Venn Diagram Application

Software Design Document

Group 12

Version 3.0.0

04/15/2020

Table of Contents

1. Introduction 4

1.1 Purpose 1

1.2 Scope 1

1.3 Identification 1

2. Scope of Work 2

2.1 System Functions 2

2.2 Similar System Information 2

2.3 User Requirements 3

3. Dynamic Models 3

3.1 Class Diagram 3

3.2 Sequence Diagrams 4

4. Maintenance Scenarios 8

5. Appendix A: Record of Changes 10

List of Figures

Figure 1:	Class Diagram	7
Figure 2:	Main Class	8
Figure 3:	ControllerCopy Class	8
Figure 4:	Save Class	8
Figure 5:	CustomLabel Class	9
Figure 6:	Model Class	9
Figure 7:	Controller Class	9
Figure 8:	Sequence Diagram for Label Alignment	11
Figure 9:	Sequence Diagram for Text Colour Change	12
Figure 10:	Sequence Diagram for Adding a Label	12
Figure 11:	Sequence Diagram for Saving the Venn Diagram Application File	13
Figure 12:	Sequence Diagram for Loading the Venn Diagram Application File	14

List of Tables

Table 1:	Record of Changes	17

1. Introduction

The Software Design Document is a document to provide documentation which will be used to aid on understanding the software by providing the details for how the software was built. Within the Software Design Document are narrative and graphical documentation of the software design for the project including a sequence diagram and a class diagram.

1.1 Purpose

The Software Design Document's purpose is to provide information necessary to provide description of the details for the software and system that has been built in order to gain a clear understanding of the system's design.

1.2 Scope

This Software Design Document provides a description of the Venn Diagram Application's functionality to show feasibility for large scale production use. This Software Design is focused on the critical parts of the system.

1.3 Identification

- This software is based in JRE, the UI style uses java FX for procession, and all coding is done on eclipse
- The Venn Diagram Application's current version number is 3.0.0

2. Scope of Work

In this section, information of the functionality of the system and the stakeholder's technical requirements are available.

2.1 System Functions

An overview of the main functionality of the Application.

- User is able to type anywhere they click
- A text can be dragged to any location inside the boundaries of the diagram
- Properties of the diagram and text can be modified (size, colour, style, etc.)
- Text can be organized using component alignment
- Text can easily be deleted

2.2 Similar System Information

Almost all other applications have a drag and drop function, a feature that allows the user to modify text size, style, colour or the font itself. This feature enables the user to change not only the whole diagram colour and size but the individual circles as well. Some systems may also have utilized the ability for the user to disable the features the user does not want to use at any time. However, our application is a cut above other systems as this application can handle input and output at the same time, and the user has the freedom of structure.

This application handles user input and output at the same time, unlike other systems. The output is seen as the user is typing the input because this system utilizes text fields. Other systems require the user to enter data and select where on the diagram the output will be located. For example, another system may require the user to click an 'Add' button, then enter the text into an input field, select which circle it will be located, and only after the user hits 'Enter' will the output appear.

Another feature that shows our system is preferable over another is that the user has the freedom of structure. There is no set-in-stone structure that the user is

constrained to when using the system. Text can be dragged or placed anywhere, it can appear beside another text, below another text or the text can be placed whoever the user desires as there are limitless possibilities. However, if the user wants structure, there is a 'component alignment' function that aligned text in three different structural ways. Whereas in other systems, they utilize one specific structure, which the user is constrained to.

2.3 User Requirements

After the stakeholder tested prototype version 1.2.0 of the Venn Diagram Application, the stakeholder has created some additional requirements that our next version will incorporate, which are listed below.

- This system allows users to select multiple objects at once in order to customize them
    - E.g. select all objects in the intersection, and increase their font size
- The system implements an Undo/Redo mechanism

3. Dynamic Models

In this section, a class diagram and a few sequence diagrams are presented in order to aid your understanding of how the software of the Venn Diagram Application works and how it was built.

