Automated Annotation of Source Code for ML Applications

DESIGN DOCUMENT

Team 20
Arushi Sharma: Client
Maxwell Sutcliffe: Client Communication Coordinator
Robby Rice: Digital Content Coordinator
Gavin Canfield: Quality Control
Tanner Dunn: Agile Framework Organizer
Amon McAllister: Individual Component Design

sdmay23-20@iastate.edu
https://sdmay23-20.sd.ece.iastate.edu/

Revised: 12/02/2022 / 1.0

Executive Summary

Development Standards & Practices Used

- IEEE 2841-2022: This standard is called the "IEEE Approved Draft Framework and Process for Deep Learning Evaluation." If group ends up creating a new machine learning algorithm specific to our needs, we will need to carefully consider this standard on how we evaluate our algorithm.

Summary of Requirements

- Program an application to take in source code in Java or Python and output a tagged label file for component of source code
- Create a tagged source-code dataset
- Implement a variety of machine learning algorithms
    - Compare with metrics such as accuracy

Applicable Courses from Iowa State University Curriculum

- COM S 228: Basic Java knowledge
- COM S 311: Algorithm design
- COM S 472: Background for some members in artificial intelligence

New Skills/Knowledge acquired that was not taught in courses

- Data Engineering
    - Data scrubbing
    - Modeling data for machine learning
    - Dataset creation
- Machine Learning
    - Jupyter Notebooks
    - Machine Learning Pipeline Design

Table of Contents

List of figures/tables/symbols/definitions (This should be the similar to the project plan)

1 Team

1.1 Team Members

Maxwell Sutcliffe – Software Engineering

Robby Rice – Software Engineering

Gavin Canfield – Computer Engineering

Tanner Dunn – Software Engineering

Amon McAllister – Software Engineering

1.2 Required Skill Sets for Your Project

The only skill preferred on arrival to this project was knowledge of both Java and Python code. The rest (machine learning background, lexers, parsers, etc.) was researched the first month of the project.

1.3 Skill Sets covered by the Team

Tanner Dunn, a member of our team, had some experience in machine learning beforehand. Gavin Canfield had experience from COM S 342: Principles of Programming Languages class on lexers and parsers.

1.4 Project Management Style Adopted by the team

We decided to iterate with the agile methodology. This methodology allowed us to make continuous changes in a very dynamic way, especially as the schedule for our project was not extremely rigorous and allowed for a lot of flexibility.

1.5 Initial Project Management Roles

Maxwell Sutcliffe – Client Communication Coordinator

Robby Rice – Digital Content Coordinator

Gavin Canfield – Quality Control

Tanner Dunn – Agile Framework Organizer

Amon McAllister – Individual Component Design

2 Introduction

2.1 Problem Statement

What problem is your project trying to solve? Use non-technical jargon as much as possible. You may find the Problem Statement Worksheet helpful.

Source code contains many different types of "tags" or words that belong to groups that perform certain functions. With the assistance of machine learning and natural language processing, we hope to create an algorithm that will correctly predict what "tag" any one word in source code belongs to, then, with the help of machine learning, provide a quick comment on what the code is doing. There is a small suite of tools available already, but our client (and PhD advisor) wants to expand these tools. These tools are trained by neural networks that are essentially a "black box" and don't reveal the connections they make to the engineer. The point of the PhD's research is to uncover these relationships and see if they are predictable.

2.2 Intended Users and Uses

Users:

Arushi Sharma, PhD student

User details:

Getting their PhD and doing research delving into the depths of machine learning. Dealing with the bleeding edge.

Needs from the project:

Need to be able to see the inside workings of the machine learning to better operate and manipulate the algorithm.

Benefits from the project:

Due to their nature as a PhD student this will help them push machine learning to depths that it has never been before.

Machine learning engineers

User details:

Employed/ salaried engineers with presumably many years in the industry, working on cutting edge technology.

Needs from the project:

Source code, documentation on how we got our data, training and test data, results.

Benefits from the project:

Implementation into their own projects, inspiration for their projects, contribution to the project and valuable insights.

Machine learning enthusiasts/ hobbyists

User details:

Interested in machine learning, most likely a technical background already.

Needs from the project:

Easy-to-read narrative on the project, illustrations, well-documented source code.

Benefits from the project:

Help further understand machine learning, inspiration to begin working on a project, easy to read and digestible information in natural language processing.

2.3 Requirements & Constraints

Functional Requirements	- End product must include annotation of source code. - Automated annotation tool must correctly chunk and label sections of code.
Resource Requirements	- If we end up training a neural network, sufficient GPU power is needed to train large models and evaluate performance.
Aesthetic Requirements	N/A
User Experience Requirements	- Labels must be either clearly marked and understood or explained.
Economic Requirements	N/A
Environmental Requirements	N/A
UI Requirements	- Documentation on how to run various scripts must be provided.

2.4 Engineering Standards

What Engineering standards are likely to apply to your project? Some standards might be built into your requirements (Use 802.11 ac wifi standard) and many others might fall out of design. For each standard listed, also provide a brief justification.

IEEE 2841-2022: This standard is called the "IEEE Approved Draft Framework and Process for Deep Learning Evaluation." We must eventually analyze the processes and performance of our algorithm if we end up creating a new one. This standard will help in creating the framework and infrastructure we need to evaluate it.

3 Project Plan

3.1 Project Management/Tracking Procedures

We're using **agile. Our project is highly dynamic and calls for changing requirements within each respective sprint. The waterfall method isn't incredibly useful to us as we must often go back to what we were doing and retest or reevaluate.**

What will your group use to track progress throughout the course of this and the next semester. This could include Git, Github, Trello, Slack or any other tools helpful in project management.

Trello. This is a service that allows for a Kanban board and lets us see who is doing what on any given sprint.

3.2 Task Decomposition

Timeline	Tasks
Semester 1	- Decide on tags for dataset - Create manually tagged dataset - Create RegEx (or other) tool to automatically tag source code - Collect large sample dataset - Evaluate how dataset will work; contextualize data and start work on neural network implementation - **Final deliverable**: open-source tool for source code dataset creation available to a wider public
Semester 2 (mostly stretch goals)	- Refine dataset/ how data is created - Experiment with several machine learning models and compare

	- Address limitations and create documentation for future work

3.3 Project Proposed Milestones, Metrics, and Evaluation Criteria

What are some key milestones in your proposed project? It may be helpful to develop these milestones for each task and subtask from 2.2. How do you measure progress on a given task? These metrics, preferably quantifiable, should be developed for each task. The milestones should be stated in terms of these metrics: Machine learning algorithm XYZ will classify with 80% accuracy; the pattern recognition logic on FPGA will recognize a pattern every 1 ms (at 1K patterns/sec throughput). ML accuracy target might go up to 90% from 80%.

