NORTHERN ARIZONA UNIVERSITY

TEAM INTELLICHIRP

Technological Feasibility Analysis

Authors
Steven ENRIQUEZ
Michael EWERS
Joshua KRUSE
Zhenyu LEI

Clients
Colin QUIN
Patrick BURNS
Mentor
Fabio SANTOS

October 23, 2019

CONTENTS

Technological Feasibility Analysis

Steven Enriquez Michael Ewers, Joshua Kruse and Zhenyu Lei

1 Introduction

It is becoming ever more important to track and manage the biodiversity that live on Earth. More and more animals are becoming extinct everyday. Soundscapes2Landscapes is a science-based project looking to further advance biodiversity monitoring in order to save the lives of plant and animal species. At the moment, Soundscapes2Landscapes place low cost audio recording devices in various sites in Sonoma, California. Once placed, these devices record one minute of every ten minutes for three to five days at each site. This results in terabytes of saved audio data per site. Using this data, a manual classification process is conducted by researchers and citizen scientists to identify different species and other audio components. The problem is that this manual identification process is an incredibly time consuming task with this much data. Not only is the process too time-consuming, but is also not user-friendly to those who are not tech-savvy as the current implementation uses a command line interface.

Our clients Colin Quinn and Patrick Burns are part of the Global Earth Observation Dynamics of Ecosystems Lab (GEODE). They work with Soundscapes2Landscapes to help achieve their biodiversity monitoring goals. They have assigned this task of automatic sound identification to our team.

1.1 Solution

Our team has developed a solution around a user-friendly user interface that hosts a machine learning model. We will be using a machine learning algorithm to automatically classify different types of sounds in these recordings. The three broad categories of sound that we will be classifying are biophony, geophony, and anthrophony. Biophony is a sound category for life, including animals noises. Anthrophony is a sound category for humans, including people speaking, and noises from cars. Geophony is a sound category for earth-related sounds, including rain, wind, thunder, and other earth-related sounds. The results of this classification will be visualized in a variety of ways on an easy to use, user-friendly web application. The solution will also be created as a standalone application for offline use in the field.

1.2 Purpose

This document is a technological feasibility, discussing all of the technological challenges that we will face with research found by the team for effective solutions. Each challenge will have a solution picked out as the best choice to handle their respective challenge, with rationale and a way to further test our decision.

1.3 Document Outline

We begin in Section 2 by analysing the major technological challenges we expect to encounter during our project. In Section 3, we analyze each of these areas carefully, including looking at alternatives, how we explored these alternatives, and rationale for choosing a particular solution. In Section 4, we look at how the technology will integrate with each other with a diagram as well as an explanation for each individual component.

2 Technological Challenges

Below are the technological challenges that our team IntelliChirp has found to be crucial for providing a successful product to the clients, each with a brief description.

2.1 Machine Learning Algorithm

We will need a Machine Learning Algorithm to accurately classify the anthrophony, geophony, and biophony in each sound file. This model must be efficient, accurate, and run fast with the amount of data that we expect to access.

2.2 Building and Training the Machine Learning Model

We will need to build and train the Machine Learning Model on labeled audio data. With the amount of data that we have and expect to access, this model must be able to handle that amount of data.

2.3 Standalone App GUI

We will need a standalone application with a Graphical User Interface. This will be used to allow for the stretch goal of bringing the solution into the field. Allowing this machine learning model to be integrated with a laptop or Raspberry Pi.

2.4 Front-End Web Application Framework

We will need a front-end web application framework to allow users to use the machine learning model and visualize the analyzed results. This framework will be used to build a scalable and user-friendly interface in a timely manner.

2.5 Back-End Web Application Framework

We will need a back-end web application framework to host the machine learning model and communicate with the front-end. This will be the connection between out model, the site, and a database.

2.6 Database

We will need a database to save user inputted audio files for future training of the machine learning model. Users will have the option to provide feedback on analyzed audio, letting them verify if the classification of their audio was correct. In return, they will be asked if they would like to help the organization by allowing the correctly labeled portion of audio to be saved in a database for further training and greater accuracy in the machine learning model.

2.7 Visualization of Results

We will need a way to visualize the results of user inputted data fed into our machine learning model. Examples of visualizations will include a pie chart of the different proportions of each category (biophony, geophony, and anthrophony), as well as a labeled spectrogram of the analyzed audio.