3.1 Class Diagram

The class diagram below describes the structure of our system by showing the system's classes, attributes, methods, and the relationships with other classes. The class diagram for the Venn Diagram Application is displayed below (figure 1).

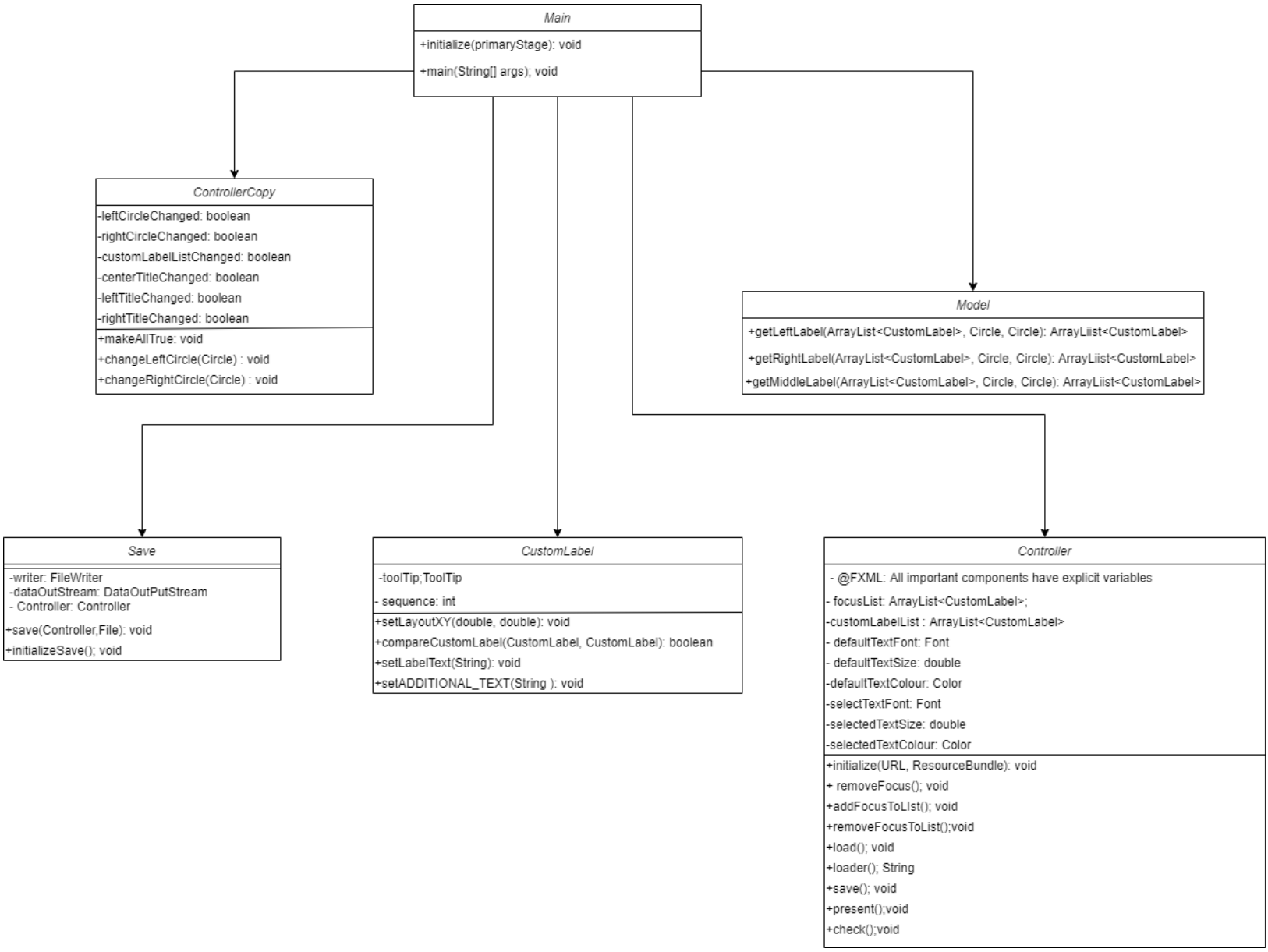


Figure 1: Class Diagram of Venn Diagram Application

Each class is displayed below in order to clearly see each method in the class. Below is the Main class (figure 2).

