Capstone: Record Identifier

Codi Smith

Student ID: 001098797

|  |  |  |
| --- | --- | --- |
|  | Table of Contents |  |
| **Prompt A** |  | **4** |
| Letter of Transmittal |  | 4 |
| Project Recommendation |  | 6 |
| Problem Summary |  | 6 |
| Application Benefits |  | 6 |
| Application Description |  | 7 |
| Data Description |  | 7 |
| Objective and Hypothesis |  | 8 |
| Methodology |  | 8 |
| Funding Requirements |  | 9 |
| Stakeholders Impact |  | 9 |
| Data Precautions |  | 10 |
| Developer Expertise |  | 10 |
| **Prompt B** |  | **11** |
| Project Proposal |  | 11 |
| PROBLEM STATEMENT |  | 11 |
| CUSTOMER SUMMARY |  | 11 |
| EXISTING SYSTEM ANALYSIS |  | 12 |
| DATA |  | 12 |
| PROJECT METHODOLOGY |  | 13 |
| PROJECT OUTCOMES |  | 14 |
| IMPLEMENTATION PLAN |  | 14 |
| EVALUATION PLAN |  | 16 |
| RESOURCES AND COSTS |  | 17 |
| TIMELINE AND MILESTONES |  | 17 |
| **Prompt C** |  | **19** |

[Application Files 20](#_Toc82008925)

[Prompt D 21](#_Toc82008926)

[Post-implementation Report 21](#_Toc82008927)

[Project purpose 21](#_Toc82008928)

[Datasets 22](#_Toc82008929)

[Data product code 23](#_Toc82008930)

[Hypothesis verification 24](#_Toc82008931)

[Effective visualizations and reporting 24](#_Toc82008932)

[Accuracy analysis 25](#_Toc82008933)

[Application testing 25](#_Toc82008934)

[Appendices 26](#_Toc82008935)

[Installation Guide 26](#_Toc82008936)

[User Guide 28](#_Toc82008937)

[Summation of Learning Experience 32](#_Toc82008938)

[References 33](#_Toc82008939)

**Prompt A**

**Letter of Transmittal**

June 30th, 2021

Kevin Lewandowski

4145 SW Watson Ave

Beaverton, OR 97005

Mr. Lewandowski,

Keeping a collection up to date is considered one of the most important aspects of being a Discogs user.

Traversing artists and release variants in the Discogs database can lead to human errors, which in turn leads to an inaccurate user collection. To reduce the impact of the human factor of an up-to-date user collection, I propose the creation of a tool to all users to find their vinyl record release entry in the database using two images; one of the album’s cover artwork and the second image being of the physical vinyl record. This will provide the use with a few music releases that are likely the record they are looking for, which they can then add to their collection. Due to the sensitive nature of the data processed by this record identifier, users will have to authorize the application’s access to their account each time they use it.

Providing the ability to search by images will give Discogs users an alternate method of managing their collections and improve the accuracy of a collection and its reliability. Data gathered from the searches can provide information on potential trends with which releases are currently popular, or perhaps what color is being identified frequently. Users and the database itself will benefit greatly from the additional information this tool can provide.

The resulting product will encapsulate standalone interface, a demonstration database and a modular processing record identifier. The record identifier will accept two images, it will combine knowledge gathered about both images, it will then find specific Discogs release entries based on the details gathered to efficiently provide a collection of the most likely releases the images identified. able to most efficiently resolve the request.

Building, installing and maintaining an application of this critical nature will require an investment of $47,000 with yearly maintenance costs totaling to $9,400 or 20%. This pricing relies on the utilization of development tools and libraries available at no additional cost and with no associated licensing fees.

To implement this solution, I will be relying on academic training that culminated in a Bachelors of Computer Science, with proficiency validated through independent certifications from CompTIA, Oracle, Microsoft and Coursera. My academic training has been refined by more

than 10 years of experience in information technology and a long history of artificial intelligence project implementations.

Sincerely,

Codi Smith

**Project Recommendation**

**Problem Summary**

An Inaccurate Discogs user collection can cause a user to believe they already have something they don’t and that they don’t have something that they already do. Collections are solely created by user input of a releases database entry. When the user input is not always reliable you run the risk of potentially missing out on a big score (record purchase). Misidentification of a release can cause a variety of issues for collectors. Not knowing the fair price of a record because different versions of the same album could have price discrepancies, believing a misidentified release you added is rarer than it actually is, buying the same variant of the same release multiple times are just a few examples of issues misidentified database entries could cause.

In resolving this, the solution will provide reliable database entries to the users of Discogs. It will require periodic labor investment for maintenance as new relevant information becomes available or existing information or functionalities become outdated.

**Application Benefits**

The solution will provide a list of database entries identified solely by two images. The interface will provide access to explanatory details, allowing instantaneous evaluation of the driving factors behind the selected release entries to determine, with confidence, which database entry best represents the vinyl record in the images. This reduces the burden of human input on a collection’s accuracy.

**Application Description**

The application will present an authorized user with their collection and they will have the ability to choose two images, one of a vinyl record and the other of its cover. They will be able to find the release entry in the Discogs database of the vinyl record in the images. Use of this process, on an internally hosted application, protects user information while maximizing the benefit gained from that information.

**Data Description**

To operate this record identifier, existing data in the Discogs database will be needed. Confidential user and proprietary information will be needed to log in. The loading of this application will require populating the database with the release entries for each release in the user’s collection, these entries will contain data on the artist’s name, the album’s name, the color, album art, and other details. The application will also include sets of past searches. Past searches will contain the tools found release entries and the identified color, the images, and the words that helped find the artist and album. The quality of the provided images is critical to the proper functioning of the record identifier.

The demonstration dataset will be composed of two image sets of a randomly selected albums from Rolling Stone’s 500 Greatest Albums of All Time, (Rolling Stone, 2021), as well as sets of images of albums selected from my personal collection. Where practicable, unnecessary information will be parsed out and retrieved as needed.