With the technological challenges overviewed, each challenge will be analyzed in detail to determine the best solution in the Technological Analysis section below.

3 TECHNOLOGICAL ANALYSIS

In this section of the Technological Feasibility, we will explain the options for solving each technological challenge as well as provide justified reasoning on which is the best choice for our product. Careful consideration and research is required to ensure a well-thought-out decision.

3.1 Machine Learning Algorithm

We will need a way to accurately classify the anthrophony, geophony, and biophony of an inputted soundscape file. To accurately assess the data a machine learning algorithm trained on specific categories using large amounts of training data is needed.

3.1.1 Alternatives

Deep Learning Algorithms are a popular choice, by modeling their internal structure off of how the human brain stores memories. Clustering algorithms are also used when grouping different types of data is needed, this uses a more statistical and math based approach. Many different machine learning algorithms are used for classifying data and each has its pros and cons.

3.1.1.1 : Deep Learning Algorithms
A type of machine learning technique useful for large data sets. Generally neural networks process given data as a stored vector, where the vector is passed through many different layers of functions that produces the determination of which category the inputted data falls under. These functions and layers are trained on large amounts of data points where each data point will slightly alter the functions used to determine the category. Depending on the deep learning algorithm used the functions and layers will differ in structure. This provides different benefits based on the type of data to be categorized.

Pros
- Supervised learning is better for large amounts of categories.
- Deep learning is more accurate when large amounts of categories are needed to be defined.

Cons
- Need large amounts of training data for the model to be accurate.
- Training the model takes a long time and is process intensive.

3.1.1.2 : Clustering Algorithms
A type of machine learning technique that involves grouping or clustering different data points together. Clustering is an unsupervised learning method, in which the categories are not determined beforehand. The algorithm determines the categories after running through all the data. The data points are grouped by the similarity of the data. For instance, if there are ten sound files that contain a truck starting, and ten sound files that contain crickets chirping; these should be evenly grouped without overlap by the algorithm. Then in the best case scenario, when a sound file with unknown data is inserted into the algorithm, it would correctly identify whether this file contains a truck starting or crickets chirping.

Pros
- Better when large amounts of testing data are not available.
- Faster at analyzing data than deep learning algorithms.
- Faster at training data than deep learning algorithms.
- Algorithm is more accurate when the number of training samples increases.

Cons
- Unsupervised learning are worse for large amounts of categories.

- Complexity is worse when the number of training samples increases.

3.1.1.3 : Support Vector Machine
A type of machine learning algorithm that is similar to unsupervised learning clustering algorithms but instead use supervised learning. Clustering algorithms try and plot similar data in groups and then determine if new data falls into one of these groups. Support Vector Machines are given the categories upfront and then the algorithms determines the lines that separate these categories. When new data is entered into the model the algorithm determines if this data falls into a region that a category defines.

Pros

- Better when large amounts of testing data is not available.
- Faster at analyzing data than deep learning algorithms.
- Faster at training data than deep learning algorithms.
- Supervised learning is better for large amounts of categories.

Cons

- Complexity is worse when the number of training samples increases.

3.1.2 Chosen Approach

From the information presented in [5, Tab.1], a Support Vector Machine should be our first choice for an algorithm that will categorize sound files. Secondly, a Clustering algorithm should be looked at, then finally a Deep Learning algorithm. A Support Vector Machine is our first choice given the large amount of categories before training, trains on smaller datasets better than a Deep Learning Algorithm, and will take less time to train the model than a Deep Learning algorithm.

TABLE 1
Machine Learning Algorithm Comparison Table

	Better With Small Amounts of Data	Large Number of Categories	Speed of Training	Speed of Deployment	Total
Deep Learning	1	5	1	2	9
Clustering	5	1	3	3	12
SVM	3	3	3	4	13

3.1.3 Proving Feasibility

In order to prove that a Support Vector Machine is the best model to go with numerous testing processes will have to take place. Cross validation is a method of error checking that involves many different iterations of training and testing data. Training is done on a percentage of data from the full data set, and tested using another percentage of the data set. The results of this testing is then cross validated with another set, that was trained and tested on a different percentage of the full data set. This process is done numerous times in order to determine the overall accuracy of the algorithm. To test our deep learning algorithm a confusion matrix is used to test the accuracy of classification. This matrix gives data on which categories the algorithm struggles with determining. In addition it gives data on which categories are very similar and where the algorithm finds difficulty in determining if an input is one or another. In addition, the overall accuracy of classification and the processing speed of identification will be a factor in determining which algorithm is the strongest to use for our purposes.