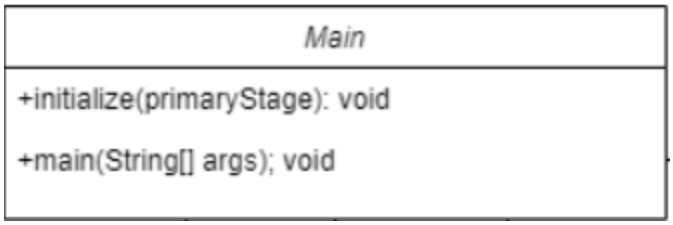


Figure 2: Main Class

The class displayed below is the ControllerCopy class which extends the Main class (figure 3).

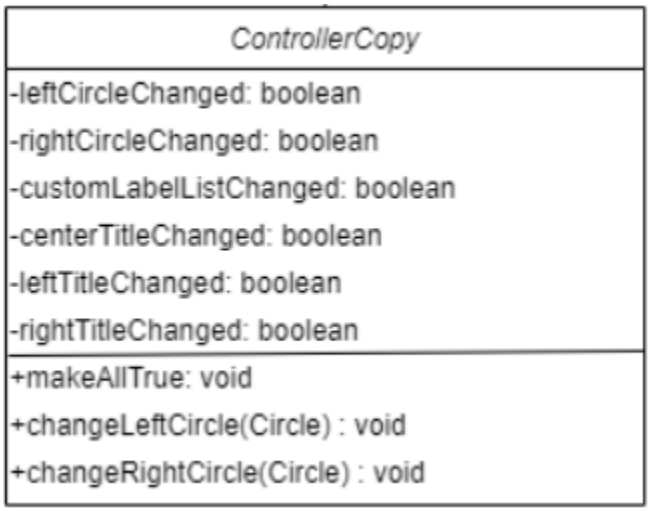


Figure 3: ControllerCopy Class

The class below is the Save class which extend the Main class (figure 4).

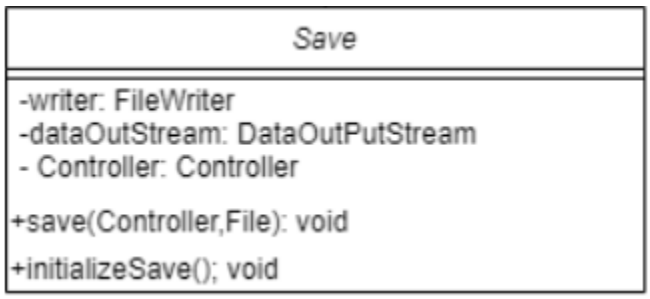


Figure 4: Save Class

The class displayed below is the CustomLabel class which extends the Main class (figure 5).

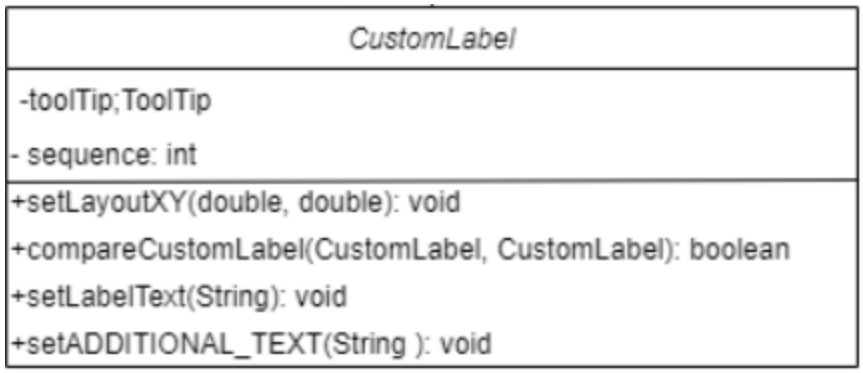


Figure 5: CustomLabel Class

The class that is displayed below is the Model class, which extends the Main class (figure 6).

