inspired by examples from YouTube and decided to use a system of wheels to slide pages across the music stand. We created a sketch model with stepper motors. By the end of the sprint the motors turned individually but not very fast at the same time. On the hardware side, we integrated an Arduino and button into the mechanical design to trigger the motors.

On the software side, we spent much of the sprint investigating and researching different hands free inputs into the system. We decided to prototype blink detection using Python and

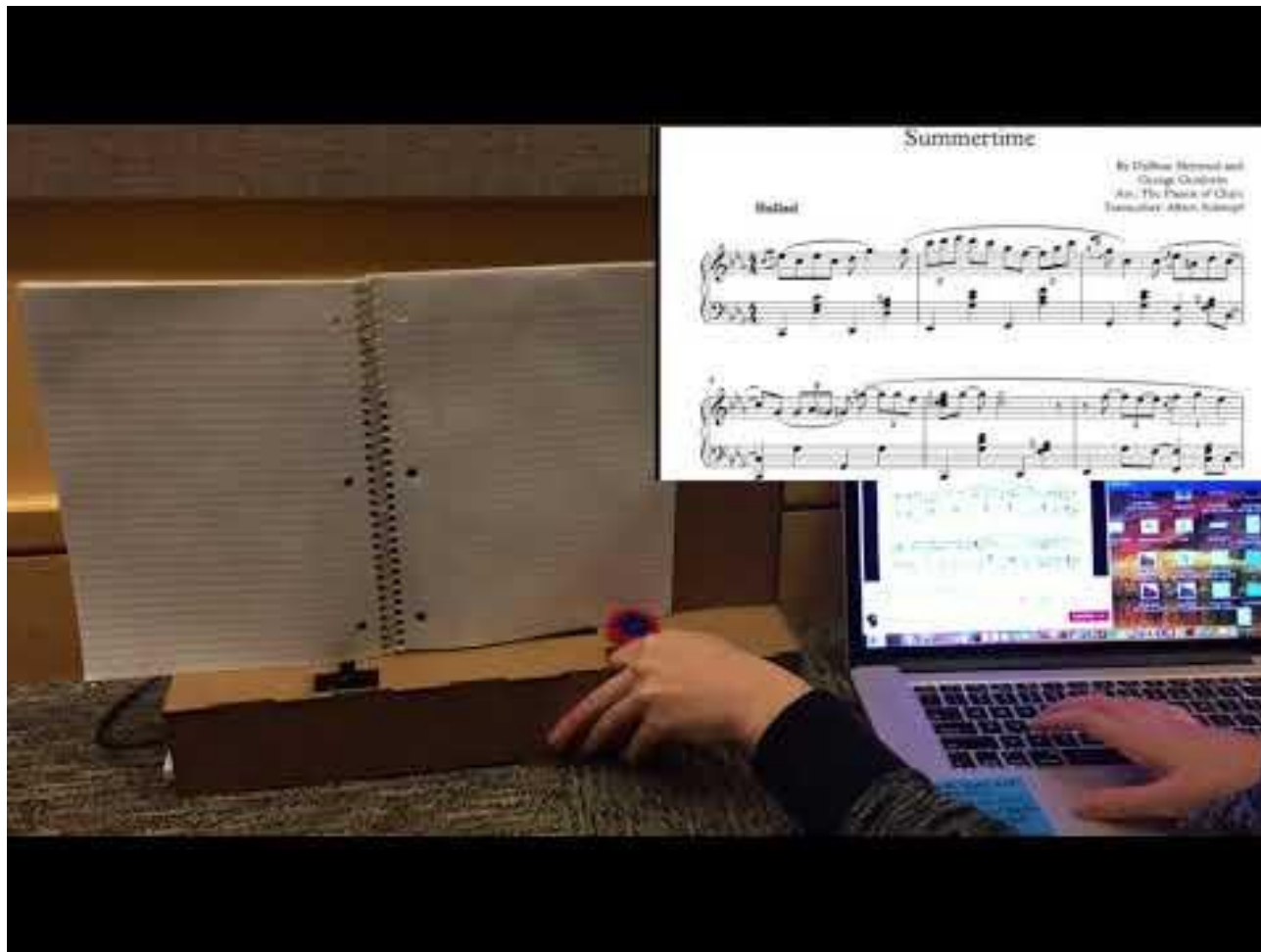
Week4/Sprint 2 Recap:

We spent more time this work working on creating a box that would be able to hold all our motors and circuitry. This proved to be pretty difficult since we are using 5 our design and two of the five need to rotate in and out of the plant. Our code to run the motors was acting up, so instead of using a library like we've been doing, we instead code which powers the coils of our stepper in the specific sequence needed to make going in a counterclockwise manner. Our steppers ran a lot more smoothly, but, as a not making this switch sooner, we need to redo our calibrations. We also received our Velostat, a material that is electrically conductive whose resistance is inversely proportional to the force applied to it. We sandwiched a sheet between two pieces of cardboard with tape taped on, in order to create our pedal prototype.

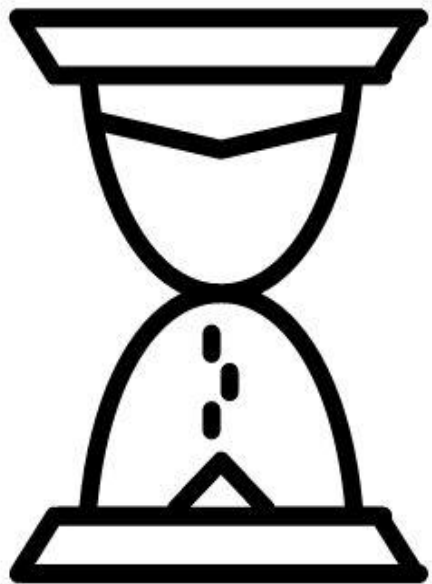
Week 3 Recap:

Hardware and mechanically this week was spent evaluating our motors. We talked with Stan about the pros and cons of different stepper motors and DC motors. The stepper we were using for our prototype turn pages well, but slowly. As such, we decided to

Demo



BIGGEST RISKS



Turning accuracy



Beat detection accuracy

3rd SPRINT GOALS

Mechanical

- Implement servo swiper
- Include springs to push motors
- Test/calibrate turning accuracy
- Have reverse functionality

Software

- More testing of beat detection code
- Investigate other possible libraries

Electrical

- Make pedal input instead of button/keyboard input

Documentation

- Get website live
- Continue blogging