3.2 Training the Machine Learning Model

The machine learning algorithm that is chosen will need to be built and trained in order to give accurate results. Machine learning frameworks allow for the use of prebuilt libraries to assist in the process of creating a high quality architecture without recreating the wheel. Training is also a critical and time consuming component to the success of this product.

3.2.1 Alternatives

The solutions that are possible include TensorFlow with Keras, PyTorch, Apache Spark, and MXNet. Choosing the correct framework is difficult, and depends on many factors. These factors include which type of machine learning algorithm will be used, the programming language that will be used, the purpose of the project, and the ability to use it for creating a model that is usable on small devices such as a Raspberry Pi.

3.2.1.1 : TensorFlow with Keras
A very powerful and widely used machine learning framework for deep learning. TensorFlow is open source and uses data flow graphs to build models. Developers using this framework are able to create large-scale neural networks.

Pros

- Large Community with easy access to online resources.
- Developed by Google.
- Great visualizations with TensorFlow's visualization toolkit, TensorBoard.
- Scalable to larger-scale models.
- Options for developing for smaller devices using TensorFlow Lite.
- Ideal for cross-platform products.
- Keras can be used as a powerful user-friendly interface, simplifying the approach to using TensorFlow.

Cons

- Uses a Static Computation Graph, so you must retrain the model after any change to the architecture.
- Higher learning curve than other frameworks.
- Debugging can be difficult.

3.2.1.2 : PyTorch
A flexible and high performing machine learning framework. This framework is used by Facebook, IBM, and many other organizations.

Pros

- Easier learning curve than other frameworks.
- Uses a Dynamically Updated Graph, allowing for changes to the architecture without having to retrain the model.
- Supports data parallelism for performance.
- Good for building quick prototypes.
- Powerful for many types of machine learning algorithms.

Cons

- Limited documentation is available.
- New framework that was initially released in 2016.

3.2.1.3 : Apache Spark
A popular machine learning framework. This framework was written mostly in Java, R, Python,

and Scala. Any developers who are comfortable with Python and R would feel comfortable working with Apache Spark's easy-to-use APIs.

Pros
- Easy-to-use APIs that are familiar to developers that have used Python and R.
- Scalability to allow for the same code to run on bigger machines.

Cons
- Is not ideal for a multi-user environment.
- Fewer algorithms than other machine learning frameworks.
- There is no automatic optimization process.

3.2.1.4 : MXNet

A deep learning framework that supports many programming languages. This framework is new and is used to train and deploy deep learning models.

Pros
- Can be run on any device.
- Very scalable and is used by large companies for speech recognition, handwriting recognition, and more.
- Offers support for many programming languages, including Python, Scala, Julia, Clojure, Java, C++, R and Perl.

Cons
- Small community
- Not popular in the research community
- Relatively new framework

3.2.2 Chosen Approach

The chosen approach was TensorFlow, as presented in [7, Tab. 2]. This framework provides a wealth of documentation online, a large community, as well as options for moving the model to different devices. Despite a larger learning curve, Keras can be added to provide a simplified approach as well as a plethora of documentation can be used to allow this powerful framework to be used for our product.

TABLE 2
Machine Learning Framework Comparison Table

	Learning Curve	Documentation and Community Size	Ideal for many Machine Learning Algorithms	Portable for Small devices	Total
TensorFlow with Keras	4	5	5	5	19
PyTorch	4	3	5	4	16
Apache Spark	4	3	3	4	14
MXNet	3	2	4	5	14

3.2.3 Proving Feasibility

In order to prove that TensorFlow is going to be the best option for our software system, we will create and conduct tests and visualization with TensorFlow and PyTorch. We will also need to test the effect in time consumption for utilizing TensorFlow's static computation graphs instead of PyTorch's dynamic computation graph. Those frameworks were the top two choices from our team's research.

3.3 Standalone App GUI

We will need to create a standalone application which must be able to host a machine learning algorithm. The application must run the machine learning model with any uploaded files locally, allowing for offline use of the application. The application must also be able to produce visualitions of the results. In order to build the application, we will need to look into different GUI frameworks. There are plenty of GUI frameworks available, but we will need to choose those which support cross-platform functionality.