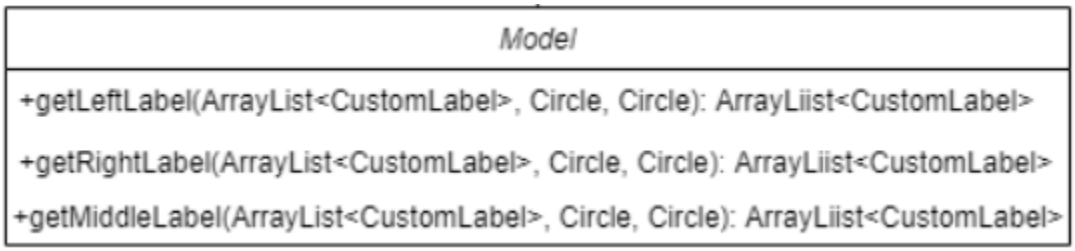
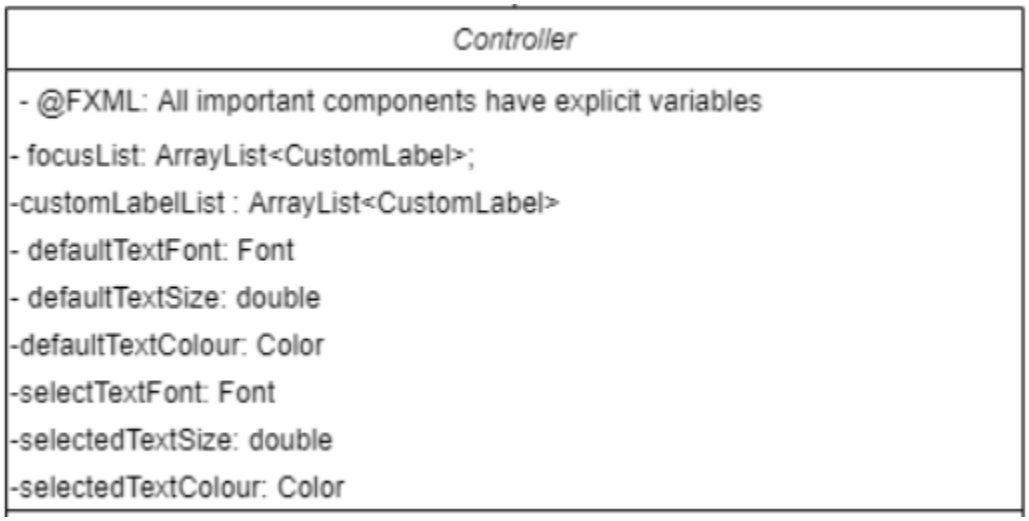


Figure 6: Model Class

The class displayed below is the Controller class, which extends the Main class (figure 7).