**Objective and Hypothesis**

Resulting from this project, a data processing algorithm will accept two images; one of a vinyl record and the other of its cover. These images can be used to find their release entry in the Discogs database. Operation of the algorithm is premised on the theory that an album can be identified by its album art and a vinyl records color can be identified by an image, leading to its database entry being obtainable with less human input. This can be used to help users manage their collections more reliably and the search result could provide more useful statistics overall as well as providing real time information about a release.

**Methodology**

The traditional waterfall project management methodology will be used to develop this product. This project management style is the most cost-effective option for smaller projects in environments with fixed requirements over the life of the project.

The requirements phase of the waterfall method corresponds to the initial discussions and refinements of needed features for this project. This task will be accomplished using easily attainable images. The design phase of the project follows requirements gathering and clarification. This will see the application architecture determined and data models defined. Configuration of the database, programming environment and coding of the application itself will occur in the implementation phase, after the conclusion of the design process. Once implementation completes, the functions, modules and application will be tested in the verification phase, ensuring proper behavior across both intended and unintended

inputs. The project will pass into the maintenance phase where any bugs missed in verification and any regulatory concerns can be resolved.

**Funding Requirements**

This application will run on each individual machine, accessing the existing Discogs database at no additional cost. It will require dedicated software engineer resources totaling to $47,000 over the course of 188 working hours at $250 per hour. The project will utilize development tools and software libraries available at no cost and with no associated licensing fees. Completing this project will cost $47,000 in total with expected quarterly maintenance averaging to $2,350 per year with $9,400 per year in labor.

Annual maintenance costs are expected to average to 20% of installed cost.

**Stakeholders Impact**

The application will allow users to more easily and reliably add recently obtained vinyl records to their collections using image searching, this feature could attract new users, creating more revenue for the company. The use of the information this tool obtains, if used in unison with other existing database information, could open opportunities to create tools for other areas of the Discogs systems, such as a way to track search popularity trends in areas such as genre, artist, or region; it could also potentially be used to make a recommendation system for the marketplace based on similar searches, increasing sales and in turn creating more revenue for the company.

**Data Precautions**

The data that will be used in the application is not sensitive, album information is publicly available as well as the images used to train the artificial intelligence model. The application is run on the local machine and needs authorization via web browser to launch using a Discogs account each time it is run.

**Developer Expertise**

This project will be completed by a degreed computer science engineer bringing more than 9 months of experience in the technical service industry; a wealth of knowledge and experience, supported by a variety of certifications, to the project.

The software engineer’s experience with project management, using the waterfall method, is underpinned by both ITIL 3 Foundation certification and CompTIA Project +. Validating this

experience with graphical user interface design technologies is CIW Site Development Associate. A Microsoft Professional Certification in Data Science and Udacity Machine Learning Nanodegree demonstrate expertise in application of machine learning techniques. These academic and professional certifications and achievements provide a strong foundation for completing this project as scheduled and budgeted.

**Prompt B**

**Project Proposal**

**PROBLEM STATEMENT**

An Inaccurate Discogs user collection can cause a user to believe they already have something they don’t and that they don’t have something that they already do. Collections are solely created by user input of a releases database entry. When the user input is not always reliable you run the risk of potentially missing out on a big score (record purchase). Misidentification of a release can cause a variety of issues for collectors. Not knowing the fair price of a record because different versions of the same album could have price discrepancies, believing a misidentified release you added is rarer than it actually is, buying the same variant of the same release multiple times are just a few examples of issues misidentified database entries could cause.

**CUSTOMER SUMMARY**

This application is intended to serve users of Discogs. Its purpose is to supplement their knowledge and input with information gathered and compiled from two user provided images. It will present a list of Discogs release entries with details about each release entry that are found as a result of the contents of the images. This opportunity to remove some of the human element from collection management, as well as gain the opportunity to review statistics of a collection, can provide these users reminders of overlooked details and opportunities to correct faulty decisions and associated knowledge.

The record identifier will be structured to be flexible in its deployment. The initial version will be targeted to run on any x64 version of Windows 10 and provide a python based graphical interface on the local machine. This application will assume users have basic web navigation skills and a Discogs account.

**EXISTING SYSTEM ANALYSIS**

Currently, users of Discogs use the general search bar to try and locate release entries of records they wish to add to their collections. Upon searching an artist, you can find all the albums they’ve released, and of all albums released, a release entry for each version of the release based on format, release data, and other things. At times, this method can cause a user to add the wrong release to their collection. At the completion of the project, the application will attempt to remove some of the human variable of user collection management. This application will use any x64 version of Windows 10 as its operating system. Python 3.9.7 will be installed with the following libraries: pillow, requests, discogs-client, matplotlib, google-search-results, beautifulsoup4, google, elasticsearch 5.5.3, imgurpython, and TensorFlow. The final, intended result of this project is to provide a method to identify the likely Discogs release entry a user wishes to add to their release from two images. If successful, this will improve reliability of the user’s collection accuracy.

**DATA**

To operate this application, the data required is publicly accessible via the Discogs Music Database, the only confidential information required is provided by the user logging into their account in an external browser and acquiring a verification upon initialization, this is not within the scope of this project. Use of this application will require populating the application with the details of the user’s collection and the releases in that collection, as well as past searches saved locally in a pickle file. The quality of this data is critical to the proper functioning of the record identifier. The demonstration dataset will be composed of image sets curated to resemble real music albums that exist in the world, each set will have the albums cover and a picture of the physical vinyl record. This dataset will be composed of famous albums and uncommon albums. As record identification searches are performed, information will be gathered to show statistics of the collection of the logged in user, as well as statistics of the record identification’s search results. Updates to the data housed within the application are updated as new releases are discovered. The application will include the interfaces required to view these statistics and all found releases in the current user’s collection and all releases found in past searches.

