**Table of Contents**

**1. Introduction………………………………………………………………………………...2**

**1.1 Project Goals……………………………………………………....………………………2**

**2. Project Scope……………………………....……………………………………………….2**

**2.1 API’s Used……………………………....…………………………………………………2**

**2.2 Exclusions……………………………....…………………………………………………2-3**

**2.3 Assumptions/Constraints……………………………....………………………………...3**

**2.4 Key Deliverables……………………………....…………………………………………...3-5**

**2.5 UML Diagram……………………………....……………………………………………...5**

**3. Work Breakdown Structure………………………………………………………………….6-7**

**4. Project Schedule……………………………………………………………………………...7**

### 

### 

### 

### 

### 

### **1. Introduction:**

This document outlines the design for the principal functions and details of the Plagiarism Checker for essays. Its details are currently being determined by the group members: Gavin Wong, Evan Khuu, Zara Khan, and Himesh Mistry.

**1.1 Project Goals:**The goal of this project is to provide teachers with an easy and efficient way to check for plagiarism. Simple to understand graphical user interfaces that prove useful when checking essays for potential plagiarism.

**2. Project Scope:**

The Plagiarism Checker functions as a tool for the use of teachers, comparing all essays inputted into the machine and returning a text file of all comparisons that are above a certain percentage. The program will have a Graphical User Interface and will open a dialog box that will allow the user to select the files. Similarity percentages will be calculated by comparing strings, and each essay will be split into sentences using periods and other punctuation marks as a delim. The matching words in each sentence will be given a percentage and the percentages of each sentence will be added up to calculate the total similarity score. Each line of the returned text file will contain the names of the two files being compared, as well as the percentage of similarity between the two. The user can specify the name of the file before it is created

**2.1 API’s Used**

* ArrayLists
* Javafx
* Apache (Used to read docx files)

**2.2 Exclusions**

* May not work for languages that do not use the western character system
  + Different delims/some characters appear as an alphanumeric code
* Code
  + The program will not run with code inputted
  + Requires different delims
* File Type
  + Will only take in docx file types as the program was designed with teacher use in mind
* Paraphrasing
  + The program will not be able to detect if the meaning of a sentence has been plagiarized and paraphrased. It will only be able to detect if a sentence has exact word similarity
* Citing Sources
  + Citations will be be counted in similarity percentage, this includes quotes
* Web Scraping
  + The program will not be scraping the web for comparisons, they will be restricted to the list of essays inputted by the user

**2.3 Assumptions/Constraints**

Assumptions:

* The user speaks English
* User has a working computer
  + Includes a modern java version (7+) and the system meets the requirements of jdk/jre
* User has an IDE installed
* User inputs a docx file
* User understands file pathing

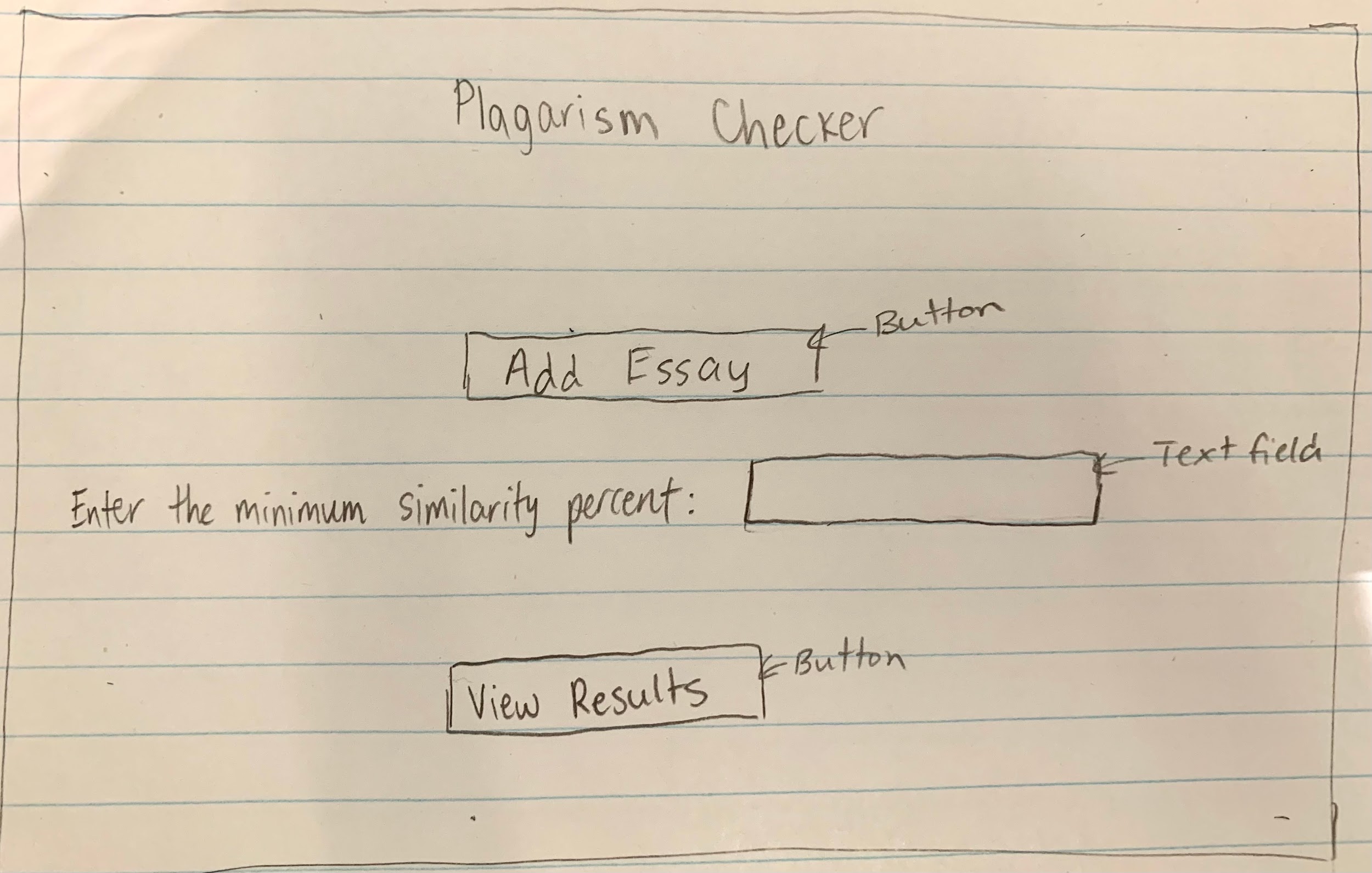
Constraints:

* Language
  + Using java on vscode for JavaFX

**2.4 Key deliverables**

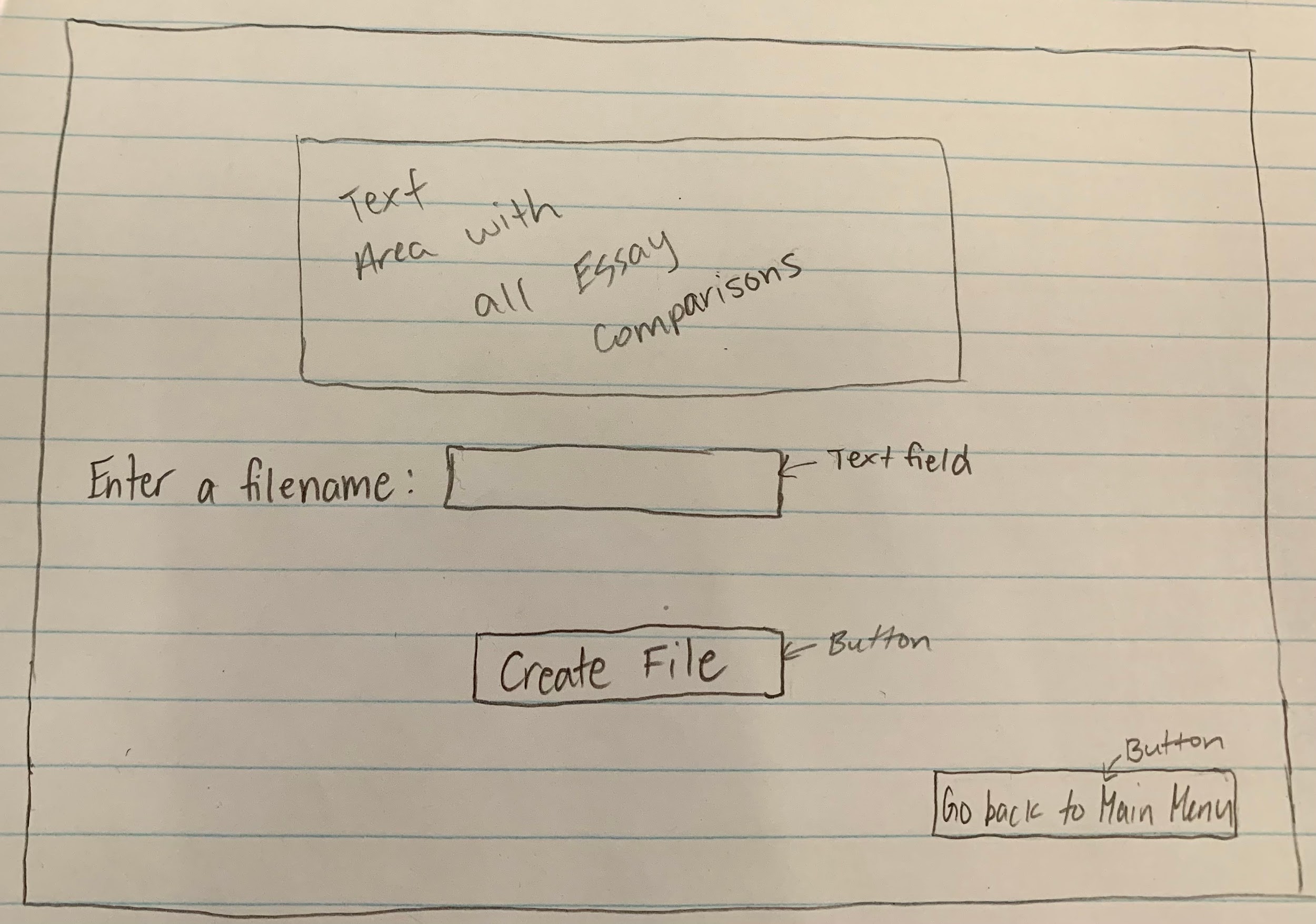
* **Main menu** 
  + **GUI**
  + **Add Essay button**
    - **Opens dialogue window**
  + **Text field**
    - **Scans for an int between 0 to 100 for the minimum similarity percent**
  + **View Results button**
    - **Calculate similarity percentages**
    - **Opens result menu**
  + **Set up controller class and methods that adds the specified essays to an arraylist once the button is clicked**

