Aleks | Aner | Axel | Cuong | Joe | Thomas

Business Requirements Document

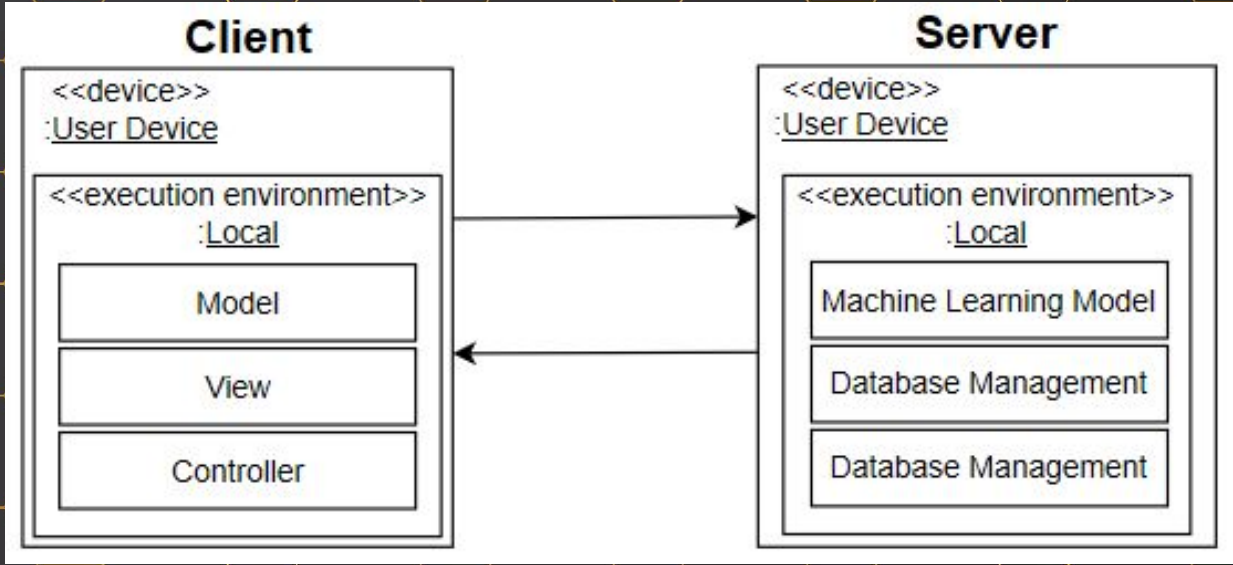
- We did not have any significant updates in our BRD