3.3.1 Alternatives

The available alternatives for this technological challenge include PythonQt, ElectronJs, JavaFx and JavaSwing. The available options are all viable for the creation of a GUI, but we will need to determine the best solution based on criteria that we deem to be important. Our criteria consist of the learning curve, how the visual design of the GUI is, how well we can implement our machine learning model within the application, and how resource intensive the GUI will be while running on a device.

3.3.1.1 : PythonQT

A python based GUI library. The PyQt library contains a visual application builder IDE, which allows for much quicker and simpler designs of an application project. The library being run in Python allows for our machine learning model to be accessed directly via the source of the application.

Pros

- Simple design and building of GUI applications in the provided PyQt Designer.
- Runs in Python, which corresponds to our machine learning algorithms and scripts required to utilize the algorithm.
- Available deployment across popular platforms without major source code changes, including Windows, macOS and Linux.

Cons

- Not a default library included with the Python installation.
- Larger framework with learning curve due to all the available functions and methods.

3.3.1.2 : ElectronJS

Creating an application GUI with ElectronJS with Python is a different approach from using a visual design builder. ElectronJS allows developers to create GUIs in the form of HTML and CSS files, and may include Python in order to run any scripts. Allowing HTML and CSS files to run the GUI will create opportunities for more intricate designs of the application GUI. Further, we would be able to effortlessly import our web application files to mimic the web application as a standalone application. Lastly, allowing Python to operate within the application is a great advantage for our machine learning model to be run in scripts.

Pros

- Web application's files can be used to create the GUI of the standalone application.
- Includes python integration, allowing for machine learning models to be utilized.
- Available deployment across popular platforms without major source code changes, including Windows, macOS and Linux.

Cons

- No standalone visual design builder.
- RAM usage on host machine can be very high.

- May start slow and run slower than other available GUI frameworks.

3.3.1.3 : JavaSwing

The JavaSwing framework is a simple Java GUI building framework, which allows the developer to visually build an application. The commonly used IDE which supports visually building an application project is Apache NetBeans. The framework is run in Java, which may pose some difficulties when hosting and running any machine learning algorithms, as the machine learning models are normally written in the Python programming language.

Pros

- Simple designing and building of GUI applications with tools in the IDE Apache NetBeans.
- Available deployment across popular platforms without major source code changes, including Windows, macOS and Linux.

Cons

- There is a learning curve for building visual designs using “Layout” tools.
- Ran in Java, which requires extra libraries such as Java Runtime or Jython to access required machine learning scripts.

3.3.1.4 : JavaFX

The framework is a simple Java GUI builder, and allows for visual development similar to JavaSwing. The framework is run with a standalone JavaFX application, which is used to build a visual representation of the application project. The JavaFX application may be connected to the VisualStudio IDE for more fluid work from visual to backend of the application. Similar to JavaSwing, the framework works in the Java language, which will pose difficulties when accessing the machine learning algorithms and running them.

Pros

- Simple designing and building of GUI applications with the provided JavaFx standalone designer.
- The connection from GUI to back-end code is highly understandable.
- Available deployment across popular platforms without major source code changes, including Windows, macOS and Linux.

Cons

- Ran in Java, which requires extra libraries such as Java Runtime or Jython to access required machine learning scripts.

3.3.2 Chosen Approach

The chosen approach will be ElectronJS With Python integrated from our Standalone GUI development as presented in [10, Tab. 3]. The ElectronJS framework provides a great set of tools for creation of a standalone GUI from Javascript, HTML, and CSS files. The framework would allow us to port over our web application design to ensure that we have consistency within all our deliverable applications. Along with portability, ElectronJS also allows for python implementation. Python integration will allow for our team to include any machine learning models, machine learning scripts, and visualization scripts.

TABLE 3
GUI Framework Comparison Table

	Learning Curve	Visual Design Process	ML Model Integration	Low Resource Usage	Total
PyQT	5	5	5	5	20
ElectronJS with Python	5	5	5	3	18
JavaSwing	4	4	3	5	16
JavaFX	5	5	3	5	18

3.3.3 Proving Feasibility

In order to prove that ElectronJS is going to be the best option for our software system, we will create a mini application which will need to be based off of a mockup web application. The mockup web application will have back-end functionality to run python scripts, and we will create a simple script to run for testing purposes. The mini application will need to show that we can fluidly port the web application UI over as well as show that the application can run the test script used in the web version. In the testing process, we will monitor the resource usage on the OS while running beside a PyQt GUI application running the same test script.

