### Team D3 - WoodWindMania

Angel Peprah, Vivian Beaudoin, Judenique Auguste Add your 12 slides after this slide... [remember, 12 min talk + 3 min Q/A]

See <u>https://gsuite.google.com/learning-center/products/slides/get-started/</u> for how to import slides

Make sure to cover: (refer to the Final Presentation Guidance):

- Use Case / Application and Primary (Quantitative) Requirements (i.e. a reminder from prior presentations)
- Solution Approach a reminder (include updates from Design Review presentation if changed)
  - E.g. block diagram(s), flow chart(s), schematic(s)
- System Implementation your complete solution
  - E.g., pictures, screenshots, video (make sure that there is CMU access to play any media)
- Testing, Verification and Validation with quantitative metrics and target values to compare with experiment
  - What tests did you run ? How many tests ? What were the results ?
  - Graphs, tables, quantitative results (compare with the metric targets & ultimately use-case requirements)
- Project Management tasks, division of labor, and schedule
- Lessons Learned

Consider that this slide already works as a introduction slide so use your first slide wisely (i.e. feel free to delete guidance text)

### **Use Case**

- **Problem**: Learning woodwind instruments has a high cost associated with it and may not be practical to do in most environments
  - Lessons plus Instrument, can cost around \$1000
  - Not beginner friendly
- **Solution**: A digital woodwind learning tool that allows users to learn fingerings on a realistic woodwind controller, in this case, a flute
- Areas: Hardware Design & Software Systems



WoodWindmania

musical learning starts here

### Requirements

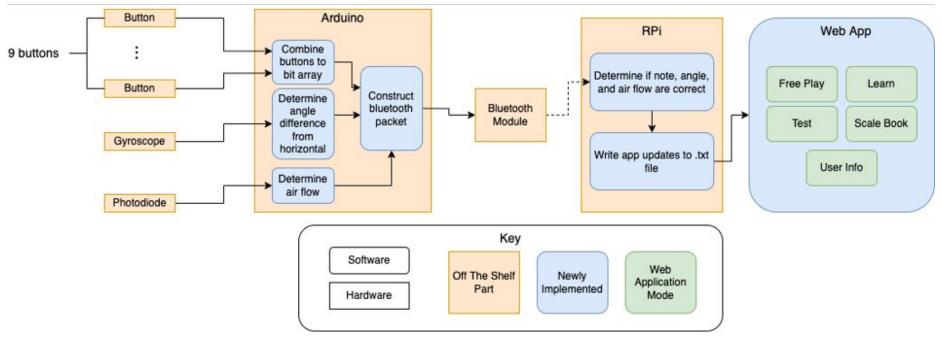
#### • User Experience

- Comparison to real instrument
  - Physical dimensions: 1.3lbs, 26"x1"
  - User satisfaction: 4/5
- Beginner friendly
- Portability
- Accuracy
  - Note feedback, orientation feedback, breath control feedback >= 90%
- Speed
  - Latency <= 500ms





### **Solution Approach**



## **Complete Solution**

#### • Physical controller

- 26" PVC pipe
- 9 push buttons to detect notes
- Breath sensor to detect between high and low octave
  - Obstacle detection using LED, photodiode, and a piece of latex to obstruct light
- Arduino Nano 33 BLE for angle detection and bluetooth communication
- 9V battery





# **Complete Solution**

#### • Raspberry Pi

- Data from controller sent via BLE to Pi
  - Connection stable from across the room
- Pi determines note, plays the correct flute sound off of SonicPi
- Writes buttons pressed, angle from horizontal, and breath sensor value to text file for web app to read

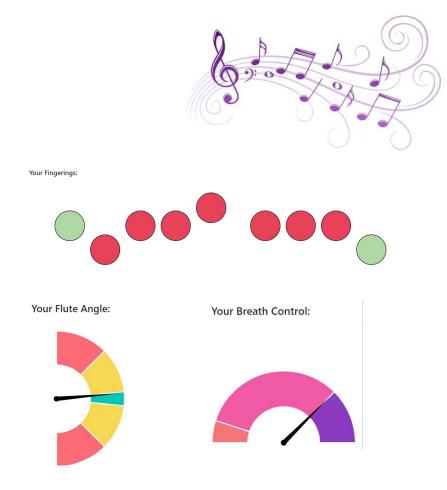




# **Complete Solution**

#### • Web Application

- Django (Python based web framework) running locally
- RPi continually writes the flute controller data to a text file
- The controller in MVC reads and interprets the text file
- Web app will update with new values on a constant interval





Requirement	Metric	Testing Strategy
Accurate Feedback	Note, orientation, breath control feedback >= 90% accuracy	Give correct/incorrect notes/orientations and determine feedback Use correct/incorrect breath control, comparing to an actual flute
Speed	<= 500ms	Record bluetooth latency and web app latency separately
User Experience	1.3lbs, 26"x1" 80% user satisfaction	Measure weight, width, length Survey beginners on user experience