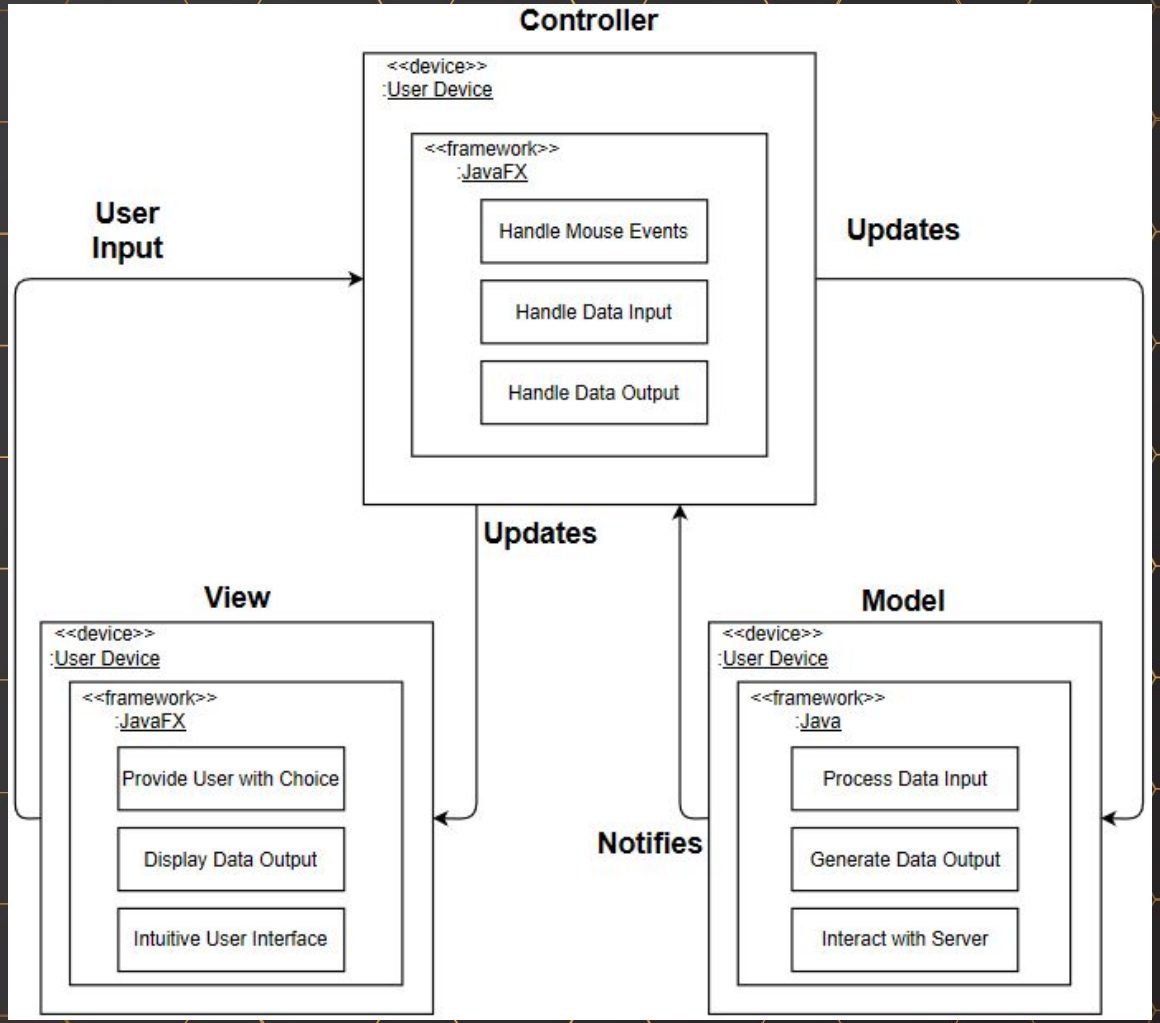
Management Plan

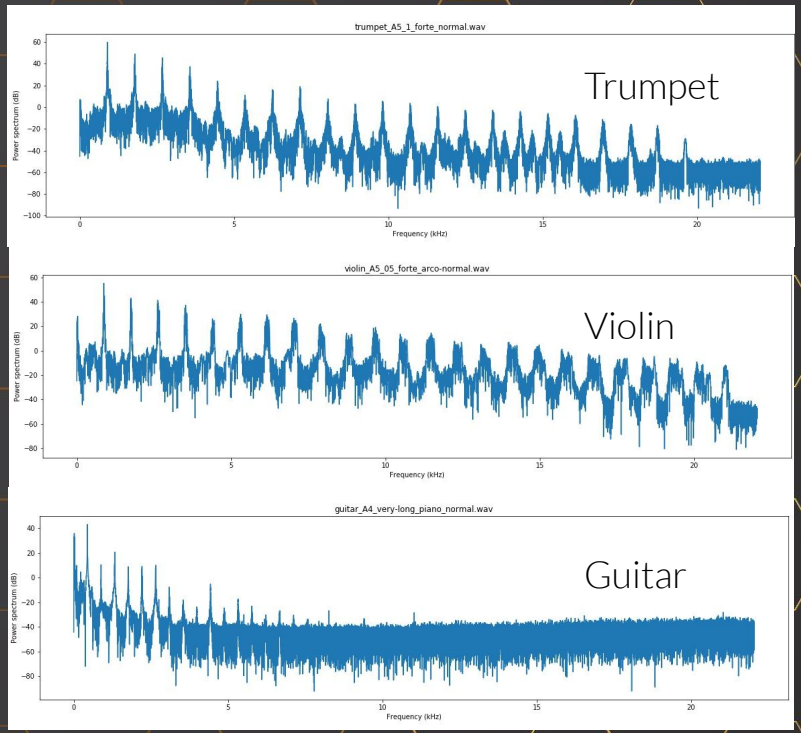
- We had updated our Sprint Board and our Burndown Chart