3.4 Front-End Web Application Framework

We will need to create a web application that is easy to use for any user. The web application will need to have an interactive UI which will allow users to upload audio files and be able to visualize the results of their analyzed audio. A front-end framework will assist in creating a user-friendly web application in a timely manner.

3.4.1 Alternatives

The available alternatives include ReactJS, AngularJS, and Bootstrap. These are all very viable options for the creation of a web application's front-end. We have created specific criteria which we have found to be important for the development of our front end. The criteria we have defined are the process and difficulty of building a web application with the respective framework, how fluidly the visual designing process works, how well the front end can connect to our backend framework, and how low the learning curve of the framework is. We will discuss the research of each alternative below.

3.4.1.1 : ReactJS
A commonly used library for front-end development by providing a Model-View-Controller style of building. ReactJS supports reusability of UI components, which are easily expressed as functions when programming. ReactJS also allows for a generally more simple process for developing interactive UIs.

Pros

- Creation of web apps becomes easier than using pure HTML files.
- Offers reusable UI components.
- Performance optimizations from virtual Document Object Models (DOM).

Cons

- Model-View-Controller models have a learning curve.
- ReactJS's syntax will have a slight learning curve

3.4.1.2 : AngularJS

An effective JavaScript framework that provides developers with a structured Model-View-Controller style of website creation. AngularJS's data binding allows for a unique synchronization of data between any view and model components.

Pros

- Data binding allows developers to not worry about their view and model components having issues.
- Document Object Models (DOM) do not require manipulation from developers.

Cons

- Model-View-Controller models have a learning curve.
- Large amount of available directives.

3.4.1.3 : Bootstrap

An open source CSS framework which is tailored towards informative development of websites. Bootstrap is more commonly used for the creation of web pages, and not web applications.

Pros

- Available use of JavaScript plugins with jQuery.
- A wealth of documentation is available
- Great grid system for structuring web pages.

Cons

- More directed towards informative web page development, instead of web applications.
- Websites can be plain, and takes extra work to create a more customized product.

3.4.2 Chosen Approach

The chosen approach will be ReactJS for our front-end web framework as presented in [11, Tab. 4]. ReactJS is ideal due to its usability within JavaScript, making the general learning curve low. ReactJS's JSX syntax allows for a combination of Javascript and HTML, making for a very fluid development process.

TABLE 4
Front-End Web Application Framework Comparison Table

	Web App Development Process	Visual Design Process	Connection to Backend	Low Learning Curve	Total
ReactJS	5	5	5	5	20
AngularJS with Python	5	4	5	4	18
Bootstrap	4	3	4	5	15

3.4.3 Proving Feasibility

To prove the feasibility of ReactJS, we will implement a demonstration of an interactive UI which will require a file upload, and be able to run a test script. The ability to run a test script will ensure that we will be able to connect our front-end to our machine learning scripts and model. The front-end demo will also be useful to test the creation of our standalone application.

3.5 Back-End Web Application Framework

Our product will need a back-end web framework that connects a machine learning algorithm and database to our web application. A framework is needed that will allow for scalability and reliability. The web application will not be monolithic, so a lightweight framework would be best suited for the product.

3.5.1 Alternatives

The researched solutions include Flask, Django, and Spring. Choosing the correct framework depends on some factors. These factors include the documentation size, the learning curve, whether or not the framework is suited for a small web app, and the speed of the framework. We will discuss the research of each alternative below.

3.5.1.1 : Flask

One of the most popular Python web application frameworks. This web application framework is lightweight, and provides the ability to get up and running easily. There are many Flask extensions and libraries to provide added functionality to a web application.

Pros

- Minimalistic without the lack of power.
- Flask has as low learning curve, so it is easy to learn and use.
- Routing URLs is simple implement.
- Very flexible and easily extensible with extensions and libraries.

Cons

- Relatively new, being released in 2010.
- Limited documentation and community.
- Limited features, compared to other web frameworks
- The flexibility can lead to a large difference between conventions, making it possible to be difficult to use by new developers

3.5.1.2 : Django

A popular Python full stack web framework that allows for quick and complete development. This framework is free and open source. Due to the monolithic amount of features, this framework could be seen as overkill for a small web application. This framework can provide everything a web application needs, including a front-end framework.