In an agile development process, these milestones can be refined with successive iterations/sprints (perhaps a subset of your requirements applicable to those sprint).

What are some key milestones in your proposed project?

Key Milestones

- The automatic dataset creation will tag 100% of tokens it comes across.

- The open-source tool will offer at least two ways of tagging Java source code.

- We will create ~100 tagged source code sets to allow for optimal training data.

3.4 Project Timeline/Schedule

• A realistic, well-planned schedule is an essential component of every well-planned project

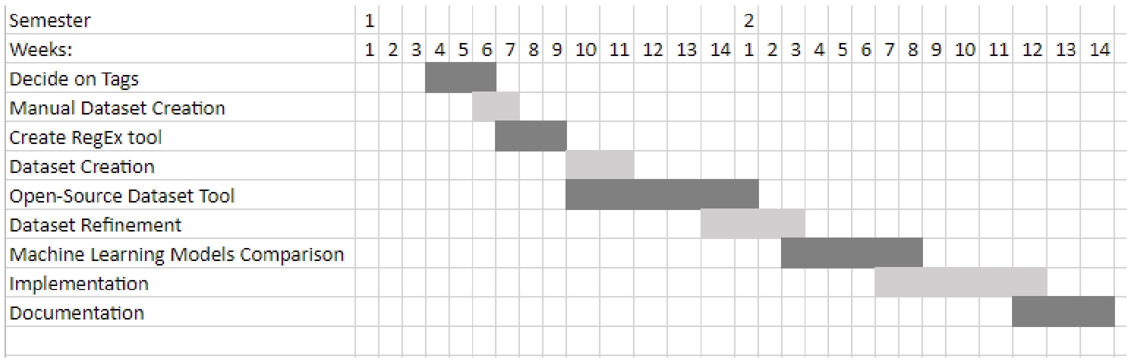
• Most scheduling errors occur as the result of either not properly identifying all of the necessary activities (tasks and/or subtasks) or not properly estimating the amount of effort required to correctly complete the activity

• A detailed schedule is needed as a part of the plan:

– Start with a Gantt chart showing the tasks (that you developed in 2.2) and associated subtasks versus the proposed project calendar. The Gantt chart shall be referenced and summarized in the text.

– Annotate the Gantt chart with when each project deliverable will be delivered

• Project schedule/Gantt chart can be adapted to Agile or Waterfall development model. For agile, a sprint schedule with specific technical milestones/requirements/targets will work.



Semester	1														2													
Weeks:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Decide on Tags				▓	▓	▓																						
Manual Dataset Creation						░	░																					
Create RegEx tool							▓	▓	▓																			
Dataset Creation										░	░																	
Open-Source Dataset Tool										▓	▓	▓	▓	▓	▓													
Dataset Refinement														░	░	░	░											
Machine Learning Models Comparison																	▓	▓	▓	▓	▓	▓						
Implementation																					░	░	░	░	░	░		
Documentation																										▓	▓	▓

3.5 Risks And Risk Management/Mitigation

Consider for each task what risks exist (certain performance target may not be met; certain tool may not work as expected) and assign an educated guess of probability for that risk. For any risk factor with a probability exceeding 0.5, develop a risk mitigation plan. Can you eliminate that task and add another task or set of tasks that might cost more? Can you buy something off-the-shelf from the market to achieve that functionality? Can you try an alternative tool, technology, algorithm, or board?

Task	Risks	Mitigation Efforts
Decide on Tags	N/A, largely a research task	N/A
Manual Dataset Creation	Incorrect Tagging, 0.2	N/A
Create RegEx Tool	RegEx not possible for a certain tag, 0.4 Tool is inaccurate, 0.6	We have multiple ways for creating datasets (line by line vs word by word) which can help with RegEx limitations. With an inaccurate tool, this will delay the schedule. Time constraints aren't too dangerous as semester 2 is mostly stretch goals.
Dataset Creation	Cannot find enough source code, 0.3	Create our own source code
Open-Source Dataset Tool	Tool not ready by Gantt chart deadline 0.2	Time limitations aren't crucial to deliverables

Dataset Refinement	Dataset doesn't fit into model, 0.7	Significant time setback. Readjust Gantt chart and reevaluate sprint deadlines.
ML Models Comparison	N/A	N/A
Implementation	N/A	N/A
Documentation	Documentation is lost, 0.1	Store digitally and rewrite.

3.6 Personnel Effort Requirements

Include a detailed estimate in the form of a table accompanied by a textual reference and explanation. This estimate shall be done on a task-by-task basis and should be the projected effort in total number of person-hours required to perform the task.

Member	Effort (weekly effort, task focus)
Robby Rice (Digital Content Coordinator)	Sem 1: ~3 hrs/ week, RegEx Tool Sem 2: ~2 hrs/week, Documentation
Maxwell Sutcliffe (Client Communication Coordinator)	Sem 1: ~3 hrs/week, Tag Deciding Sem 2: ~3 hrs/week, ML Models Comparison
Gavin Canfield (Quality Control)	Sem 1: ~4 hrs/week, Dataset Creation Sem 2: ~3 hrs/week, Open-Source Dataset Tool
Tanner Dunn (Agile Framework Organizer)	Sem 1: ~3 hrs/week, Dataset Creation Sem 2: ~4 hrs/week, Implementation
Amon McAllister (Individual Component Design)	Sem 1: ~3 hrs/week, RegEx Tool Sem 2: ~ 2 hrs/week, Implementation

3.7 Other Resource Requirements

We may potentially need a GPU to run ML calculations on.