Architecture and Design

- We had slightly updated our architecture diagrams to reflect the optimizations we had done in the code.


Client
<<device>>
:User Device
<<execution environment>>
:Local
Model
View
Controller
Server
<<device>>
:User Device
<<execution environment>>
:Local
Machine Learning Model
Database Management
Database Management


Controller
<<device>>
:User Device
<<framework>>
:JavaFX
Handle Mouse Events
Handle Data Input
Handle Data Output
User Input
Updates
Updates
View
<<device>>
:User Device
<<framework>>
:JavaFX
Provide User with Choice
Display Data Output
Intuitive User Interface
Model
<<device>>
:User Device
<<framework>>
:Java
Process Data Input
Generate Data Output
Interact with Server
Notifies



Our input data

- We gathered all the Music samples from the LibROSA database.
- Converted the files from mp3 to WAV.
- From there we tried many different graphs to see what would fit CNN the best.
- Decided to use Power Spectral Density (PSD) graphs.

```python
import numpy as np
import matplotlib.pyplot as plt
import os, random, cv2, pickle

DATADIR = "C:/Users/Joe/Desktop/musical data"
CATEGORIES = ["Guitar", "Trumpet", "Violin"]

training_data = []

def create_training_data():
    for category in CATEGORIES:
        path = os.path.join(DATADIR, category)
        class_num = CATEGORIES.index(category)
        for img in os.listdir(path):
            try:
                img_array = cv2.imread(os.path.join(path, img), cv2.IMREAD_GRAYSCALE)
                new_array = cv2.resize(img_array, (360, 1440))
                training_data.append([new_array, class_num])
                #plt.imshow(img_array, cmap="gray")
                #plt.show()
            except Exception as e:
                pass

create_training_data()
random.shuffle(training_data)

#print(len(training_data))
#for sample in training_data[:10]:
#    print(sample[1])

X = []
y = []

for features, label in training_data:
    X.append(features)
    y.append(label)

X = np.array(X).reshape(-1, 360, 1440, 1)

pickle_out = open("X.pickle","wb")
pickle.dump(X, pickle_out)
pickle_out.close()

pickle_out = open("y.pickle","wb")
pickle.dump(y, pickle_out)
pickle_out.close()

pickle_in = open("X.pickle", "rb")
X = pickle.load(pickle_in)
```

Importing our data:

1. Access grouped files
2. Creating numeric labels
3. Resize graphs
4. Randomize training data
5. Separate X and y
6. Save using pickle

Layers

- MFCC -> PSD
- 43x128 -> 360x1440

Input size	Description
1 × 43 × 128	mel-spectrogram
32 × 45 × 130	3 × 3 convolution, 32 filters
32 × 47 × 132	3 × 3 convolution, 32 filters
32 × 15 × 44	3 × 3 max-pooling
32 × 15 × 44	dropout (0.25)
64 × 17 × 46	3 × 3 convolution, 64 filters
64 × 19 × 48	3 × 3 convolution, 64 filters
64 × 6 × 16	3 × 3 max-pooling
64 × 6 × 16	dropout (0.25)
128 × 8 × 18	3 × 3 convolution, 128 filters
128 × 10 × 20	3 × 3 convolution, 128 filters
128 × 3 × 6	3 × 3 max-pooling
128 × 3 × 6	dropout (0.25)
256 × 5 × 8	3 × 3 convolution, 256 filters
256 × 7 × 10	3 × 3 convolution, 256 filters
256 × 1 × 1	global max-pooling
1024	flattened and fully connected
1024	dropout (0.50)
11	sigmoid