Pros

- Allows for fast development from the many plugins and customization.
- Secure
- Django is very scalable, supporting large-scale applications such as Instagram and Pinterest.
- There is a wealth of documentation.
- APIs are available through the Django Rest Framework
- Django is widely used and has a large community
- Offers benchmarks for checking the speed and bottlenecks of a web application.

Cons

- Large learning curve
- Components get deployed together
- Not ideal for static one pagers or microservices

3.5.1.3 : Spring

A popular framework for enterprise Java. This framework is under the Apache 2.0 license and was first released in 2003. Enterprise-class applications are possible with Spring and it is very modular. The most popular aspect of Spring is Dependency Injection, which is useful for complex applications with classes that developers want to be independent.

Pros

- Modular with the many packages and classes Spring provides.
- Provides support for Dependency Injection.
- Very popular with enterprise applications.

Cons

- Bigger learning curve than other frameworks.
- Not suited for smaller web applications with the large size of the framework.

3.5.2 Chosen Approach

The chosen approach was Flask, as presented in [13, Tab. 5]. This back-end framework is lightweight, flexible and easily extensible. Choosing Flask will be the best decision for a small web application that only requires hosting a machine learning algorithm as well as connecting to a database to save audio files for further training.

TABLE 5
Back-End Web Application Framework Comparison Table

	Documentation and Community	Learning Curve	Lightweight for a Small Web Application	Speed	Total
Flask	3	5	5	5	18
Django	5	3	4	4	16
Spring	4	3	3	5	15

3.5.3 Proving Feasibility

In order to prove that Flask is going to be the best option for our software system, we will create a demo website, host a machine learning model found on the internet, and create a demo connection to a database using the top two alternatives for this technological challenge. The top two alternatives are Flask and Django. This will test the performance of Flask being used as a back-end web framework.

3.6 Database

When users upload their audio files for analysis, the machine learning algorithm will classify the audio components that were found. We will want to store correctly classified audio in a database for future training of the machine learning model. Therefore, the user will be asked to verify if the machine learning model correctly classified their audio, and if they would like to submit the audio for this further training.

3.6.1 Alternatives

The solutions that are possible include MongoDB, DynamoDB, MariaDB, and MySQL. Choosing the correct database is difficult, and depends on many factors. These factors include the learning curve, scalability and performance. We will discuss the research of each alternative below.

3.6.1.1 : MongoDB

A document-oriented database. It is distributed and easily scalable geographically and horizontally for better performance.

Pros

- MongoDB is a schema-less database which gives us flexibility and freedom to store data of different types.

- Easy to access documents by indexing.
- When you have to handle a large data, you can distribute MongoDB to several machines.
- MongoDB has advanced features for ad hoc queries.

Cons

- MongoDB doesn't support JOINs like a relational database.
- There is data redundancy in MongoDB. This results in increasing unnecessary usage of memory.
- MongoDB limits the document size to a maximum of 16MB.
- You cannot perform nesting of documents for more than 100 levels.

3.6.1.2 : DynamoDB (Amazon Web Services)

Amazon DynamoDB is a specialized NoSQL database hosting service that fully supports key values and document data structures. DynamoDB is provided by Amazon as part of the Amazon Web Services portfolio.

Pros

- DynamoDB is easy to integrate with AWS Lambdas and API Gateway.
- It is secure with all the different ways AWS can secure resources.
- Boto's SDK is pleasant and easy to use.
- No infrastructure needs to be managed.

Cons

- Not possible to use DynamoDB without an AWS account.
- DynamoDB does not let you dump large JSON blobs in a key value store above a certain size.
- DynamoDB does not back up your tables for free.
- There is a higher learning curve compared to other database options.

3.6.1.3 : MariaDB

MariaDB is a branch of the MySQL Relational Database Management System. This database alternative is commercially supported. It aims to maintain a high degree of compatibility with MySQL.

Pros

- MariaDB supports parallel replication and multi-source replication
- MariaDB is free and open-source.
- MariaDB is fast and stable.
- Large-scale web applications have transitioned to MariaDB such as Wikipedia.
- Extensible structure and plugins allow for customization.
- MariaDB has progress bars for query progress.
- MariaDB can encrypt at the network, server and application levels.

Cons

- MariaDB is fairly new.
- Free database but online support is not free.