**PROJECT METHODOLOGY**

Over the course of this project, planning and scheduling will follow the waterfall method. Using traditional waterfall project management will see the application development process pass through requirements gathering, design, implementation, testing and maintenance. This method, while less flexible than alternatives and less robust when coping with requirements changes, is cost effective and efficient on small projects and those instances where requirements are expected to remain stable.

Requirements gathering includes the initial discussions about the needs of Discogs users and concludes when an agreement is reached on the feature set to be included in the application. Design follows requirements gathering and clarification. This will see the application architecture determined and data models defined. These phases will be completed prior to implementation of the application, when configuration of the programming environment and coding of the application itself will occur. Once implementation completes, the application will be tested in the verification phase, ensuring proper behavior across both intended and unintended inputs. As a result of this testing decision, extra time will be allotted during the testing phase to resolve bugs discovered. Finally, the project will pass into the maintenance phase where any bugs missed in verification and any regulatory concerns can be resolved.

**PROJECT OUTCOMES**

Upon completion of the project, the finished application will be delivered to the customer. The application will be included as a PyCharm project that when run launches a Python based graphical user interface, providing the user access to the implemented interfaces and allowing secure access to the verified users collection. In addition, the finished application will include the demonstration image sets, and a demonstration user for testing of the application’s individuality, as the application will show different information depending on the logged in user. For documentation purposes, the user guide will be provided, providing installation instructions and explaining the user interface. Finally, the project schedule with projected and actual milestone completion dates will be provided.

**IMPLEMENTATION PLAN**

Coding of the application will be completed in a top-down approach. This approach has been chosen to allow functionality verification of each module and to ensure that they interface correctly with the main application. First, images of vinyl records for the Tensorflow model will be gathered as well as the test image sets. Second, the data models for releases, artists, users, and searches will be defined and implemented, as well as methods to populate them with data from the Discogs database. Third, a login module will be created for providing secure access to user management. Fourth, a generic interface class will be established, providing generic pass-through functionality for testing the record identifying core algorithm. Each of the required modules will be constructed as stubs, derived from the generic interface. Fifth, the reverse image search module and then the Tensorflow machine learning model will be trained for the image classification module, upon completion of these the record identification module will be built saving the results using past search creating module. Sixth, the main GUI will be designed and implemented using the Tkinter library. Seventh, methods to populate the main GUI with user data, releases, and to maintain search data will be implemented. Eighth, the additional GUIs will be constructed for the visualization and maintenance tools, these will be attached to the completed main GUI. Upon completion of the GUI, each test task will be added to the database. As the processing steps are completed, their outputs will be logged as past search objects for examination, these will be saved in a pickle file. When errors occur, the responsible module will be examined to locate and correct the faulting logic before the entire test process is restarted.

Once functionality has been verified, deployment of the application to end users will follow a separate set of phases; environment setup and application installation. During the environment setup phase, any x64 version of Windows 10 x64 will be installed and all updates will be installed. The x64 version of Microsoft Visual Studio 2015, 2017, and 2019 will then be installed. PyCharm Community will then be installed. Finally, Python 3.9.7 will be installed and added to the Windows PATH environment, along with the dependency libraries.

Application installation will require creation of the permanent directory for the application and configuration of the Run/Debug Configuration of the application in PyCharm. Launching the application once the configuration is in place will allow it to launch correctly, store the releases and past searches found, and access the Tensorflow machine learning model. Finally, if the application is ready for deployment to production, the provided demonstration data and demonstration save data must be removed. Save data is automatically generated if the files are not found. During the development process, all portions of the environment setup can be completed. The development process has the same dependencies within application deployment.

Upon delivery of the application, the customer will also receive the installation guide and user guide. As extended maintenance is expected, documentation of repairs made, updates performed and enhancements applied will be provided after each maintenance event.

**EVALUATION PLAN**

The application will be validated using curated two-image sets. These image sets have been selected as representative of expected input, the first image is to be of the albums cover art and the second image of the physical vinyl record media. The Record Identifier is said to have been successful if the ideal Discogs release database entry is in the list of found releases generated by the applications use of the two images. In the case of the demonstration database, with 20 provided sets, success requires the ideal release entry to be found in 95% of the test sets. Overall, the application will be considered successful and reliable for end user needs if it can achieve 70% success over all the test image sets. The albums that were chosen to create image sets of were selected from, “Rolling Stone 500 Greatest Albums of All Time” as well as my personal collection; 10 image sets from each group, totaling 20 demonstration image sets. Each test set will be a folder in the test\_data folder with the artist’s name, album name, expected color, and expected release number to be found. The release number is to be made available in the Current Search list intended for the GUI. This record identifier acts as a tool to provide assistance to existing Discogs users in managing their collections, using information readily available to the user and organization. Concentrating it and analyzing it in this way does not create new liabilities under existing United States regulatory frameworks, nor does it interact meaningfully with existing industry standards.

**RESOURCES AND COSTS**

For this application, a single host computer will be required. Assuming a computer is already accessible, no additional hardware or application costs apply.

The libraries and applications used in this project are available for use without additional licensing costs.

Development, testing, and deployment are anticipated to require 180 hours of labor, costing $45,000. Final delivery is anticipated to require 8 hours of labor, costing $2,000, as accounted for in the initial project schedule and budget.