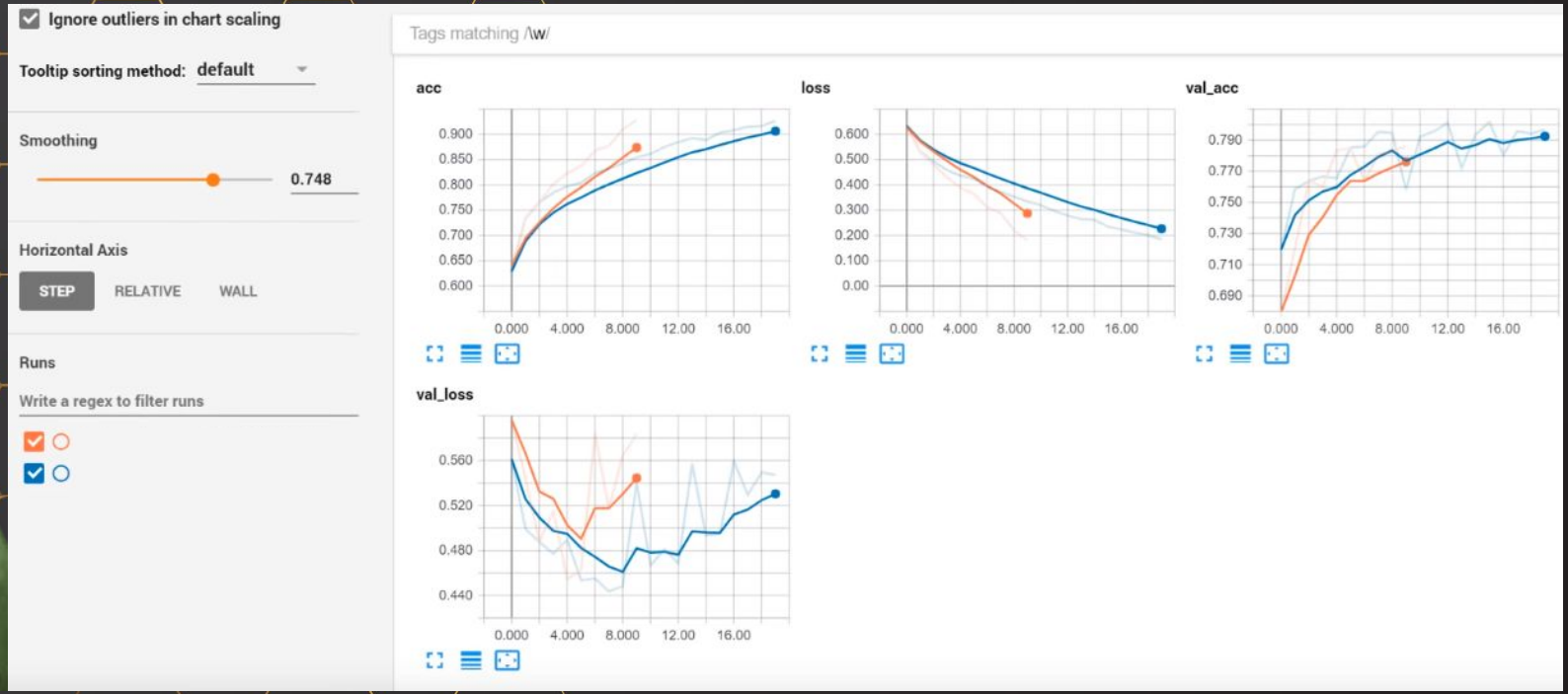
- How many layers does the model in the code have?
  We are currently modeling nine
- What are the names of the major layers in the model?
  Input layer, convolutional layers, and output layer
- How many layers is the team planning to incorporate for the next sprint?
  15-20 based on previous estimates from the Oxford model

```
import tensorflow as tf
from keras.models import Sequential
from keras.layers import Dense, Dropout, Activation, Flatten,\
    Conv2D, MaxPooling2D
import pickle

X = pickle.load(open("X.pickle","rb"))
y = pickle.load(open("y.pickle","rb"))

X = X/255.0

model = Sequential()
model.add(Conv2D(64, (3,3), input_shape = X.shape[1:]))
model.add(Activation("relu"))
model.add(MaxPooling2D(pool_size=(2,2)))

model.add(Conv2D(64, (3,3), input_shape = X.shape[1:]))
model.add(Activation("relu"))
model.add(MaxPooling2D(pool_size=(2,2)))

model.add(Flatten())
model.add(Dense(64))

model.add(Dense(1))
model.add(Activation('sigmoid'))

model.compile(loss="categorical", optimizer="adam", metrics=['accuracy'])

model.fit(X, y, batch_size=32, validation_split=0.1)
```