4 Design

4.1 Design Context

4.1.1 Broader Context

The following is a list of areas in which our project may affect second-handedly:

Area	**Description**	**Our Answer**
Public health, safety, and welfare	How does your project affect the general well-being of various stakeholder groups? These groups may be direct users or may be indirectly affected (e.g., solution is implemented in their communities)	Our tool will lead to more quality datasets for developers working on code-to-text applications. This goes on to their applications, possibly making society better as a whole.
Global, cultural, and social	How well does your project reflect the values, practices, and aims of the cultural groups it affects? Groups may include but are not limited to specific communities, nations, professions, workplaces, and ethnic cultures.	Our work is tied to a more specific community, that being the development community. Worldwide effect will be made by machine learning eventually, and our tool will be a very small subset in that.
Environmental	What environmental impact might your project have? This can include indirect effects, such as deforestation or unsustainable practices related to materials manufacture or procurement.	Increased energy usage by a deep neural network.
Economic	What economic impact might your project have? This can include the financial viability of your product within your team or company, cost to consumers, or broader economic effects on communities, markets, nations, and other groups.	Our project may save machine learning engineers time (and therefore money for their org) by providing better datasets and a way to obtain datasets.

4.1.2 Prior Work/Solutions

– There are existing projects that make up components of what we are trying to accomplish. There is an existing project called ‘jflex’ that we have utilized to get an idea of how we are going to break up and label our source code. We have also used some preexisting datasets of chunked code as a baseline for what our datasets could look like.

– In the aforementioned project ‘jflex’ we have used that to our advantage to see one method of what we want to accomplish. The disadvantage is that the existing project is not as granular as we want it to be, and we will have to use our own method to match our specs.

Pros

- Get an idea of how we will use regular expressions to assign labels to fragments of code
- We can compile a nice dataset to compare against the dataset we will eventually use

Cons

- The existing project does not use our exact labels and it is a general idea of the final product
- We will have to create our own parser using different regular expressions to get more granular labels.

4.1.3 Technical Complexity

Our project is technically complex as it pertains to analyzing code that is written by humans, chunking it in possibly numerous differing ways. This chunked code will be used as data for a machine-learning algorithm. Analyzing code is quite a complex task due to the numerous ways that the code could be chunked and the vast amount of edge cases that java has. Machine learning algorithms are not trivial either since there are or always will be numerous ways to accomplish the same goal. Different programming patterns can be used to achieve the same or similar results. Once both tasks are completed, we will need to examine the results and reexamine the chunking method to maximize results.

Both things combined, coupled further with the reiteration to ensure a quality product, are going to make this technically complex.

4.2 Design Exploration

4.2.1 Design Decisions

Language

Due to this project being centered around programming, we needed to determine what language we should use since it would be unproductive, to say the least, if we all used different ones. Our client narrowed down the selection significantly by saying we should use either Java or Python. We chose Java as it is more verbose and gives us more to work on. We also are more familiar with Java.

Chunking

This project's final output includes chunking existing code into a more condensed syntax, which will later be used by the client. As a result, this decision cannot be made lightly and will most likely include many iterations and trying multiple options and may even include some capability to alternate between them depending on further direction from client.

4.2.2 Ideation

Chunking Decision

Since this design decision is vital to the project's success and outcome, the potential options were brainstormed with us and the client. We identified six different ways to chunk the code.

- Word by word
  - Creating a chunk for every single line of code.
- Common Phrases
  - The most abstract one on the list and probably the most complex, but it would identify common programming patterns and phrases such as a guard clause or something on a broader scale, such class inheritance.
- Locality
  - This would chunk the code based on the variable locality. For example, there is the file locality, class, function, if and for loop localities to be considered.
- Functions
  - Like locality but for just the functions in the code.
- One line
  - Chunking each line and it function such as for loop start or assignment or decrement etc.
- Many lines
  - Like one line but would attempt to group similar lines since often times lines next to each other are often similar in purpose.

Language Decision

When deciding between using java or python source code to create labels, the biggest thing we compared is the syntax. When it comes to creating an algorithm to label source code the more well defined and structured the programming language's syntax is, the easier it appears to be to define an algorithm to parse through source code and create labels. Due to the structure and more symmetrical syntax that java requires, we determined java source code would be easier to label with an algorithm.

4.2.3 Decision-Making and Trade-Off

Thus far we have not had large or complex decisions to make as our project has been very dynamic. Which form of text chunking to use will be a future decision that is decided upon the results of our machine learning model results. Due to this we have decided to use multiple text chunking methods that are defined in the previous section that we can test and decide upon later.

The main decision that we have made is choosing to use java or python source code. The decision process we used to do this was by getting together as a group and discussing/debating the programmability of chunking java or python source code. Through this extended group discussion, it was determined that due to the structure and symmetry of the Java programming language it would be easier to chunk out. Things like line statements ending with a ';', if statement conditionals being contained in '()', and loops being contained in '{}' are all examples of syntaxial advantages that the Java programming language has that are not required in python. Through our group discussion/debate over the two options it became clear that Java's syntax would be very advantageous to the chunking of source code and made the decision easy to go with Java.

4.3 Proposed Design

Currently, the majority of our time has been creating the tool that takes input source code and outputs a label file. This, in the following section, we only detail these design decisions. Next semester will focus more on the machine learning pipeline using a dataset created by the tool.

4.3.1 Overview

We are creating an input program that allows users to give the program a file, that will then send the formatted output to a text file(tentatively). We take the user input, create an Abstract Syntax Tree with the help of a module called JavaParser, and output to a label file. The internals of each step are described below.

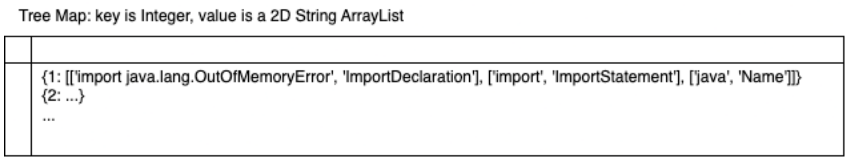
4.3.2 Detailed Design and Visual(s)

4.3.2.1: JavaParser

We first start with grabbing user input, asking the following questions:

1. Input file (stored in ./resources as our project is set up in Maven)
2. If label file already exists with that name, ask if overriding is okay
    a. Quit if user says no
3. Ask what level the user would like their label file
    a. High: only "larger" chunks of code. Omit things like names, variables, integers, etc. Keep features like import declarations, function declarations, if statements, loops, etc.
    b. Low: Include everything.

If user input is parsed correctly and we get expected values, we move on to creating the AST. We use the help of JavaParser to create this. After creation, we start to dissect the tree's nodes, again with the help of JavaParser. We use a data structure called a TreeMap to help us print later on. A TreeMap is a data structure native to Java. It is exactly like a HashMap in functionality, where you have a key of one type and a value of another. The fundamental difference is that a TreeMap will order your keys for you on insertion. This is vital when printing the label file. Our keys are Integers, or the line number for the token, and the value is a 2D String ArrayList. The outer ArrayList is a List of all tokens for that line. The inner String ArrayList is a tuple that contains the token itself and the token's assigned tag.