**Diagram:**

****

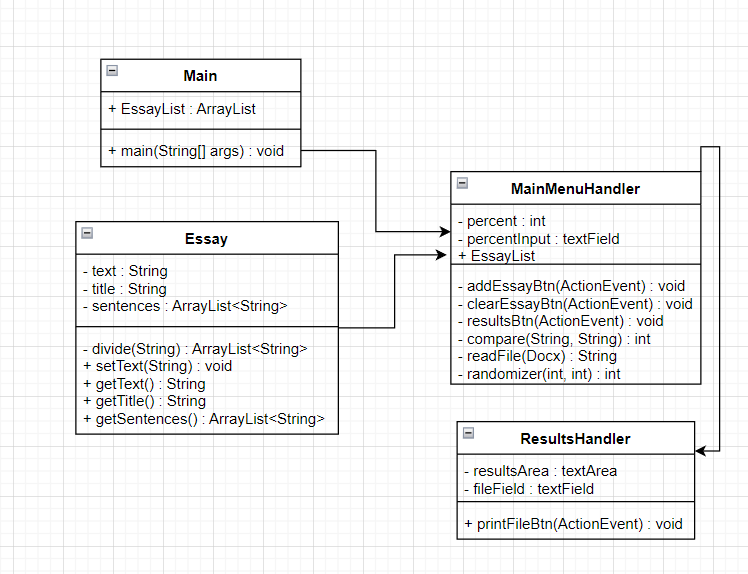
* **Result menu**
  + **GUI**
  + **Text Area**
    - **Contains all essay comparisons made that were above a specified percentage**
  + **Text Field**
    - **Will be used to select the filename of the result file created**
  + **Create file button**
    - **Will capture the current input in the text field and return a string value**

**Diagram:**

****

* **Result file**
  + **Takes and print the percentages of similarities between the files available and write the info into the file**

**2.5 UML Diagram**

****

**3. Work Breakdown Structure**

| **Task** | **Function description** | **Program requirements** | **Test case** | **Status** |
| --- | --- | --- | --- | --- |
| **Created all GUI Screens** | Visual display and activity | GUI   * Button * Dialog box   Main menu   * Dialog box to ask for the essay files used to be compared * Dialog box used to ask for the result file   Result menu   * Button to delete the files from the page |  | Incomplete |
| **essay arraylist** | Create an array list and store the essay data in it | A method that creates and convert essay data into array list |  | Incomplete |
| **Create Essay filereader and divider** | Read and write file | Filereader method that splits the docx file into strings and stores them in an arraylist  Filewriter  Method that loops through an entire arraylist before printing everything to a file of the users choice |  | Incomplete |
| **Create sentence comparison method** | Compares individual sentences for word percentage similarity | Int method to be run in a for loop or recursively to return a number percentage |  | Incomplete |
| **Monte Carlo sentence randomizer** | Used to find run and find the percentage when it comes to similarities | Method to generate random seed to determine which sentence indices to compare |  | Incomplete |

**4. Project Schedule**

| Task: | Created all GUI Screens | Link GUI screens to their respective controller classes | Create essay arraylist + button to add to arraylist | Create Essay filereader and divider | Create sentence comparison method and percent calculator | Create filewriter | Create  Monte Carlo sentence randomizer |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Member: | TBD | TBD | TBD | TBD | TBD | TBD | TBD |
| Date: | 1/10/2022 | 1/12/2022 | 1/10/2022 | 1/10/2022 | 1/12/2022 | 1/10/2022 | 1/14/2022 |
| Duration: | 30 min | 30 min | 1 hour | 1 hour | 1.5 hour | 30 min | 2 hours |