**TIMELINE AND MILESTONES**

Development and deployment are anticipated to require 180 hours of labor over the course of 10 weeks. This schedule is a result of outside constraints on the schedule of the assigned software engineer. Completion of this project is dependent on the milestones described below.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Mile- stone | Pre-requisi tes | Activity | Resource Assigned | Hours | Start | End |
| 1 | - | Requirements approval | Project Manager | 8 | 6/30/21 | 7/7/21 |
| 2 | 1 | Architecture design | Software Engineer | 8 | 7/8/21 | 7/12/21 |
| 3 | 1 | Interface design | Software Engineer | 8 | 7/13/21 | 7/16/21 |
| 4 | 2, 3 | Development environment configuration | Software Engineer | 4 | 7/19/21 | 5/20/21 |
| 5 | 4 | Test image dataset and Tensorflow dataset set curated. | Quality Assurance | 8 | 7/21/21 | 7/23/21 |
| 6 | 4 | Class implementations | Software Engineer | 8 | 7/26/21 | 7/28/21 |
| 7 | 6 | Login module | Software Engineer | 4 | 7/29/21 | 7/29/21 |
| 8 | 7 | Discogs database entry parsing modules | Software Engineer | 4 | 7/30/21 | 7/30/21 |
| 9 | 8 | Identifier reverse image search module | Software Engineer | 12 | 8/2/21 | 8/4/21 |
| 10 | 8 | Tensorflow image classification module | Software Engineer | 12 | 8/5/21 | 8/10/21 |
| 11 | 4, 7 ,8, 9,  10 | Past search creation module | Software Engineer | 8 | 8/11/21 | 8/12/21 |
| 12 | 11 | Record identifier module | Software Engineer | 4 | 8/13/21 | 8/13/21 |
| 13 | 12 | Record identifier GUI | Software Engineer | 16 | 8/16/21 | 8/18/21 |
| 14 | 12 | Visualization modules | Software Engineer | 4 | 8/19/21 | 8/19/21 |
| 15 | 13 | Visualization GUI | Software Engineer | 4 | 8/20/21 | 8/20/21 |
| 16 | 13 | Maintenance GUI | Software Engineer | 4 | 8/20/21 | 8/20/21 |
| 17 | 5, 6, 7,  8, 9, 10,  11, 12, 13 | System testing | Quality Assurance | 40 | 8/23/21 | 8/27/21 |
| 18 | 17 | Application deployment | Software Engineer | 8 | 8/30/21 | 8/31/21 |
| 19 | 18 | Application verification | Software Engineer | 16 | 9/1/21 | 9/6/21 |
| 20 | 19 | Final project delivery | Software Engineer | 8 | 9/7/21 | 9/8/21 |

**Prompt C**

## Application Files

\Record\_Identifier

\application\_files

\record\_color\_model\\* The preprocessed TensorFlow Keras model that is used to predict the color of the vinyl record in the provided media image.

\vinyl\_record\_colors\\* The image dataset I made that is used to train the TensorFlow Keras model that is used to predict the color of the vinyl record in the provided media image.

\image\_unavailable.png The default image used at initialization of the application and also when a release’s artwork cannot be found.

\saved\_data

\all\_releases.pkl Contains all releases downloaded by the application so they do not need to be downloaded every time the application is used.

\past\_searches.pkl Contains all past searches performed by the application so that they can be referenced any time the application is used.

\test\_files\\* The sets of test images used to test the functionality of the application.

\gui.py User interaction handling.

\main.py Application initializer; presents a login window requesting verification. The main application will not load unless authorization is verified.

\maitenence\_gui.py The GUI and methods and functions of the maintenance tool.

\record\_identifier.py Contains the main algorithms used to find the Discogs release entries from images.

\Record\_Identifier\_Library\_Setup.bat A batch script to install the required libraries once Python is installed. (Only works for Python 3.9)

\release.py Class definitions of the release, artist, and past search classes.

\user.py Class definitions of the data structure and user classes.

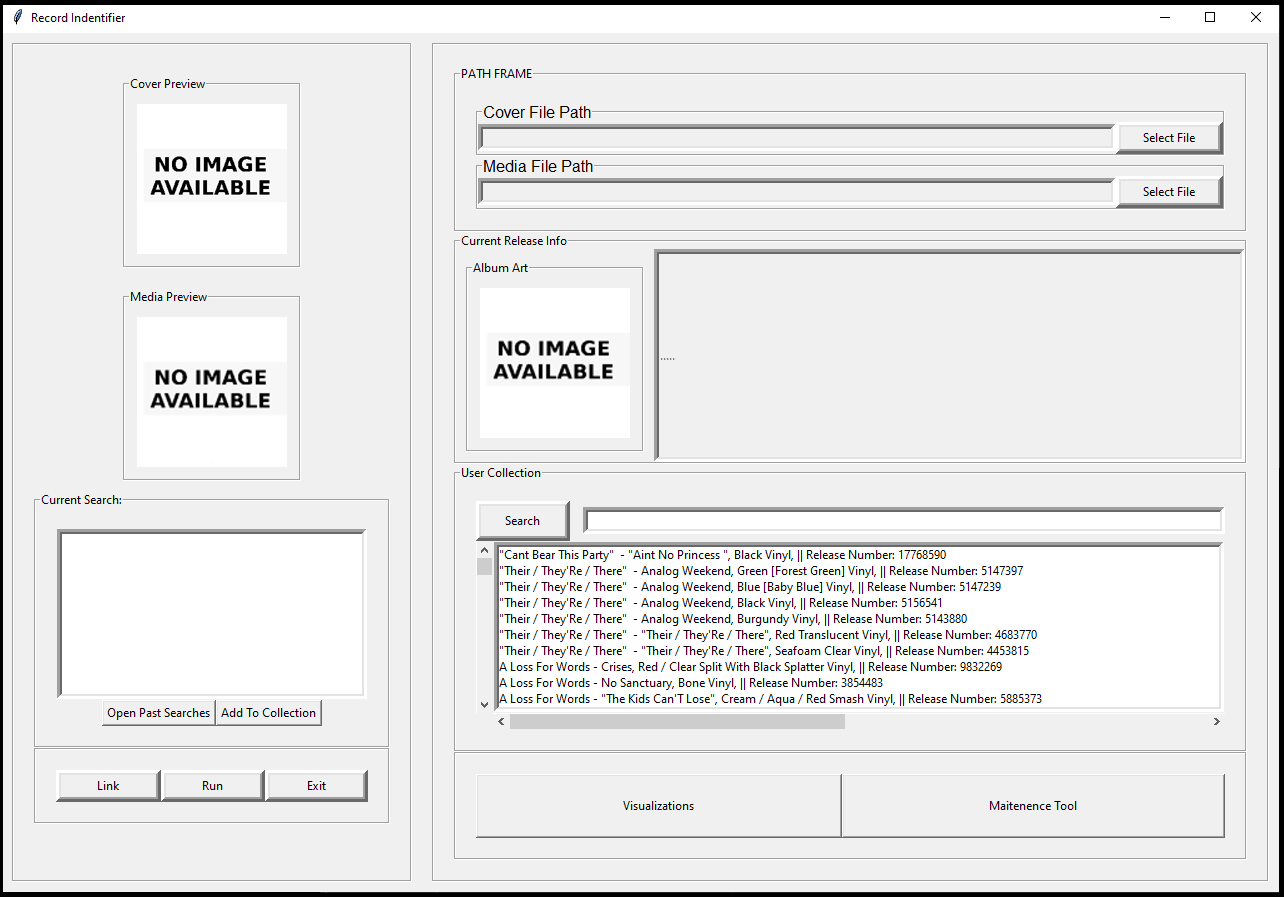
\visualization\_gui.py The GUI and methods and functions of the visualization tool.