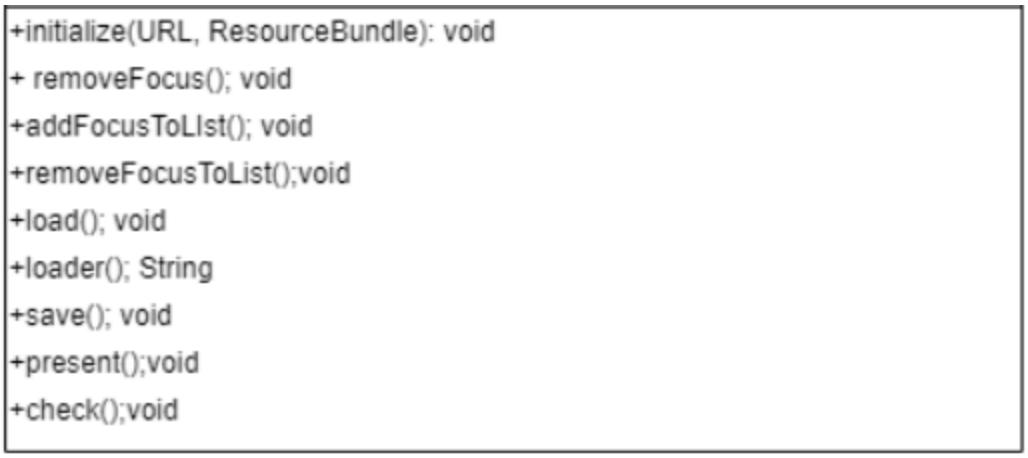


Figure 7: Controller Class

3.2 Sequence Diagrams

The sequence diagrams depict the interaction between objects in a sequential order. The following sequence diagram below, is used to portray the interactions between the User, UI Interface, Controller class, and Model class when the user interacts with the Design Tab to align a label (figure 8).

