Validation and Verification

**Test 1: Digital Flute mechanism**

**Date**: 14th of february 2025

**Goal**: Test whether the Digital Flute outputs 4 different frequencies as expected when exposed to 4 different distances.

**Subsystems** **involved**: Digital Flute, emergency stop mechanism

**Testers**: Clementine, Mathew

**Purpose of test:**

Accurate sensor readings are critical for effective operation of the instrument. For the digital flute to be usable, the ultrasonic sensor must consistently detect the position of the user’s hand. Extensive testing of the accuracy and characteristics of the sensor has already been performed, so only basic integration testing was done to ensure that all four notes could be played.

**Test Procedure**:

1. Equipment setup
   1. Connect a touch sensor to port 1, this is the play button.
   2. Connect the ultrasonic sensor to port 2, this is the sensor for the flute mechanism.
   3. Connect the second touch sensor to port 3, this is the emergency stop.
   4. Run the script *flute\_test.py* from the project folder
2. Data collection
   1. Place your hand at x distance away from the touch sensor once for x taking each of the following values [10,20,30,40]
   2. Press stop
   3. Verify file has been saved to .csv
3. Graph generation
   1. Copy the color\_sensor.csv file from the robot to the same folder on your computer using FileZilla
   2. Run python3 music\_sensor\_visualization.py. (This will not work if Matplotlib is not downloaded on your computer.) This should generate a scatter plot.
   3. Save the graph as an image to your computer.

**Test data:**

Table X: Representation of sound played depending on distance sensed by tester

Graph X: Representation of sound played depending on distance sensed by tester

**Test conclusions:**

As we can see from the recorded data and more clearly from the scatter plot, the expected intervals map to the expected sound. That is C4 -> , D4 -> , E4 -> , F4 -> . During the experiment we noticed the sound was not as perceptible as we would hope. We conclude that we will integrate a speaker into our design.

## **Test 2: Drumming mechanism**

**Date**: 21st of February 2025

**Goal**: Testing the hardware for our drumming mechanism validating both sound and rhythm

**Subsystems involved**: Drumming mechanism, Emergency stop

**Tester**: Clementine

**Test Procedure**:

1. Equipment setup
   1. [Insert mounting description]
   2. Run *drum\_test.py* as seen below.
2. Part 1:
   1. Register time of each percussion
   2. Record start time
   3. Save in table form to ensure consistency (confirm durability)
3. Part 2:
   1. Place the first percussion material under the drumstick.
   2. Run the **drumming mechanism** for **10 seconds** at a fixed striking force.
   3. Record the **average decibel level (dB)**.
   4. Repeat steps 1–3 for each percussion material.
   5. Compare the recorded decibel levels to determine the loudest and quietest materials.

**Test data**:

Table X: Table representing the frequency of the percussion to verify consistency.

| DB (average) | Material |
| --- | --- |
|  | Plastic |
|  | Wood |
|  | Benchtop |

Table: representing the average percussion sound per material

**Test conclusions:**

Try different speeds and catalog frequency of the ball hitting the surface. Add qualitative observations The drumming mechanism is activated by a button press, causing a motorized arm with a steel ball bearing to strike a drum surface, generating a percussive sound at regular intervals.In conclusion, we have tested multiple different drumming candidates and have found the one which meets most our specifications.Impact on iterative design process

​​

Test 3: Full system integration

**Date**: 21st of February 2025

**Goal**: Validate the fully integrated system, ensuring both software and hardware components function correctly.

**Subsystems involved**: Flute mechanism, Drumming mechanism, Emergency Stop mechanism

**Tester**: Clementine

**Test Procedure**:

1. Equipment setup
   1. Connect the touch sensor to port 1. This will be the start/stop for the drumming mechanism.
   2. Connect the ultrasonic sensor to port 2. This will be the sensor for the flute mechanism.
   3. Connect the second touch sensor to port 3. This will be the emergency stop.
   4. Ensure drumming mechanism is properly mounted
2. Data collection
   1. Validation 1: Drumming mechanism output
      1. Run the script *integration\_test.py* from the project folder
      2. Record the frequency of sound to ensure rhythm remains consistent
      3. Engage emergency stop mechanism
   2. Validation 2: Flute mechanism output
      1. Restart the system and run the script *integration\_test.py* from the project folder
      2. Place your hand at x distance away from the touch sensor. Test each note once.
      3. Engage emergency stop mechanism
      4. Verify file has been saved to .csv
   3. Validation 3: Combined drumming mechanism and Flute mechanism output
      1. Restart the system and run the script *integration\_test.py* from the project folder
      2. Place your hand at x distance away from the touch sensor. Test each note once.
      3. Engage emergency stop mechanism
      4. Verify file has been saved to .csv

**Test data** (in table form):

| Distance from touch sensor | Sound played | Drum | Emergency stop |
| --- | --- | --- | --- |
|  |  |  | OFF |
|  |  |  | OFF |

Table representing the observed data

**Test conclusions:**