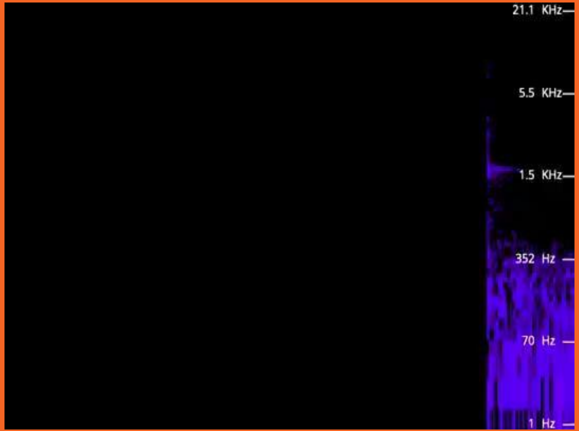
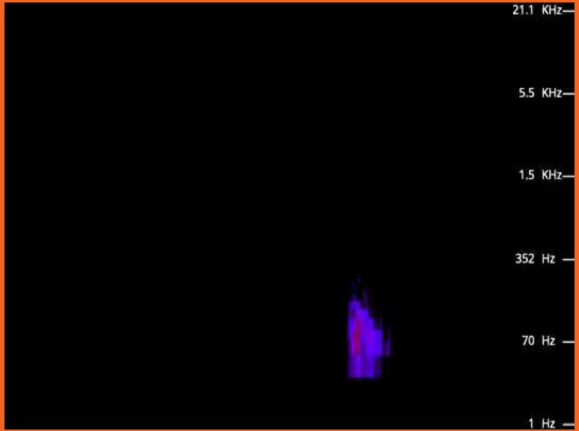
‘Sing us a song, you’re the piano pi’

Talking Piano

Team B2
Angela Chang, John Martins, Marco Acea

Motivation


21.1 KHz
5.5 KHz
1.5 KHz
352 Hz
70 Hz
1 Hz


21.1 KHz
5.5 KHz
1.5 KHz
352 Hz
70 Hz
1 Hz

Use-Case

- Traditional piano performances are limited by the physical capability of the performer. There is an entire realm of sounds that exist outside the notes reachable by our ten fingers.

ECE Areas: Software Systems, Signals and Signals

Use Case Requirements

Audio Processing

- Convert human speech to encoded piano notes
  - Less than 20% of frequencies missing from original speech file
  - Since not all frequencies are mapped to by keys, we still hope to catch the majority

Note Scheduling

- Serialize encoded piano notes to the piano interface
  - 95% of all notes correctly streamed with minimal data loss
  - We hope to minimize loss of the data stream through the sending process

Use Case Requirements Cont.

Piano Interface

- Play the scheduled notes on a piano
  - 80% fidelity rate measured from assessing the accuracy of identifying correct input speech by listening to the output piano
  - 95% playback accuracy from the inputted bit stream of notes

Web Interface

- Support user control for piano playback
  - Less than 200ms lag time between pressing pause/play and the reaction from the system

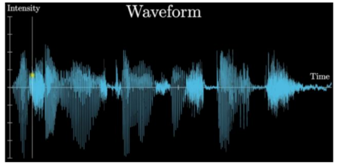
Technical Challenges

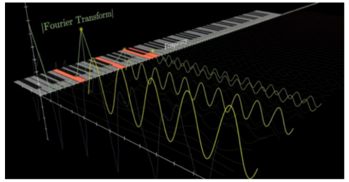
- Generating Piano notes from human speech
- Isolating voice in high Signal to Noise ratio environments
- Solenoid control that mimics human precision
- Speed and Accuracy from scheduler
- Transmitting data from web app to interface
- Creating a network system for every component
- Physical Hardware Circuitry / Control

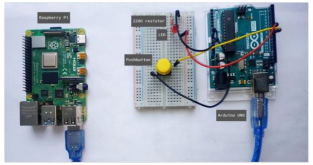
Solution Approach

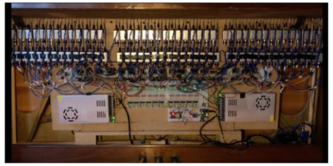
The are four major areas of development:

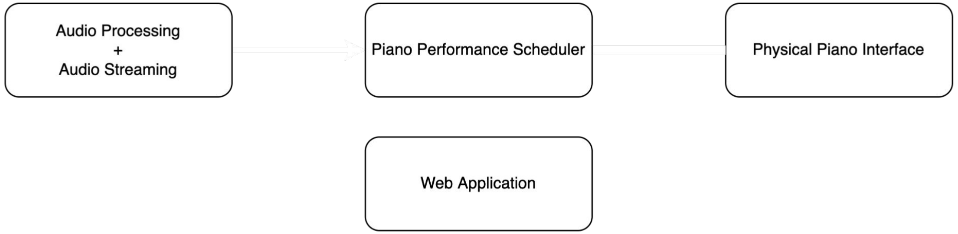
- Audio Processing
  - Python
  - Numpy, Pydub
  - Fast Fourier Transform
  - Kafka Streams
- Piano Performance
  - STM32
  - Low-Level Sockets Interface
  - PWM Solenoid Control
  - Real-Time Scheduling

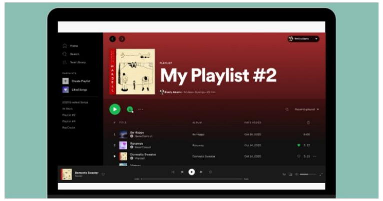








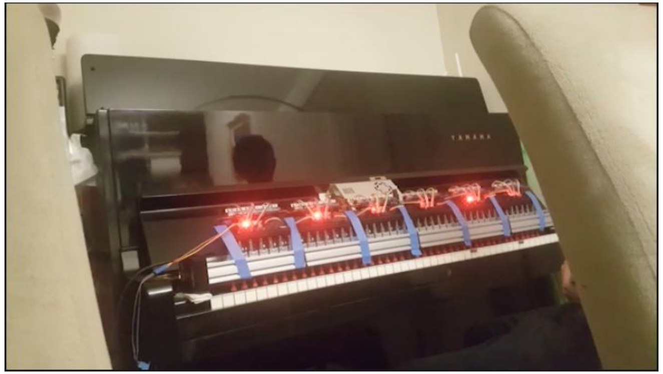

Audio Processing + Audio Streaming
Piano Performance Scheduler
Physical Piano Interface
Web Application

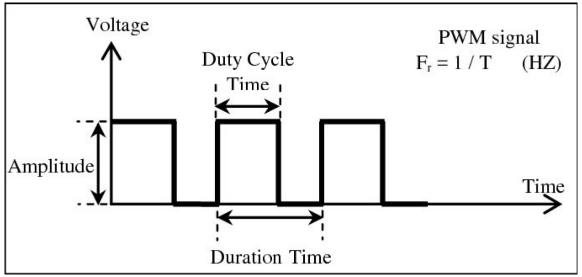


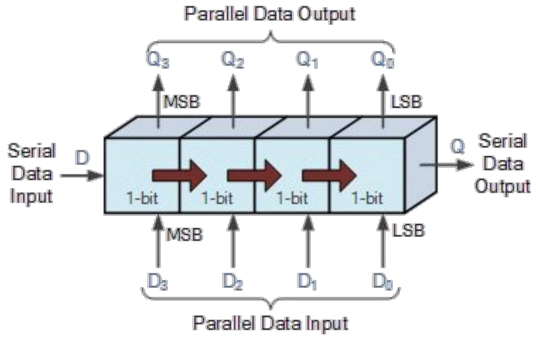
Solution Approach

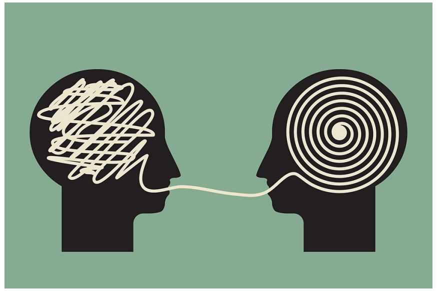
The are four major areas of development:

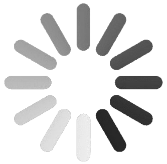
- Web Application
  - Django Backend (Python)
  - Bootstrap for UI components
  - AJAX JSON Interface
  - Kafka Streams
- Physical Piano Interface
  - STM32
  - PWM Solenoid Control
  - Transistor Switch connected to a shift register

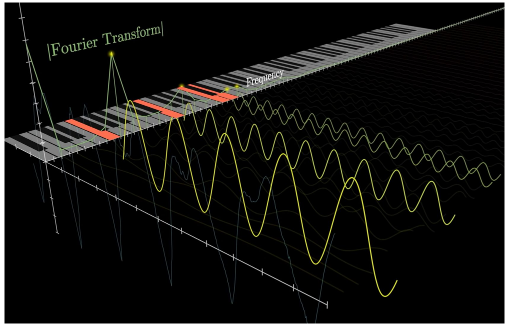












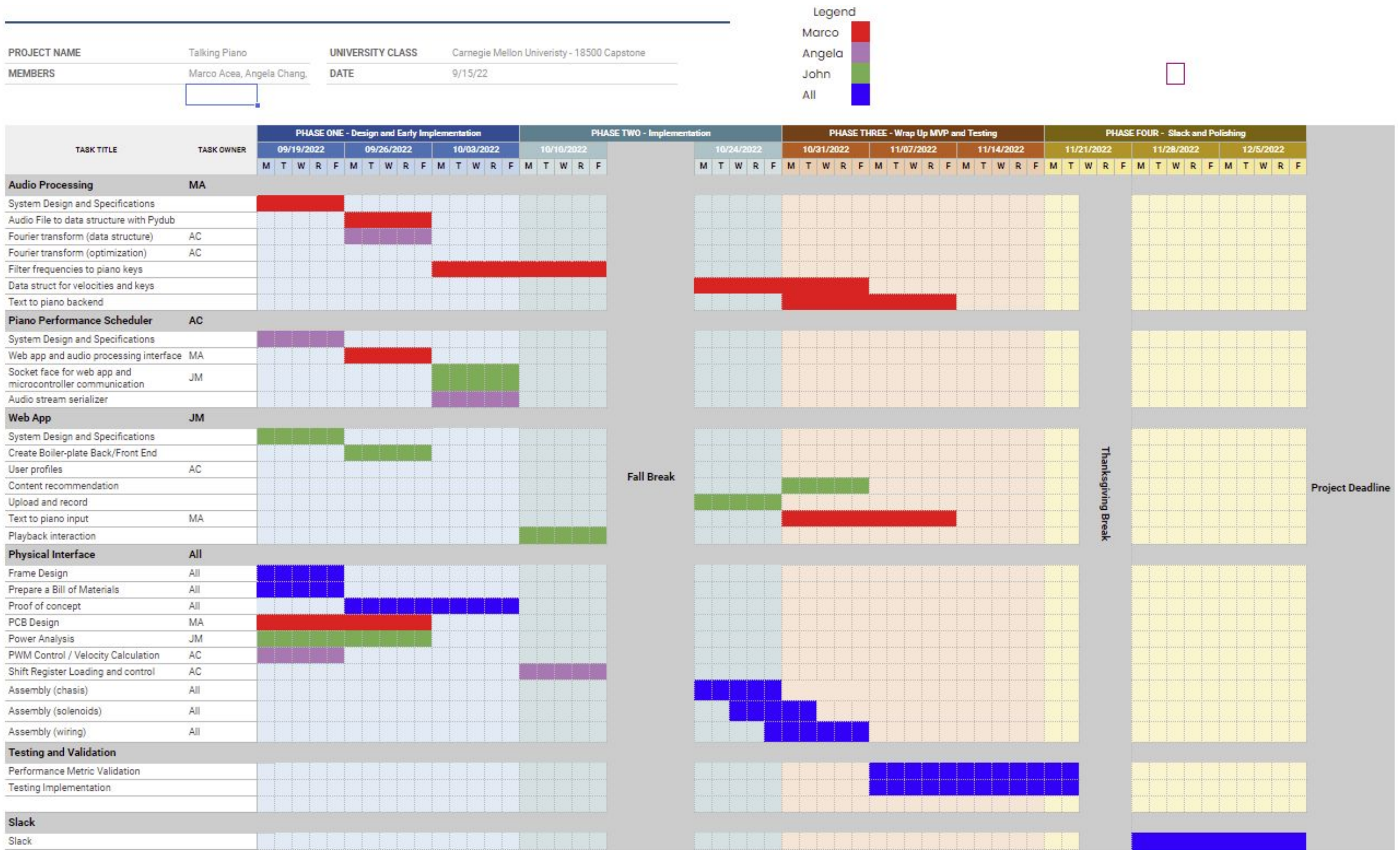
Testing, Verification, Metrics

- Fidelity of Output Audio: Surveying a group of listeners and recording how often the original prompt can be made out from the piano notes.
- Processing Speed: Measure end-to-end processing delay using input recording of varying sizes
- Loss: Comparing the piano performance to the original speech input to calculate the percentage of frequencies lost.

Tasks, Division of Labor

- Each member leads one major area of development while we work together to implement the physical interface
- Audio Processing (Marco Acea)
- Piano Performance Scheduler (Angela Chang)
- Web Application (John Martins)
- Physical Interface (All)
    - PCB Design and Frame Design (Marco)
    - Circuit Analysis (John)
    - Firmware (Angela)

Gantt Chart



PROJECT NAME	Talking Piano	UNIVERSITY CLASS	Carnegie Mellon Univeristy - 18500 Capstone
MEMBERS	Marco Acea, Angela Chang,	DATE	9/15/22

Legend	
Marco	🟥
Angela	🟪
John	🟩
All	🟦

		PHASE ONE - Design and Early Implementation			PHASE TWO - Implementation			PHASE THREE - Wrap Up MVP and Testing			PHASE FOUR - Slack and Polishing		
TASK TITLE	TASK OWNER	09/19/2022	09/26/2022	10/03/2022	10/10/2022		10/24/2022	10/31/2022	11/07/2022	11/14/2022	11/21/2022	11/28/2022	12/5/2022
Audio Processing	**MA**												
System Design and Specifications		🟥											
Audio File to data structure with Pydub			🟥										