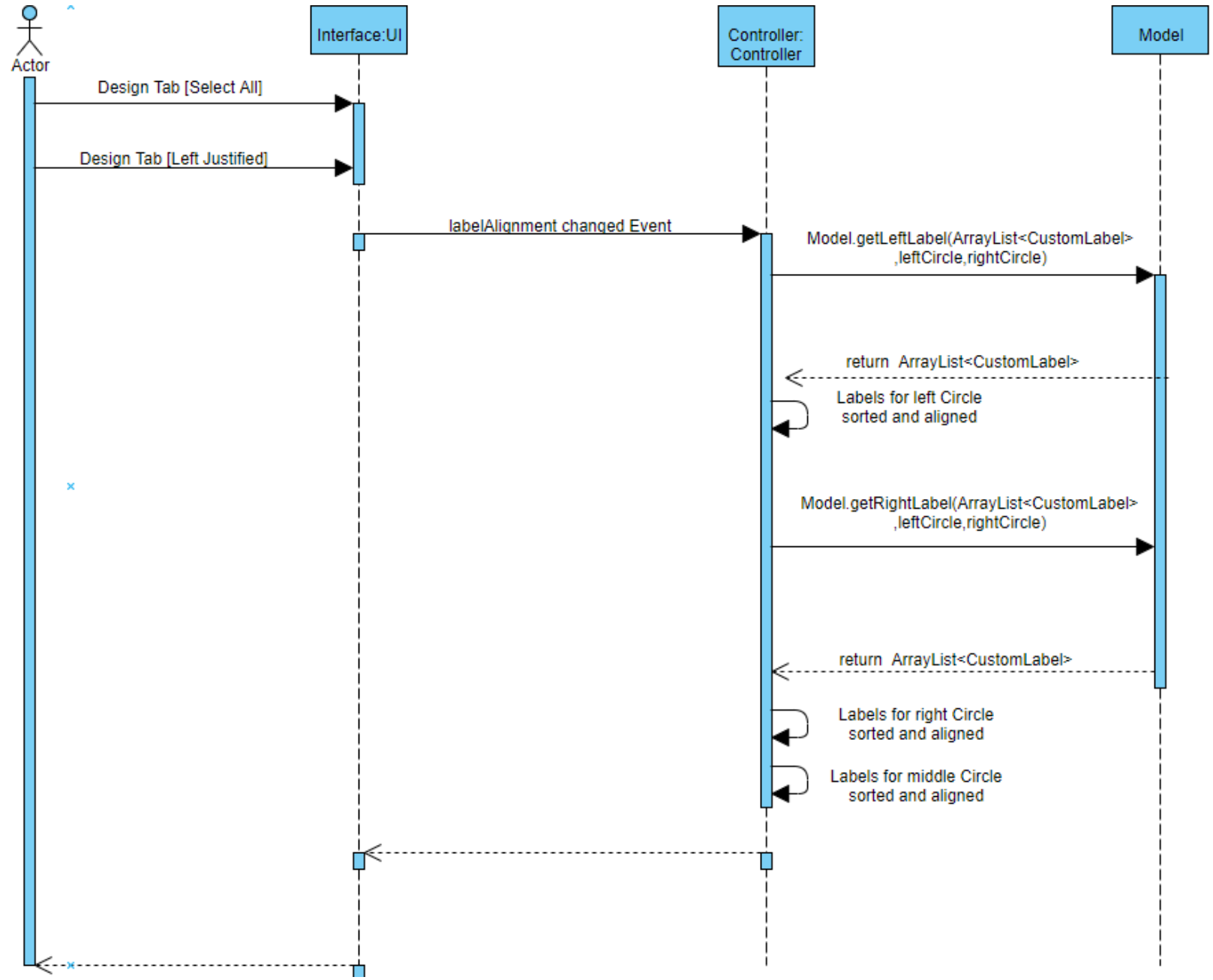


Figure 8: Sequence Diagram for Label Alignment

The next sequence diagram, which is below, shows the relationship between the User, UI Interface, and Controller class when the user changes the colour of a text selected or multiple texts (figure 9).

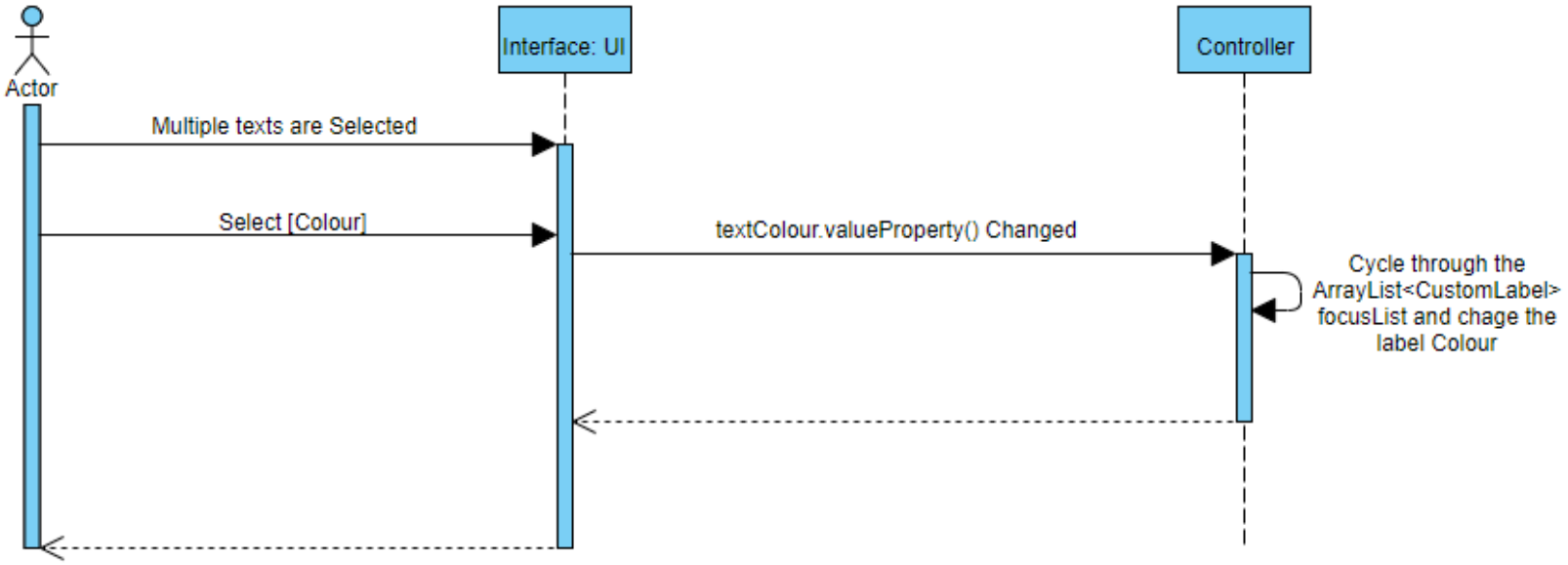


Figure 9: Sequence Diagram for Text Colour Change

The sequence diagram below describes the relationship between the User, UI Interface, and Controller class when the user creates a label (figure 10).