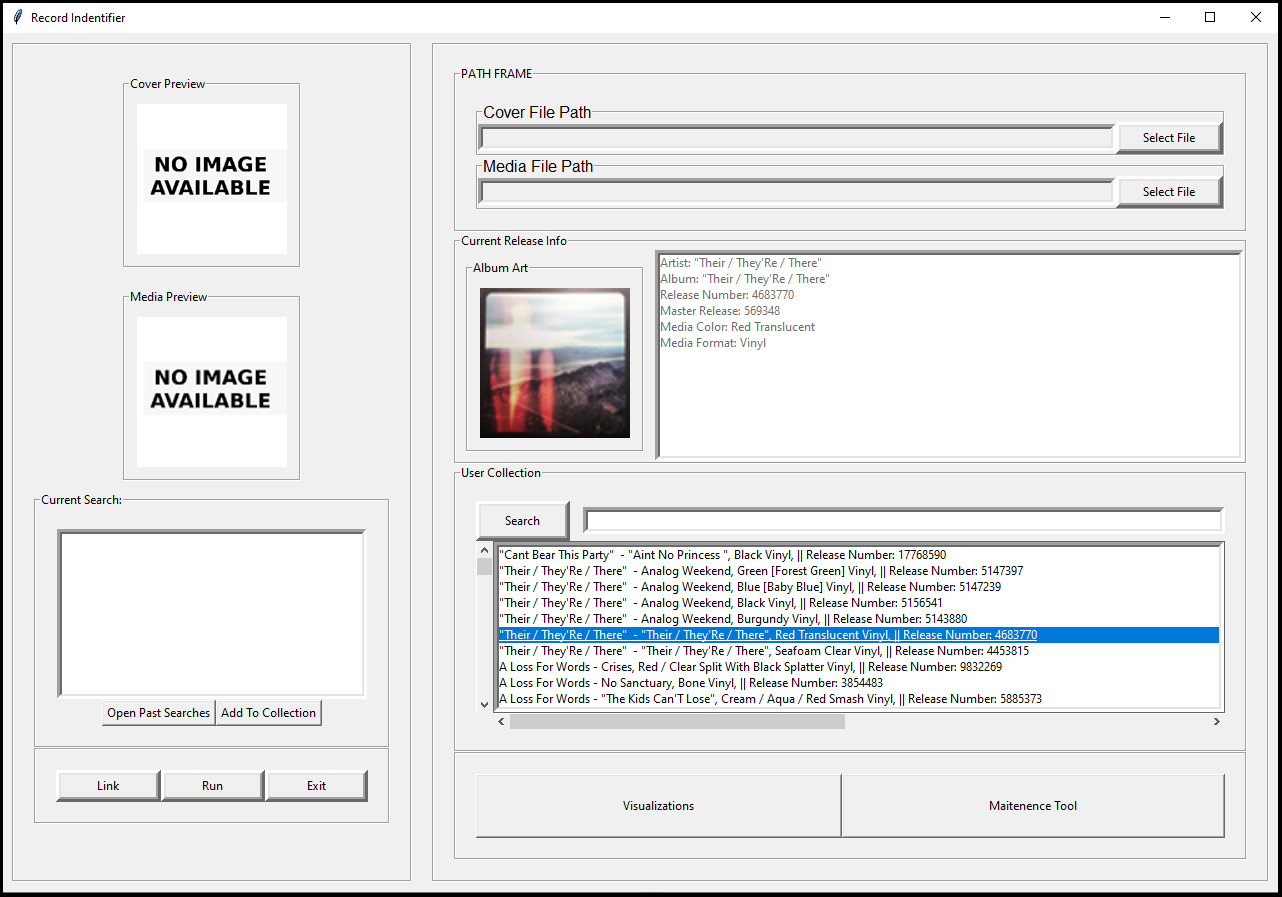
# Prompt D

## Post-implementation Report

### Project purpose

The record identification project provides a function to find the Discogs database vinyl release entry of two images it is provided. Once two images are selected in the image paths in the upper right, the “Run” button in the “Current Search” area can be used to run the core algorithm. The “Current Search” listbox is then populated with the found likely releases. These can be used to add the release to a user collection, reducing the human element of finding a release entry to add to a user collection, in turn raising reliability of a collection’s accuracy.



On the right side, the “Current Release Info” section will display the album art and details of the selected release in the “User Collection” area. The “User Collection” area also includes a search feature for checking releases already in the user collection. Below that is a button to open visualizations of the user collection and the tool’s search results, as well as a button to open a tool to check the connectivity of the necessary services that must be accessed for the tool work. 

### Datasets

The data required for the TensorFlow Keras image classification model that returns the color of the record are sets of vinyl images that are publicly available and were scrapped from the internet. The data for effective operation of creating release entries and generating the user collection are also publicly available via the Discogs API, authentication is used solely to add to the collection. As a result, data was generated using existing information with the aim to recreate the look and feel of the Discogs environment. Releases and past searches are stored in 2nd Normal Form. This format was chosen for easy accessibility of the primary keys that reference the release and past search objects.