Fourier transform (data structure)	AC		🟪										
Fourier transform (optimization)	AC												
Filter frequencies to piano keys				🟥	🟥								
Data struct for velocities and keys							🟥	🟥					
Text to piano backend								🟥	🟥				
Piano Performance Scheduler	**AC**												
System Design and Specifications		🟪											
Web app and audio processing interface	MA		🟥										
Socket face for web app and microcontroller communication	JM			🟩									
Audio stream serializer				🟪									
Web App	**JM**												
System Design and Specifications		🟩											
Create Boiler-plate Back/Front End			🟩										
User profiles	AC												
Content recommendation						Fall Break		🟩			Thanksgiving Break		Project Deadline
Upload and record							🟩						
Text to piano input	MA							🟥	🟥				
Playback interaction					🟩								
Physical Interface	**All**												
Frame Design	All	🟦											
Prepare a Bill of Materials	All	🟦											
Proof of concept	All		🟦	🟦									
PCB Design	MA	🟥	🟥										
Power Analysis	JM	🟩	🟩										
PWM Control / Velocity Calculation	AC	🟪											
Shift Register Loading and control	AC				🟪								
Assembly (chasis)	All						🟦						
Assembly (solenoids)	All						🟦	🟦					
Assembly (wiring)	All						🟦	🟦					
Testing and Validation													
Performance Metric Validation									🟦	🟦	🟦		
Testing Implementation									🟦	🟦	🟦		
Slack													
Slack												🟦	🟦

Conclusion

MVP

- Audio Processing Library with support for real-time conversion
- Real-Time Performance Scheduler
- Web Application
- Physical Piano Interface with 88-key support

Gain

- Insight into the possibilities of electro-mechanical musical systems
- Insight into a simplified model of how the human brain processes speech