Challenges:

- Coordination
  - Late data
    - Extended Research
- Runtime
  - Computer go BOOM
    - 1500x360x1440 nodes
    - Reshape data
  - Integrity vs Efficiency
    - GPU offloading
- Batches
  - Desirable subset
    - 3 instruments with 500 graphs

Data Visualization using TensorBoard



Demo!

Login Hash

```
def hash_sha1(string):
    str = hashlib.sha1(string.encode())
    str_hex = str.hexdigest()
    hash_password = str_hex
    return hash_password
```

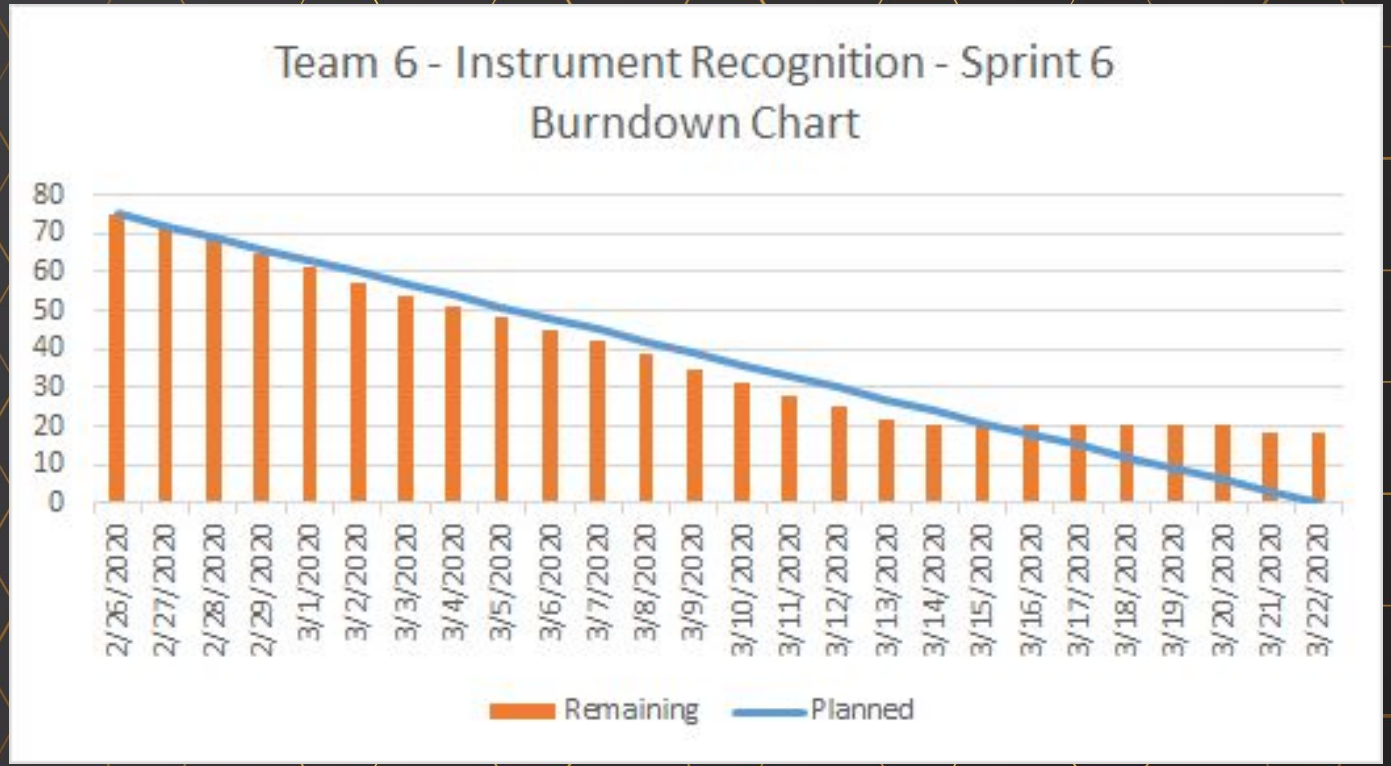
- Passwords tend to be easily attacked and hacked with brute force or a dictionary attack.
- What hashing will do is it will make it more difficult to guess what the password is which in theory will take more time away from the hacker as it'll take longer to guess.
- The Hash password is encrypted into random characters so humans would not be able to guess what the password is.
- Secure Hash Algorithm 1 is a cryptographic hash function which takes input and produces a 160 - bit (20 - byte) hash value known as a message digest - typically rendered as a hexadecimal number, 40 digits long.