### Data product code

In an effort to facilitate future modifications and to reduce exposure to systemic bugs, the record identification process was decomposed into discrete, disconnected steps. The steps, and therefore modules, included in the record identifier are accessing APIs to upload images and perform queries and use those results for text parsing and object initialization. The first two modules of the record identifier each receive one of the images.

The first module receives the image of the album art selected, the Imgur API is used to upload the image to the internet, a URL link to that image that is used as input for SerpAPI’s Google Reverse Search function to gather web links related to that image, this text is parsed for the most frequent words that appear in the returned web links.

The second module receives the image of the physical media selected and is used as input for the trained TensorFlow Keras model to identify and return the color of the vinyl record in the chosen image.

The third module uses the results of the first two modules to perform a google query, using python, of the string of the most common words returned by the reverse image search, the string of the color returned by the second module, and the words “Discogs” and “Vinyl”. The results of this google query will then be parsed to gather any Discogs release numbers found.

The application will then generate release objects based on the found release numbers and generate a past search object which can be used to add identified releases to the validated user collection. Past search results are saved and can be reloaded.

### Hypothesis verification

The design of this application was informed by the hypothesis that an individual release entry in the Discogs database could be found using two images. Using the gathered test dataset of 20 albums, the hypothesis could be confirmed as long as the album or artist was of some notoriety. The behavior of the system and performance levels that were reached suggest that the hypothesis should not be rejected. Improvements to any part of the identification algorithm, or to the reverse image search algorithm, could show significant improvements in the performance of the record identifier.

### Effective visualizations and reporting

As an application for the management of a collection, viewing statistics about the collection is of primary importance to the collector. In this regard, from the dashboard, the user is able to quickly access a chart of how many releases they have per artist, as well as a chart depicting the number of artists in their collection that they have a certain number of releases for.

In the same dashboard, is a pie chart that depicts the frequency of the media color that the TensorFlow Keras model has identified. These statistics could lead to opportunities in the future to use the acquired data to create new tools within the application.

### Accuracy analysis

For the accuracy metric for this application, it was determined that a recommendation would only be considered successful if the correct Discogs release entry was within the found release entries. Of the 20 test image sets, the record identifier achieves that level of success 100% of the time. By this predetermined metric, on randomly selected albums, the record identifier performs at reasonable levels. Review of the data and predictive behavior suggests that greater differentiation between technical discrepancies may improve identification performance dramatically.

### Application testing

Using a top-down development methodology allowed the functionality of each module to be tested as part of the integrated whole throughout its coding. This resulted in the input and output of each module being inspected at the database level to ensure the expected results were achieved. Each modification to a module required resetting the demonstration database and re-executing the record identifier. This proved to be more labor intensive than implementing unit testing. The resulting record identification application has been confirmed to behave as intended with each module accepting the expected inputs and providing the expected outputs.

# Appendices

## Installation Guide

Prerequisites:

* Microsoft Windows 10 x64
* Microsoft Visual Studio 2015, 2017, and 2019:
  + <https://support.microsoft.com/en-us/topic/the-latest-supported-visual-c-downloads-2647da03-1eea-4433-9aff-95f26a218cc0>
  + Use the x64 version, reboot required after installation.
* Pycharm Community 2021:
  + Download and Install Pycharm Community: <https://www.jetbrains.com/pycharm/download/#section=windows>
* Python 3.9.7 with supporting libraries:
  + Download Python 3.9.7: <https://www.python.org/downloads/>
  + Before clicking the “Install Now” button, check “Add Python 3.9 to PATH”
  + Click “Install Now”, once finished close the installer

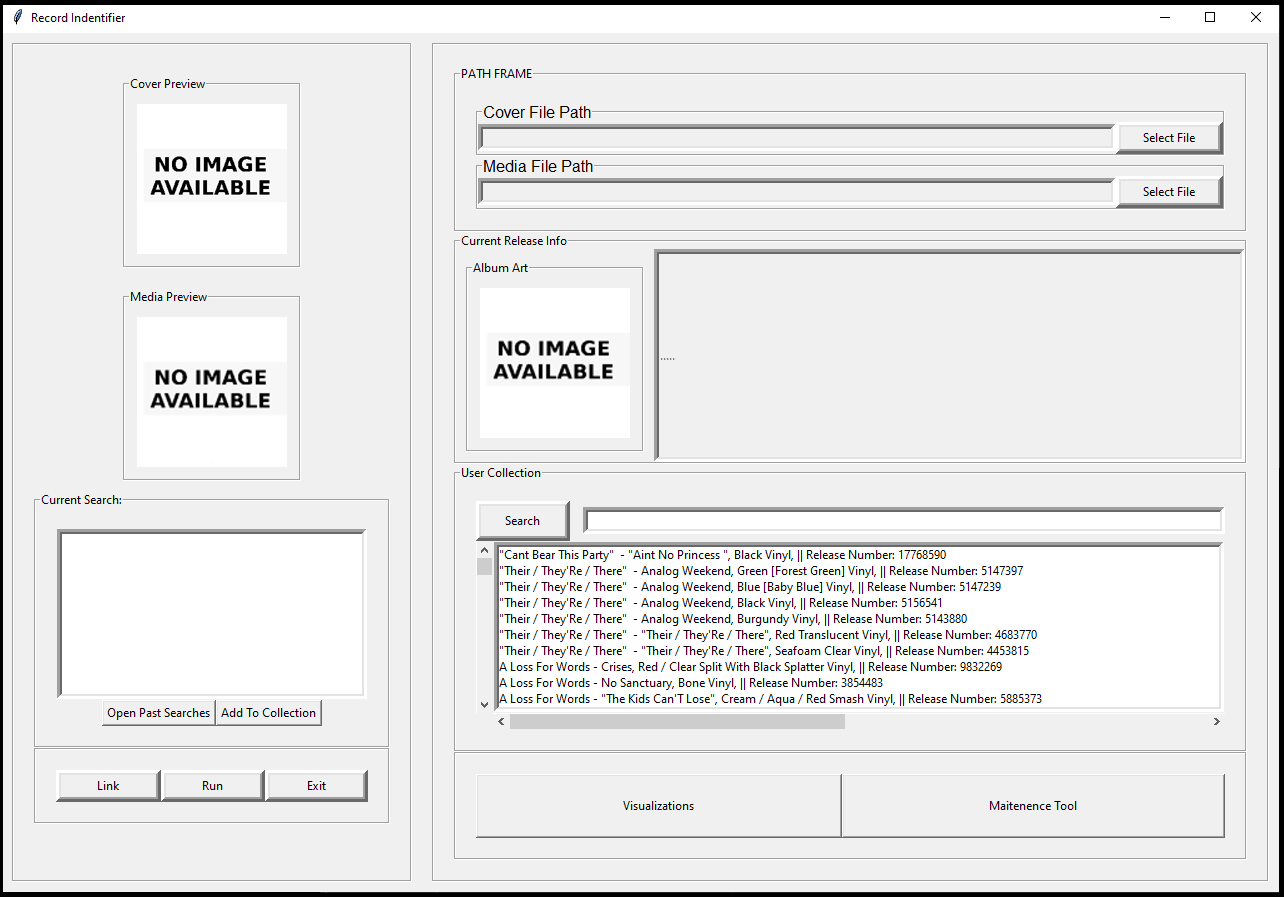
\* A batch file (Record\_Identifier\_Library\_Setup.bat) has been included in the project folder \* \* to automatically perform the following two steps. \*