3.6.1.4 : MySQL

MySQL is one of the most popular databases for web-based applications. It's fully multithreaded by using kernel threads, and can handle multiple CPUs if they are available, which allows a higher performance than other models. MySQL provides transactional and non-transactional storage engines, and has a high-speed thread-based memory allocation system. It also works on many different platforms.

Pros

- MySQL is free to use.
- There are a variety of UIs that can be implemented.
- It can be made to work with other databases.
- Provides stability for large amounts of data.
- Widely used with a large community.

Cons

- MySQL is hard to automate.
- There is no built-in support for XML or OLAP.
- MySQL is not completely open source.
- MySQL is Oracle-owned instead of community driven.

3.6.2 Chosen Approach

The chosen approach was MariaDB, as presented in [15, Tab. 6]. This database is flexible, easy to learn, easy to use, and provides great performance. Choosing this high performance database will be the best decision for our web application.

TABLE 6
Database Comparison Table

	Learning Curve	Scalability	Performance	Total
MongoDB	4	5	3	12
DynamoDB	2	5	3	10
MariaDB	4	5	5	14
MySQL	4	4	4	12

3.6.3 Proving Feasibility

In order to prove that MariaDB is going to be the best option for our database, we will create a Maria database and connect it to a backend framework to simulate a series of operations such as uploading and deleting audio files. This will test the performance of MariaDB as a database.

3.7 Visualization of Results

Data can be very difficult to understand when not displayed in an intuitive manner. The critical aspect of data visualization is to present visuals in a way that provides the user with the most insights into the information as possible while maintaining simplicity. We will provide users of our product with a pie chart containing the proportions of different audio components, including biophony, anthrophony, and geophony, as well as a labelled spectrogram with classified components displayed.

3.7.1 Alternatives

After careful study, we believe that the visualizations of analyzed audio will not be as large of a challenge than we expected. We analyzed many visualization tools on the market and found that mainstream solutions include Plotly, Matplotlib, and others. Both Plotly and Matplotlib offer visualizations of pie charts and spectrograms, and we can easily switch between different options for visualization if that is needed.

3.7.2 Chosen Approach

The chosen approach was Plotly as presented in [16, Tab. 7]. Choosing this web-friendly visualization tool will be the best decision for our web application. This tool is capable of displaying the needed visualizations that our product will need and more.

TABLE 7
Visualization of Results Comparison Table

	Visualizes Pie Charts and Spectrograms	Functionality	Low Learning Curve	Total
Plotly	5	5	45	15
Matplotlib	5	5	5	15

3.7.3 Proving Feasibility

After conducting research on different visualization tools, we have found a solution and feel that proving the feasibility will not be needed. Many options are equally viable and come down to choosing any of them. In case the visualization tool needs to be changed, the difficulty of switching a visualization library is very low.

4 Technology Integration

The building of our envisioned system consists of the following components. Starting with the higher level set of challenges, we will need to solve the implementation and testing of training our machine learning model, choosing the correct machine learning algorithm to suit our problem, and the creation of a database. The next level consists of the main pieces of our website, being the front-end and back-end development. Lastly, the previous levels connect to the creation of the Standalone application and Web application which will produce our visualizations and the results from the machine learning model's predictions. The visualization of our envisioned system is illustrated in [17, Fig. 1].