Completion

- We are planning on completing, testing, and demonstrating our machine learning model with actual features from an actual piece of music by April 8th.

Sprint 6

- Sprint Goals:
  - Create a logo and update the desktop application with the new logo
  - WAV to Fast-Fourier Transformation conversion
  - Fast-Fourier Transformation to Power Spectral Density graph conversion
  - Determine the layers and filters to be used by a preliminary model
- User Story Points:
  - Planned: 75
  - Achieved: 52


Team 6 - Instrument Recognition - Sprint 6 Burndown Chart
80
70
60
50
40
30
20
10
0
2/26/2020
2/27/2020
2/28/2020
2/29/2020
3/1/2020
3/2/2020
3/3/2020
3/4/2020
3/5/2020
3/6/2020
3/7/2020
3/8/2020
3/9/2020
3/10/2020
3/11/2020
3/12/2020
3/13/2020
3/14/2020
3/15/2020
3/16/2020
3/17/2020
3/18/2020
3/19/2020
3/20/2020
3/21/2020
3/22/2020
Remaining
Planned

Sprint Board - Instrument Recognition Software

CECS 491A Free | Public | AD CP JF +2 | Invite

Sprint Goals

- Sprint #6
- LOGO
- WAV -> FFT -> PSD conversion
- LAYERS and FILTERS

+ Add another card

Project Backlog

+ Add a card

Sprint Backlog

+ Add a card

In Progress

- Desktop Application: Work on our user authentication through MongoDB
- Desktop Application: Work on streamlining our user interface
- Machine Learning: Work on implementing the desired model
- Machine Learning: Figure out how to upload custom data sets
- Life: Survive the Coronavirus Pandemic
- Life: Trade bottlecaps for toilet papers

+ Add another card

Done

- General: Create a logo. A placeholder is fine. 2
- Desktop Application: Update the desktop application with the new logo.
- Machine Learning: WAV to FFT conversion
- Machine Learning: FFT to PSD conversion
- Machine Learning: Determine the layers and filters to be used by a preliminary model

+ Add another card

Sprint Retrospective

- Pandemics aren’t fun (staying safe from COVID-19)
- Need to plan out better and manage time better
- Computer specs aren’t up to the task of simulating the CNN

Thank you for listening!