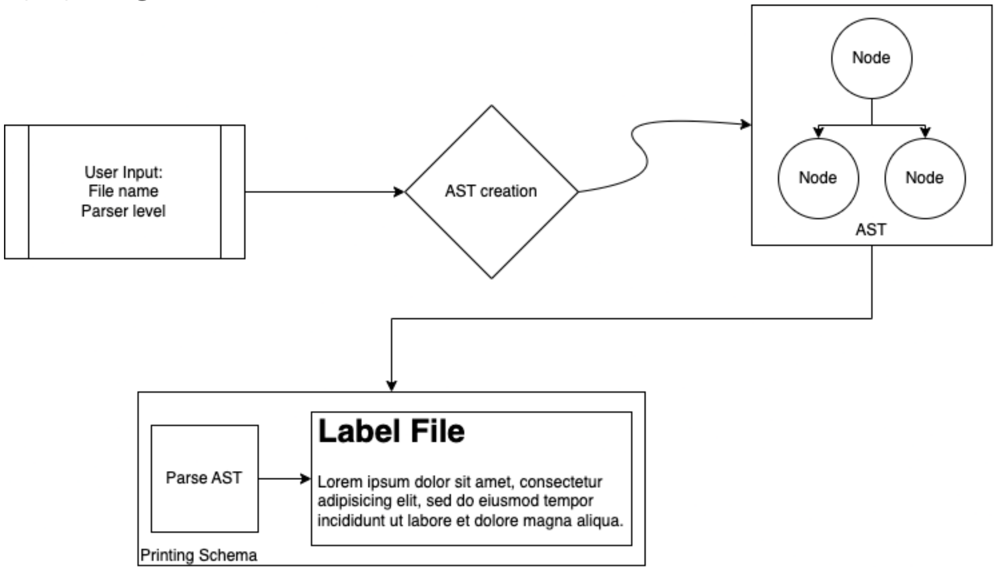

Tree Map: key is Integer, value is a 2D String ArrayList

```
{1: [['import java.lang.OutOfMemoryError', 'ImportDeclaration'], ['import', 'ImportStatement'], ['java', 'Name']]}
{2: ...}
...
```

4.3.2.2: Formatted Label File

Upon receiving the TreeMap data structure, the printing function will loop from 0 to the highest key. If there is no key for the line while iterating, we simply print the line number and continue. As of right now, we just print the contents of the ArrayList, but plan to change how this is done for ease of input into the machine learning pipeline. Some ideas are including metadata, changing to JSON, and more. Once the label file is done printing, we exit.

4.3.2.3: Diagram


User Input:
File name
Parser level
AST creation
Node
Node
Node
AST
Parse AST
Label File
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
Printing Schema

4.3.3 Functionality

Upon the completion of our system, a user will be able to input the data that they wish to be chunked. The data in question is intended to be files of code, they would then specify the level at which they would want it to be parsed. The user would specify the level at that they would want their data to be chunked which there are multiple levels; word by word, line by line, common phrases, locality, functions, or single line. The system upon receiving the data and the method in which to parse it will then turn the data into a dataset that is intended for use of a machine learning algorithm.

4.3.4 Areas of Concern and Development

Throughout the semester, a lot of time has been spent directly with our users to hash out the fine details of our project design. Currently, the design plans we have set in place will satisfy and meet all of our user's basic and highest-level needs. One of our biggest concerns for delivering this product is going to be the Machine Learning pipeline implementation. Our group has had little experience with the machine learning process, so we are expecting there to be a little bit of a learning curve when it comes to the implementation. We have discussed this concern with our stakeholders and are doing R&D in the meantime to prepare the best we can.

4.4 Technology Considerations

This project is being written in Java. Every member has experience using java and it has plenty of support and tools available online. One of those tools was initially Jflex which is a scanner generator. It's written in java for java and was used for generating a scanner used to analyze the code we were looking at. However, it proved time consuming to write the multitude of regular expressions necessary for what we needed. We then considered Antlr, a popular syntax parser, but decided it did more than we needed, resulting in us having to backtrack on a lot of work that could've been simple using a better tool. We finally found JavaParser, a tool that will simply parse code along with a suite of helpful tools for analysis, including generating the Abstract Syntax Tree for us. We decided to use this, and it proved successful.

4.5 Design Analysis

Recent setbacks in technology have set us a bit behind schedule in our technology considerations and planning, but after a critical look at the technology we were previously working and researching better solutions, we've found a more solid footing and a clearer idea about what our next steps are. We are more confident that our design will work. We will soon get more answers about the long-term success of this technology and what our data and machine learning pipeline will look like. The team has multiple prototypes ready and can start generating the dataset very soon.

5 Testing

Our overall testing philosophy in this project is to test whenever human error is possible and test the outcome of several machine learning algorithms. Most of our testing for the dataset creation tool will come from unit testing and integration testing. Unit testing is critical especially as we are dealing with external services that may be out of our control. This idea is also elevated to the project level – machine learning outputs are never certain, so we much be careful about the data we feed and the inputs of the system to ensure that even if the algorithm doesn't give us what we expect, it's still performing accurately. Basically, we need to make sure all data is clean and responsibly made so the factors out of our control later will be made in good faith. This type of testing is specific to our project as it involves every aspect of the machine learning pipeline, unlike other machine learning projects where data is already available.

5.1 Unit Testing

What units are being tested? How? Tools?

Below is a table of our current (and possible future) units that will undergo unit testing. They roughly correlate to the methods being used in source code for ease of unit testing.

Name	Description	Test Plan
User Input	Grabs user input file and asks for parameters for label file creation.	Give sample input and ensure file location exists and writes correct file output. Edge cases include empty strings and illegal characters that would interfere with label file creation.
Abstract Syntax Tree Creation	Creates an abstract syntax tree from source code.	Out of our control as we use a tool called JavaParser for this, however, we can test it with edge cases (empty .java files).
AST Classname Gatherer	Given an AST, we traverse through the AST nodes and collect the line number and type of feature.	Create Java files that do not follow convention but can technically compile. Test parameters (low vs high level classnames) are being correctly sorted.
Label File Creation	Given a classname TreeMap, print keys and values to label file.	Test what happens when label file cannot be created, or dictionary is empty. Edge case testing.