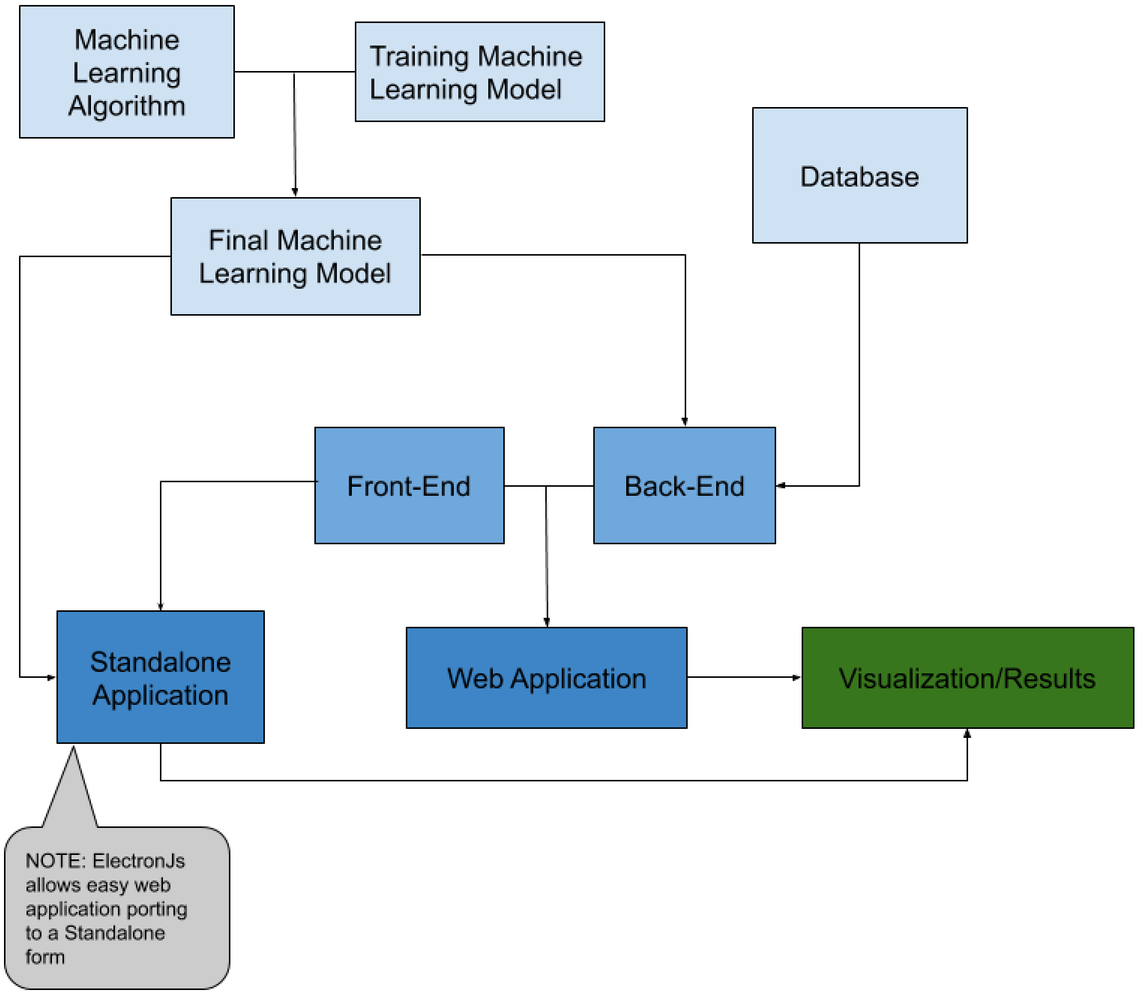

Fig. 1. Visualization of Envisioned System

Our web application will come together by connecting our MariaDB and machine learning model to our application with the back-end. The back-end of our web application will be developed with the Flask framework. Flask is a Python framework, which will be useful when calling our python scripts. Our front-end UI framework will be developed with ReactJS, and will be responsible for the intractability of our web application. ReactJS will connect to Flask to perform necessary interactions such as button clicks, file uploads, etc, which will then fire any functions required to run our scripts. The standalone application will use an ElectronJS GUI to incorporate all of the functionality that the web application will provide. Provided in this section is how the technology will integrate with each other to produce our product and provide all of the needed functionality.

5 Conclusion

The problem that we are trying to solve is the time-consuming manual identification process of audio components in recordings from various sites in Sonoma, California. Our solution is to develop a user-friendly web application that hosts a machine learning model to automatically classify these audio components. As a stretch goal, we will also be implementing a standalone application for use on a laptop or Raspberry Pi. In order to get one step closer in making this solution a reality, we have put together this technological feasibility document. This document's goal is to make sure that creating this project is technologically feasible. A summary of each technical challenge we are facing, and what technology we have chosen to solve each respective challenge is presented in [18, Tab. 8].

Proving the feasibility will continue in further testing. This analysis process allowed us to look intently at each technological challenge and research what some of the best solutions are. Our solutions to the challenges below will allow us to complete our project effectively and efficiently. Our team is confident that we will be able to produce a product that saves researchers and citizen scientists time with automated classification and an accessible user-friendly interface.

TABLE 8
Chosen Technology Summary

Challenge	Chosen Technology
Machine Learning Algorithm	Support Vector Machine
Training the ML Model	TensorFlow
Standalone App GUI	PyQT
Front-End Web App Framework	ReactJS
Back-End Web App Framework	Flask
Database	MariaDB
Visualization of Results	Plotly

INTELLICHIRP