- Accuracy
  - Note feedback
    - Give correct and incorrect note fingerings
    - Average accuracy over 10 trials for each note in a scale
  - Orientation feedback
    - Use a level to test 5 degrees away from the horizontal in both directions
    - Average difference over the 10 trials
  - Breath control feedback
    - Blow soft/hard enough to play a lower/higher octave on a flute into the controller
    - Average accuracy over 10 trials



Requirement	Metric	Result
Note feedback	>= 90%	100%
Orientation feedback	>= 90%	TBD
Breath control feedback	>= 90%	TBD

#### • Speed

- Ignore latency from sensors to Arduino, focus on Bluetooth and web app latency
- Bluetooth latency
  - RPi sends data to the Arduino, the Arduino responds right back
  - Average the latency over 10 trials
- Web app latency
  - RPi writes the current time in the text file
  - Average the latency over 100 trials



Requirement	Metric	Result
Bluetooth Latency	<= 100ms	96.95ms
Web App Latency	<= 400ms	269ms



### • User Experience

- Dimensions
  - Measure length, width, and weight of device
- User satisfaction
  - Survey 10 beginners
  - Questions, rating from 1-5:
    - Comparison to a real flute
    - Comfort of pressing buttons
    - Comfort of breath control
    - Overall comfort

Requirement	Metric	Result
Length	26"	29" with breadboard
Width	1"	1 ¾"
Weight	1.3lbs	TBD
User satisfaction	4/5	TBD

### **Trade-Offs**



#### • Buttons

- We selected buttons with similar feel to flute buttons
- $\circ$  Limited options at this size, so not the easiest to press

### • Audio Latency

- Reasonable audio latency ~20ms, average Bluetooth latency 100-300ms
- Prioritized wireless controller over audio latency

### • Length

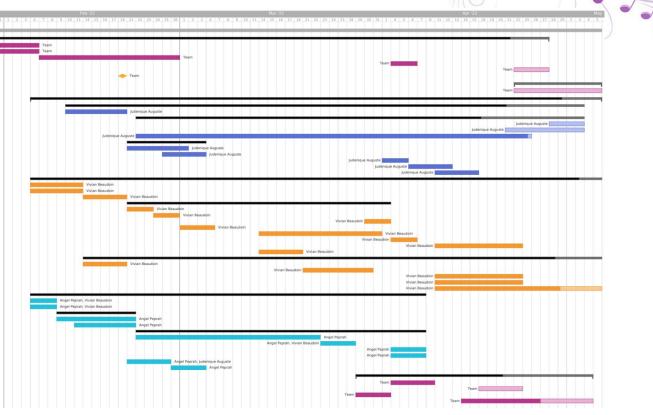
- Arduino Nano and battery mounted at the end of the controller, makes it longer than 26"
- Easier to fix wires, upload code, change batteries
- Length at the end of the flute isn't important because user doesn't hold there

#### • Width

- Outer diameter of PVC is <sup>3</sup>/<sub>8</sub>" larger than the flute
- Prioritized fitting components inside over this small difference

### **Project Management**

WoodwindMania	Oh	88%
Project Planning/Presentations	Oh	93%
Research	0	100%
Project Proposal	0	100%
Design Review Interim Demo	0	100%
Interim Demo Final Presentation	0	100%
	0	0%
Figure out all equipment and order it		
Slack Slack	0h 0	0% 0%
Components	Ob	93%
WebApp	Oh	85%
Design layout	0	100%
Implement django model view	Oh	77%
Complete Play Mode	0	0%
Complete Test Mode	0	0%
Complete Learn Mode	0	99%
Create fingering chart and sca	Oh	100%
Design fingering and scale layout	0	100%
Implement fingering and scales	0	100%
Set up Apache	0	100%
Create fake user with test inputs	0	100%
Test user database access	0	100%
Physical Controller	Oh	96%
Determine essential fingerings	0	100%
Determine the location and num of	0	100%
Order buttons and sensors	0	100%
Buttons	Oh	100%
Wire a couple buttons	0	100%
Test/calibrate buttons	0	100%
Wire all buttons and test	0	100%
Initial testing for breath control	0	100%
Calibrate breath control sensor	0	100%
Make breath sensor to fit inside c	0	100%
Make breath sensor adjustments	0	
Wire up gyroscope plus calibration Connect sensors to the appara	Oh	100%
Determine PVC pipe/structure to	0	100%
Drill holes to connect buttons	0	100%
Attach gyroscope	0	100%
Attach breath sensor	0	100%
Test/adjust	0	75%
Communication	Oh	100%
Determine Arduino and Pi devices	0	100%
Determine bluetooth device	0	100%
Setting up Raspberry Pi	Oh	100%
Install OS	0	100%
Enable Bluetooth	0	100%
Work with dummy info for Pi S	Oh	100%
Send dummy Bluetooth data fr	0	100%
Determine what note the user is	0	100%
Processing breath control info for	0	100%
Determine position from the gyros	0	100%
Determine how to/send data from	0	100%
Look into playing note off the pi	0	100%
Integration	Oh	62%
Send correct information from senso	0	100%
Testing the accuracy and speed bet	0	0%
Integrate RPi and web app	0	100%
Combine all components and test	0	60%



9: 0

0