We will use JUnit tests to ensure that these unit tests work. Our project is currently using the Maven framework which easily incorporates this testing framework.

5.2 Interface Testing

What are the interfaces in your design? Discuss how the composition of two or more units (interfaces) are being tested. Tools?

Name	Description	Test Plan
AST Internals	Units related to internal AST creation and traversal.	These will be testing with pre-defined user input and sample files. This will test the accuracy of the main driver functionality – label creation.
User Interaction	Units related to user interaction and I/O.	Ensure I/O is properly accepted and does not interfere with other internals. Data scrubbing & cleaning. Dealing with file location and type.

5.3 Integration Testing

Our most critical integration was getting AST into our program. As stated above, we will test edge cases for our AST application (javaParser). These edge cases can simply be tested with Junit stubbing. We will have consistency tests that will run against a select few files that make sure we aren't getting any variability with the AST output.

5.4 System Testing

Our system level testing strategy has been TDD. There should not be any lines that should not have a test case. This will be apparent if a line of code has been deleted or altered and an error a test case will fail. We will be using Junit and Junit libraries.

User input testing

- We will test if a valid input has been given
- We will have test for in there is invalid or no input given

AST functionality testing

- There will be junit stubs testing that methods have been called
- We will be testing that the methods have been called with expected parameters

Consistency testing

- As afore mentioned, we will be testing files and there expected output
- These tests ensure that our AST integration is functioning as expected.

Output testing

- We will test output for valid and invalid inputs
- We will test output format
- We will test output destination

5.5 Regression Testing

We will run our unit tests each time a new feature or source code is added. We cannot have the main functionality of the label file creation application break, that being accepting user input and outputting a label file. Everything else added, whether it be a new feature, minor tweaks, or improvements, must make sure all unit tests pass. We will continue using JUnit for these tests.

5.6 Acceptance Testing

This is the most important part of our testing plan. We must ensure that label files are accurate and correlate to source files accurately. If this part is inaccurate, the machine learning algorithm will have inaccurate data and have a high bias. We will not move on until all our requirements are met. The requirements will be set out by our client, Arushi Sharma, who will give the “okay.” One requirement requires us to have a rate of 90% accuracy with label files. Accuracy, in this case, means not missing any features of code.

5.7 Security Testing (if applicable)

Not applicable – no live application.

5.8 Results

We have two main testing types involved in our project, JUnit and Acceptance Testing. Below is a table elaborating on our expected results and the requirements that will be verified by these test types.

Testing Type	Expected Results	Requirements Verified
Junit Testing	All JUnit tests will result in pass or fail. It will be expected that all JUnit tests pass. In any case where JUnit tests fail we will have to resolve the issue before pushing the code to our live version.	JUnit Tests ensure user use requirements are being met such as file uploading, as well as our AST integration. User requirements and AST functionality must be met at all times.
Acceptance Testing	We require and expect our label files to be accurate. Accuracy in this case will be	Label file accuracy is critical to the success of our project. These labels

	defined as 90% accuracy rate with our label files. This is to be verified by our advisor.	will be needed for our machine learning algorithms later on.

6 Implementation

We already have half of the proposed requirements done – the tool for creating the dataset is nearly done. While plans haven't concretely been made for next semester's implementation of machine learning, we've had half of our team working on research and examples for a machine learning pipeline. Once our dataset is created over winter break, we hope to start testing pipelines all of next semester to gather results and compare.

7 Professional Responsibility

7.1 Areas of Responsibility

IEEE code of ethics	How it addresses	Own words	Differs
To hold paramount the safety, health, and welfare of the public, to strive to comply with ethical design and sustainable development practices, to protect the privacy of others, and to disclose promptly factors that might endanger the public or the environment;	Health Safety and wellbeing and sustainability and social responsibility	Broadly speaking this helps ensure that the work that is being done is protecting people and environment	This one is pretty similar to one included in the NSPE although IEEE does point out that privacy protection is also part of ethics
To improve the understanding by individuals and society of the capabilities and societal implications of conventional and emerging technologies, including intelligent systems;	Communication Honesty	Make people aware of potential consequences of your work	IEEE seems to encourage a more open development process while NSPE is more concerned with deliberate conceit and deception

To avoid real or perceived conflicts of interest whenever possible, and to disclose them to affected parties when they do exist;	N/A	Ensure that the motives you have for working on a project are pure and if they are not then share that information with relevant parties.	N/A
To avoid unlawful conduct in professional activities, and to reject bribery in all its forms;	Social responsibility	Stay legal and don't be bribed	The IEEE is close to its NSPE counterpart with them both mentioning professionalism and legality however NSPE goes further and mentions enhancing the profession honor, reputation, and usefulness
To seek, accept, and offer honest criticism of technical work, to acknowledge and correct errors, to be honest and realistic in stating claims or estimates based on available data, and to credit properly the contributions of others;	Communication honesty	This one comes down to understanding that engineering in general is a collective effort with many people having many different points of views and perspectives that are valuable and in addition don't claim or estimate something you know isn't accurate	Both discuss the fact that lying is unethical, but IEEE discusses more collaboration and citing contributors as well.
To maintain and improve our technical competence and to undertake technological tasks for others only if qualified by training or experience, or after full disclosure of pertinent limitations;	Work competence	Do work that you are qualified to do or let relevant people know of limitations	Similar in the way that they both mention doing work that you are qualified and component at, but I do like how IEEE mentions what to do for a task that you are not already experienced/qualified for.
To treat all persons fairly and with respect, and to not engage in discrimination based on characteristics such as race, religion, gender, disability, age, national origin,	Doesn't really have an equivalent	Basically, don't judge people based on surface level stuff and don't discriminate people	Doesn't have an equivalent

sexual orientation, gender identity, or gender expression;			
To not engage in harassment of any kind, including sexual harassment or bullying behavior;	Social Responsibility	Don't bully or sexually assault people, generally just be excellent to each other	Not a super close comparison but they both deal with upholding laws and honor of the profession. Breaking this rule would not be honorable.
To avoid injuring others, their property, reputation, or employment by false or malicious actions, rumors or any other verbal or physical abuses;	Communication honesty Health safety wellbeing	Don't be mean to people on individual basis either verbally, or physically	Communcation honesty is usually for more public matters, but I feel it applies here as well since they both deal with lying. Health and safety are mentioned just to minimize risks which can be interrupted to be more of an active effort, but IEEE is more personal to you as in you should not hurt people which is more of a lack of an action.
Support colleagues and co-workers in following this code of ethics, to strive to ensure the code is upheld, and to not retaliate against individuals reporting a violation.	N/A	Help ensure others are ethical as well	N/A