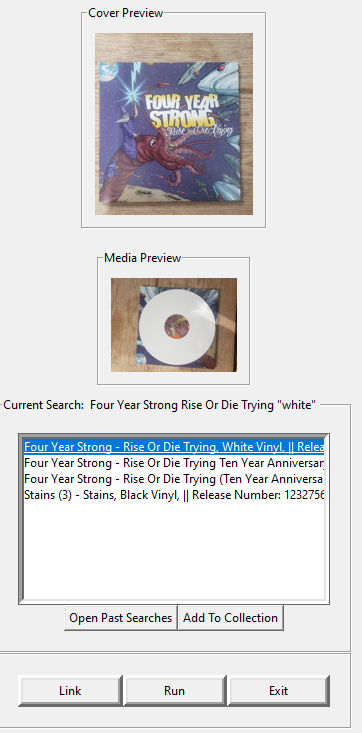
* + Pip install the following libraries: pillow, requests, discogs-client, matplotlib, google-search-results, beautifulsoup4, google, elasticsearch 5.5.3, imgurpython, and TensorFlow
  + Manually update the models.py file of the discogs-client library from the source GitHub: <https://github.com/joalla/discogs_client/blob/master/discogs_client/models.py>

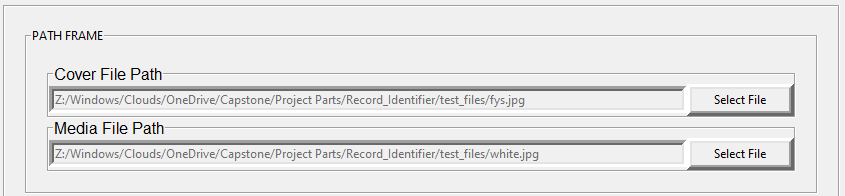
1. Install the prerequisite applications in order.
2. Extract Record\_Identifier.zip into the directory from which the application will run.
3. Open PyCharm Community and select the “Open” button in the Projects tab.
4. Navigate to location of the extracted Record\_Identifer folder, select it, and click OK.
5. In PyCharm’s upper right toolbar, select the “Add Configuration” button.
6. In the popup window, click “Add new run configuration”
7. Double click Python.
8. Select main.py as the Script path and ensure that the Python Interpreter is set to “Python 3.9”
9. Click “Apply”. Then click “OK”.
10. If not already logged in in your default browser, log in to Discogs.com using one of the user accounts below:
    1. Username: iamcodismith Password: WGUtest2021
    2. Username: wgu-test Password:WGUtest2021
11. Click the “Run” button in Pycharm to launch the application.
12. A Discogs webpage will be loaded, click “Authorize” and copy the verification code.
13. See the User Guide for directions on how to use the software.

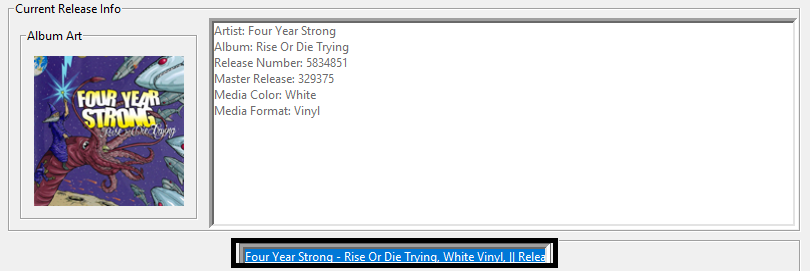
## User Guide

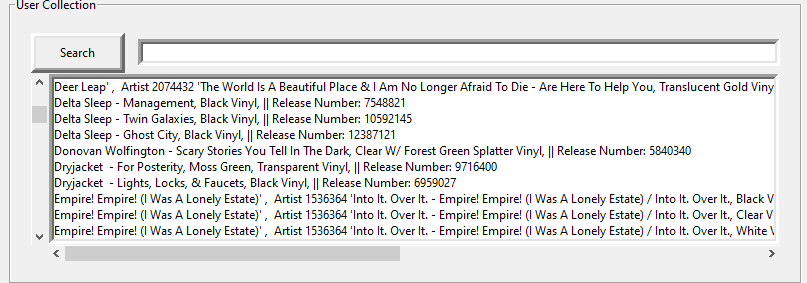
The application will begin with a login prompt asking for a verification code. The application will launch your default web browser and open a link to a Discogs authorization page, if you are not logged in you will be asked to login. Once logged in, the authorize button will provide the verification code and it can be copied and pasted into the login GUI.