7.2 Project Specific Professional Responsibility Areas

Area of responsibility	**Importance**	**Why**	**Current level**	**Why**
Work competence	Low	Due to the nature of this project, with us being in school and still very much learning, our competence level is already known	Low	The project that we are doing will be using Machine Learning, which our group has very little experience with. Due to our little experience, we have been working hard to set in place plans to learn and further our knowledge in this area so we can be competent and successful in our future work in this area.
Financial responsibility	High	If we don't manage to produce a product that pleases our client and does not have a realizable value, then this whole project was a failure	N/A	While we are making progress towards this, we do not have a finished product or service to be measured.
Communication honesty	High	I truly believe that just about any problem that happens in a professional environment was because something was not communicated truthfully or successfully	High	We have been open about what we are doing and our next objectives as well. We have communicated effectively as a team.
Health safety well being	Low	Due to our project being about the creation of datasets	Low	It's difficult to hurt people directly with datasets and machine learning.

		this is not relevant to us		
Property owner ship	Medium	I think that it is important to give credit where credit is due, but I also understand that when using other people's code, they usually know that they won't be credited and often times may even intend it that way	Low	We are not far enough in the project to know without a doubt what other people's code idea software etc. will make it to the end. It was only recently that jflex was kicked out and replaced with antler.
Sustainability	N/A	While computation can take a lot of resources, it would not be feasible for our project to handle it this issue.	N/A	N/A
Social responsibility	High	Due to the nature of our project being a tool to help other people use it for the purpose of assisting in machine learning projects it is vital that our product helps society and these communities	N/A	The same as financial responsibility in which we do not have a finished product to judge but I do think we are making progress.

7.3 Most Applicable Professional Responsibility Area

Communication honesty

As mentioned before, communication is essential to any project of any complexity and ours is no exception. It does not matter how great we are at anything else if objectives are not communicated between the team and the stakeholders. We were open and clear about what we were doing. What was going well, what wasn't going well, and finally, what was going to happen next. This has allowed us to get applicable advice and redirection from our client that has allowed the project to go smoother.

8 Closing Material

8.1 Discussion

The greatest result thus far has been our tool for crawling through source code, parsing results, and outputting a label file. This was the biggest hurdle in our project with respect to decision-making. We had many ideas technically but had to carefully consider all of our future plans and weigh the pros and cons.

8.2 Conclusion

We started the semester with a very broad and loosely defined goal of creating a dataset. While we didn't quite achieve this in time, we learned a lot about what goes into a large project such as this one and concluded that project guideline decision-making is almost more important than technical decision-making. We were impeded a lot by lack of clarity on some decisions we made about the scope of this project rather than technical details and implementation decisions. Going forward, we now know we need to be explicit in what's asked each sprint.

In summary, the "why" must be more defined while the "how" takes a backseat, something us software engineers may not be used to.

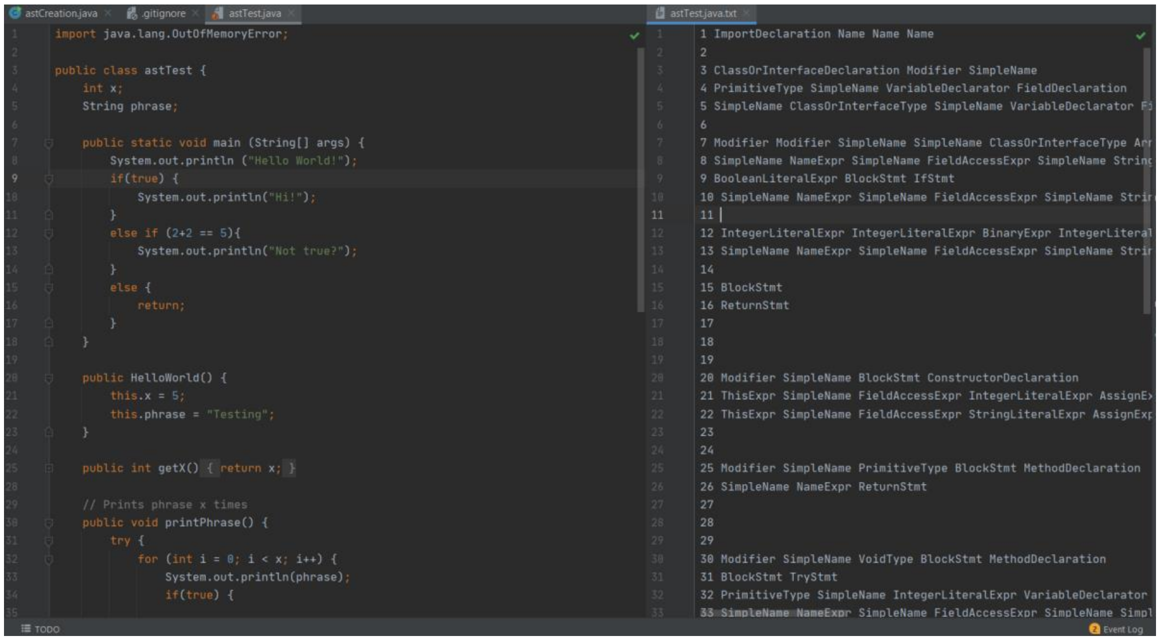
8.3 References

Klein, Gerwin. *JFlex*, https://www.jflex.de/.

"Hugging Face – the AI Community Building the Future." *Hugging Face – The AI Community Building the Future.*, https://huggingface.co/datasets?search=code_x_glue.