Once verified, the application will open to the main dashboard which will have 5 main sections.

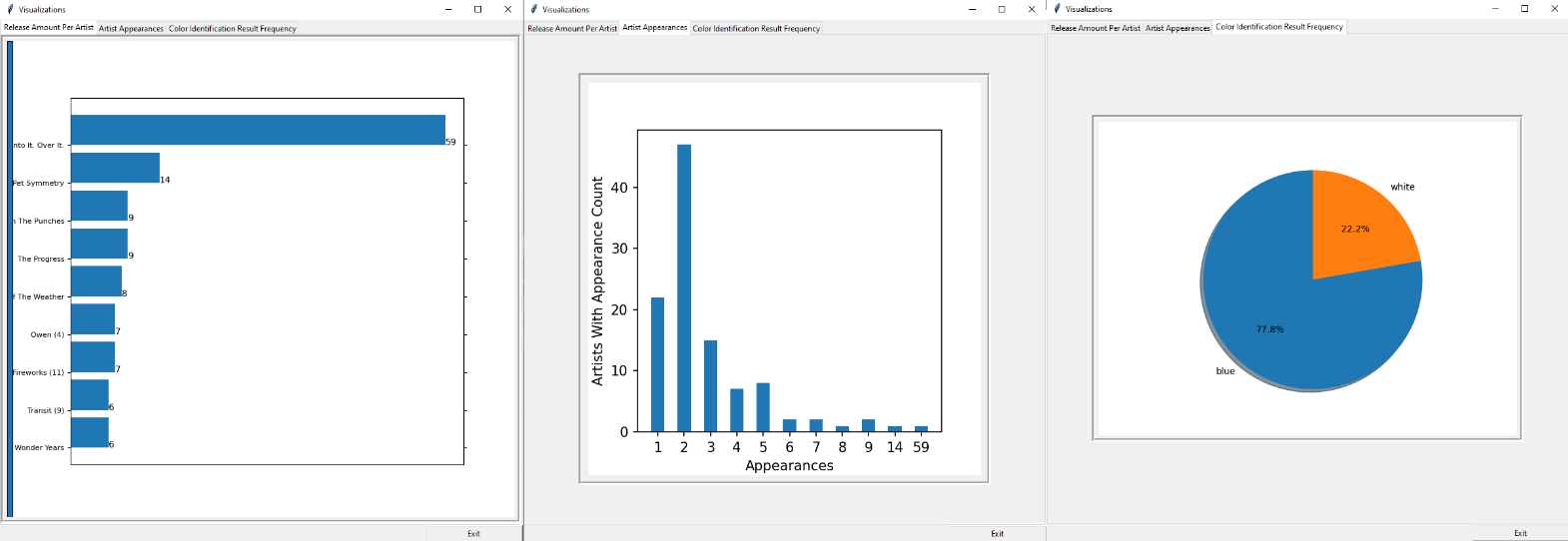
 Section one is the left side displaying the preview images of the searches, initialized to blank images, below it, is the area that will display a list of the release entries found by the search, with a button to open past searches and another button to add the most recently selected release from the search entry list to the user’s collection as long as the release is not already in their collection. Below that frame is a “Link” button to open the most recently selected search release in browser on the Discogs website, a “Run” button to run the record identification tool once the images in the path section have been selected, and an “Exit” button to close the application.

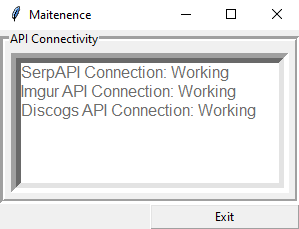
The second section begins in the upper right, the path frame is where the input images will be selected. The first field is for the cover of the record, the second path is for the actual physical vinyl record. Both must have a selected .png, or .jpg image file for the “Run” button to run.

The third section is on the center right and displays the album art and information about the most recently selected release entry from either the current search list or the user collection list.

The fourth section contains a list of all the releases in the verified user’s collection and has a search feature.

The final section houses a button to open the visualizations and another button to open the maintenance tool to ensure the functionality of required resources.

The visualization window has three tabs to display 3 visualizations. 

The maintenance window displays the connection status of each API that is required for the application to function correctly.

## Summation of Learning Experience

In approaching this assignment, I was again confronted with my own issues with troubles with vague directions. My educational experiences to this point and desire to be productive with my time encouraged me to choose a task that would solve an issue I encounter in my daily life. Learning to build graphical interfaces in Python using the Tkinter library gave me a grounding in graphical interfaces I previously lacked. This was the project where I first felt like I was effortlessly creating object classes in such a way to manipulate data and to build methods and function that I envisioned. Finally, building multiple applications from scratch, resulting in functional interfaces, gave me the confidence to approach a more serious developmental challenge that I, as a developer and an end user, could also benefit from.

I chose to build this application in python, a choice that required I learn python-specific graphical interface technologies. Learning to use machine learning for image classification and how to use external APIs to access external databases and function libraries was the most rewarding experience of this project. For this, I chose Tkinter, a simple framework that would support the project development. Learning the Tkinter ecosystem also introduced me to matplotlib, providing a significantly improved visualization interface compared to the external display of visual data using a different display framework. Learning these additional libraries and techniques from the documentation was a challenging and enlightening experience. Aside from checking in with my course instructor and program mentor every week or two, this work was completed without assistance. This project has encouraged me to continue expanding my repertoire of programming knowledge and to seek certification of those skills.

## References

Rolling Stone. (2021, August 13). *The 500 Greatest albums of all time*. Rolling Stone. https://www.rollingstone.com/music/music-lists/best-albums-of-all-time-1062063/.