8.4 Appendices

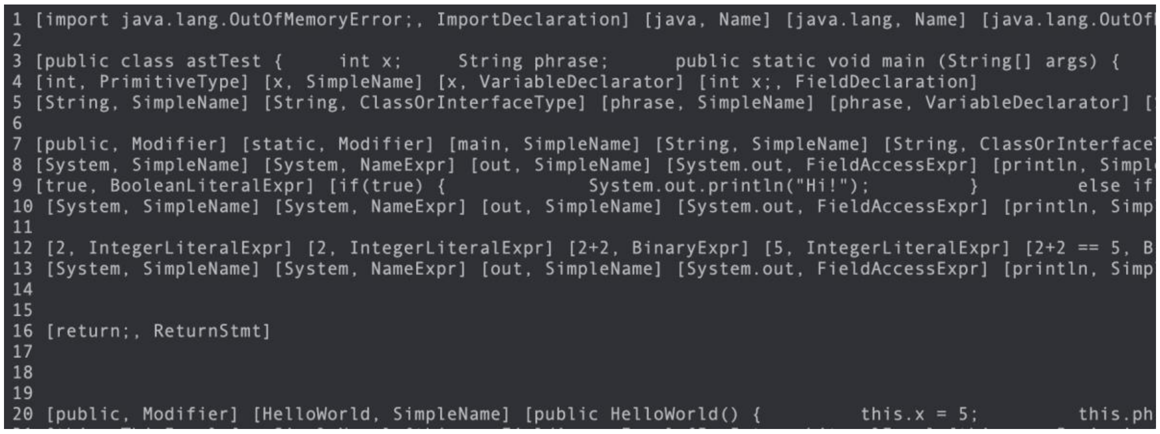
It may be helpful to include a visual of what our output may look like.



```java
import java.lang.OutOfMemoryError;

public class astTest {
    int x;
    String phrase;

    public static void main (String[] args) {
        System.out.println ("Hello World!");
        if(true) {
            System.out.println("Hi!");
        }
        else if (2+2 == 5){
            System.out.println("Not true?");
        }
        else {
            return;
        }
    }

    public HelloWorld() {
        this.x = 5;
        this.phrase = "Testing";
    }

    public int getX() { return x; }

    // Prints phrase x times
    public void printPhrase() {
        try {
            for (int i = 0; i < x; i++) {
                System.out.println(phrase);
                if(true) {
```

```
1 ImportDeclaration Name Name Name
2
3 ClassOrInterfaceDeclaration Modifier SimpleName
4 PrimitiveType SimpleName VariableDeclarator FieldDeclaration
5 SimpleName ClassOrInterfaceType SimpleName VariableDeclarator Fi
6
7 Modifier Modifier SimpleName SimpleName ClassOrInterfaceType Arr
8 SimpleName NameExpr SimpleName FieldAccessExpr SimpleName Strin
9 BooleanLiteralExpr BlockStmt IfStmt
10 SimpleName NameExpr SimpleName FieldAccessExpr SimpleName Stri
11
12 IntegerLiteralExpr IntegerLiteralExpr BinaryExpr IntegerLiteral
13 SimpleName NameExpr SimpleName FieldAccessExpr SimpleName Stri
14
15 BlockStmt
16 ReturnStmt
17
18
19
20 Modifier SimpleName BlockStmt ConstructorDeclaration
21 ThisExpr SimpleName FieldAccessExpr IntegerLiteralExpr AssignEx
22 ThisExpr SimpleName FieldAccessExpr StringLiteralExpr AssignExp
23
24
25 Modifier SimpleName PrimitiveType BlockStmt MethodDeclaration
26 SimpleName NameExpr ReturnStmt
27
28
29
30 Modifier SimpleName VoidType BlockStmt MethodDeclaration
31 BlockStmt TryStmt
32 PrimitiveType SimpleName IntegerLiteralExpr VariableDeclarator
33 SimpleName NameExpr SimpleName FieldAccessExpr SimpleName Simpl
```

Initial comparison and first label file prototype.



```
[import java.lang.OutOfMemoryError;, ImportDeclaration] [java, Name] [java.lang, Name] [java.lang.OutOf

[public class astTest {     int x;     String phrase;      public static void main (String[] args) {
[int, PrimitiveType] [x, SimpleName] [x, VariableDeclarator] [int x;, FieldDeclaration]
[String, SimpleName] [String, ClassOrInterfaceType] [phrase, SimpleName] [phrase, VariableDeclarator] [

[public, Modifier] [static, Modifier] [main, SimpleName] [String, SimpleName] [String, ClassOrInterface
[System, SimpleName] [System, NameExpr] [out, SimpleName] [System.out, FieldAccessExpr] [println, Simpl
[true, BooleanLiteralExpr] [if(true) {            System.out.println("Hi!");        }            else if
[System, SimpleName] [System, NameExpr] [out, SimpleName] [System.out, FieldAccessExpr] [println, Simp

[2, IntegerLiteralExpr] [2, IntegerLiteralExpr] [2+2, BinaryExpr] [5, IntegerLiteralExpr] [2+2 == 5, B
[System, SimpleName] [System, NameExpr] [out, SimpleName] [System.out, FieldAccessExpr] [println, Simp


[return;, ReturnStmt]



[public, Modifier] [HelloWorld, SimpleName] [public HelloWorld() {          this.x = 5;          this.ph
```

Most recent label file example.

8.4.1 Team Contract

Team Members:

1) Robby Rice
2) Maxwell Sutcliffe
3) Gavin Canfield
4) Tanner Dunne
5) Amon McAllister

Team Procedures

1. Day, time, and location (face-to-face or virtual) for regular team meetings:

We plan to meet weekly face-to-face on Wednesdays at 4:00pm with our advisor for our project. After our Wednesday meeting with our client we plan to have a schedule/planning meeting for the next week. Our group also plans to meet Monday's virtually at 4:00pm to collaborate and discuss the action items we have been working on for our sprint and to prepare for Wednesday's meeting with the client.

2. Preferred method of communication updates, reminders, issues, and scheduling (e.g., e-mail, phone, app, face-to-face):
    a. Discord is the preferred method of communication, with email being the secondary choice. We have our client included in a discord server to communicate with and ask questions as needed.
3. Decision-making policy (e.g., consensus, majority vote): Majority vote.
4. Procedures for record keeping (i.e., who will keep meeting minutes, how will minutes be shared/archived): Meeting minutes will be kept by one person for each sprint and switch to the next member after the sprint is completed (2 weeks). This will follow the order that our team members are listed in above.

Participation Expectations

1. Expected individual attendance, punctuality, and participation at all team meetings:

a. For weekly meetings full attendance is expected. If you cannot show up you must let everyone know in advance. In the case that a meeting is missed by the member, the member needs to plan a meeting with one or more of the other members to get a recap of the meeting and be filled in on what was missed.

2. Expected level of responsibility for fulfilling team assignments, timelines, and deadlines:
    - a. It is expected team members commit to their tasks assigned to them during the two-week sprint. Team members should be expected to clarify their exact task if they are unclear. Elsewhere in this documents outline what a contributor can do if they feel they cannot take on the work after it is decided.
3. Expected level of communication with other team members:
    - a. Communication made during weekdays is expected to receive prompt replies.
        - i. Discord
            1. Within 4 hours
        - ii. Email
            1. Within one day
4. Expected level of commitment to team decisions and tasks:
    - a. Once a decision has been made it is expected each member commits fully to it even if they originally objected to it. If after having truly attempted to follow through with the decision and issues arise, the decision may be questioned.

Leadership

1. Leadership roles for each team member (e.g., team organization, client interaction, individual component design, testing, etc.):
    - Tanner Dunn – Agile framework organizer – Create a sprint board and discuss with client to establish stories for each sprint.
    - Max Sutcliffe – Client Communication – Act as our team's mouthpiece when need be. I will be the go-to person if our client needs to get in touch.
    - Robby Rice - Digital Content Coordinator – Responsible for all digital content within our project, from planning documents, assignments, source code, training data, etc.
    - Gavin Canfield – Quality Control – Responsible for maintaining an acceptable level of professional organization of code developed as well as ensuring submitted code meets expected functionality.
    - Amon McAllister - Individual Component Design – Responsible for the structure and quality of written code to effectively solve the designed purpose.

2. Strategies for supporting and guiding the work of all team members:
    - a. Team members will be responsible for their work outlined in each sprint. It should be communicated at a reasonable time (not an hour

before a meeting) if a contributor doesn't think they can fulfill their work for that sprint.

b. If a team member feels lost in what they need to be doing, they should reach out to the team about discussing how to move forward on their story/ sprint.

3. Strategies for recognizing the contributions of all team members:

   a. We will be employing a Kanban board to track stories and sprints. This will be the easiest way to track who is doing what and what contributions a team member has historically made.

Collaboration and Inclusion

1. Describe the skills, expertise, and unique perspectives each team member brings to the team.

   - Tanner Dunn – Data Science Minor with experience in Machine Learning and the Machine Learning pipeline, as well as Tokenization and sentiment analysis. As well as one year of industry experience in an Agile environment.
   - Max Sutcliffe – Front End/ React skills, working with APIs as well as bottom up design experience. Familiar with the Agile process.
   - Robby Rice - Primarily Python, JavaScript (React), and microservice experience. Interest in ML and natural language processing and have done independent work looking into it. Used Agile in internship.
   - Gavin Canfield – Different major from others, computer engineering, provides a differing viewpoint on issues from someone with more experience on lower-level code. Experience in an Agile/SCRUM environment during internship.
   - Amon McAllister – Software engineer who has experience in immersive VR and creating serious games.

2. Strategies for encouraging and supporting contributions and ideas from all team members:

   a. Team members will be encouraged to ask questions and fill in knowledge gaps so we have a stronger project overall. If a team member doesn't feel like speaking up during a meeting, they can always expect an answer by sending a message in our Discord channel.

   b. All ideas will be considered and thought through. It's important not to discount any idea, especially as we all may approach a complicated problem such as machine learning in a different way.

3. Procedures for identifying and resolving collaboration or inclusion issues (e.g., how will a team member inform the team that the team environment is obstructing their opportunity or ability to contribute?)

   a. If the team member is comfortable, they may bring it up to the group. The group should be attentive and take the concern seriously. If they don’t feel they can voice their concern, they should go to the TA or the professor with their concern.

Goal-Setting, Planning, and Execution

1. Team goals for this semester:
    - Build a strong team with a positive atmosphere.
    - Meet deadlines and establish a strong relationship with our client.
    - Learn and follow Agile methodologies to outline and complete work.
2. Strategies for planning and assigning individual and teamwork:
    - Once individual experience and interests have been assessed, the team will plan long-term goals for the semester. After this, we break it down further into large steps we need to accomplish (decide on tags, create RegEx program to assign tags, etc.). We then split these large tasks into small, two-week sprints and assign team members stories, or small tasks, to complete. Team members will most likely work together on some of these stories.
3. Strategies for keeping on task:
    - While our sprints are two weeks, we will have a meeting in between the weeks to check on progress and assess if the work given is too much or too little.
    - Reaching out to contributors working on a similar task. As we all will be on the same sprint, it’s more likely than not we’ll have contributors working on adjacent or similar tasks. Whether it’s background knowledge missing or not knowing how our code base operates, team members will stick together to create a strong knowledge base and help those out who may not understand.

Consequences for Not Adhering to Team Contract

1. How will you handle infractions of any of the obligations of this team contract?
    a. Failure to uphold communication expectations will be met with:
        i. A group meeting will be held to communicate with the member about possible changes that could allow them to meet future expectations
    b. Failure to meet deadlines
        i. In the event a team member is unable to meet a deadline, he must notify the group at least 1 day in advance. In this situation, the member would face a punishment of an earlier deadline on the next task of at least one day. In the event the member repeatedly fails to meet deadlines, the member must provide daily updates on any progress made on future tasks.
    c. Unacceptable Work
        i. In the event that a team member produces work that does not meet the group’s standards, the team member will be notified

through one of our communication channels within 24 hours of the work being completed. This notification will include specific feedback about what aspects of the work are unsatisfactory and what needs to be done to bring the work to an acceptable level. The team member will have the next sprint to revise their work before the team must review it. If the work is still unsatisfactory, other team members will assist in the completion of the work as needed.

2. What will your team do if the infractions continue?
    a. The first step is to diagnose the problem to determine where the problem lies. Either with the work assigned, group dynamic, etc. Then the problem should be alleviated in some way either by redistributing workload if that's determined to be the issue, rearranging sub teams if it is a problem with group dynamic, etc.
    b. If infractions continue after attempts have been made to solve the underlying issues, then further action needs to be taken, which will be determined in a group meeting as a group for the specific member should this scenario come to actualization.

****

a) *I participated in formulating the standards, roles, and procedures as stated in this contract.*
b) *I understand that I am obligated to abide by these terms and conditions.*
c) *I understand that if I do not abide by these terms and conditions, I will suffer the consequences as stated in this contract.*
1) Max Sutcliffe DATE 23/09/22
2) Tanner Dunn DATE23/09/22
3) Robby Rice DATE 23/09/22
4) Amon McAllister DATE 23/09/22
5) Gavin Canfield DATE 23/09/22