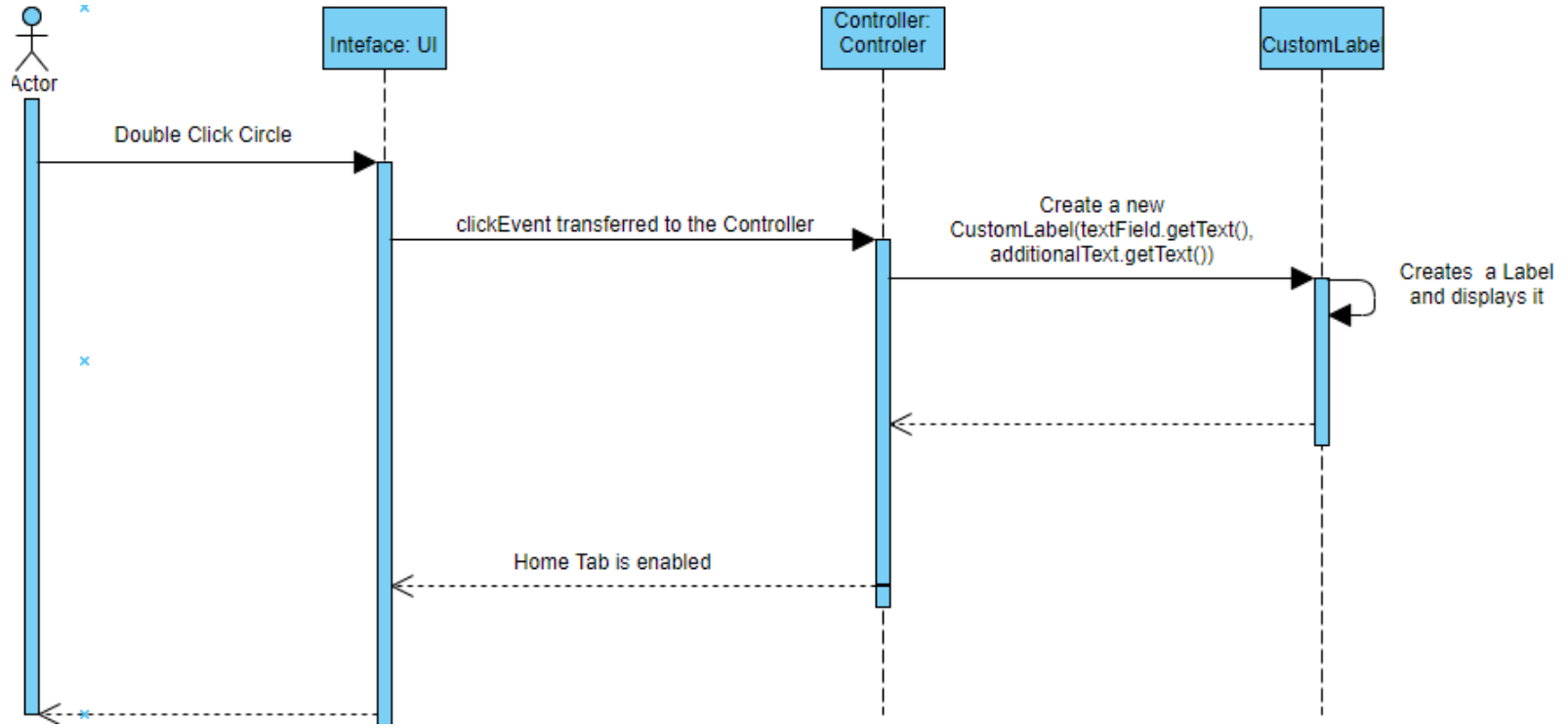


Figure 10: Sequence Diagram for Adding a Label

The sequence diagram below describes the relationship between the User, UI Interface, and Controller class when saving the Venn diagram application (figure 11).

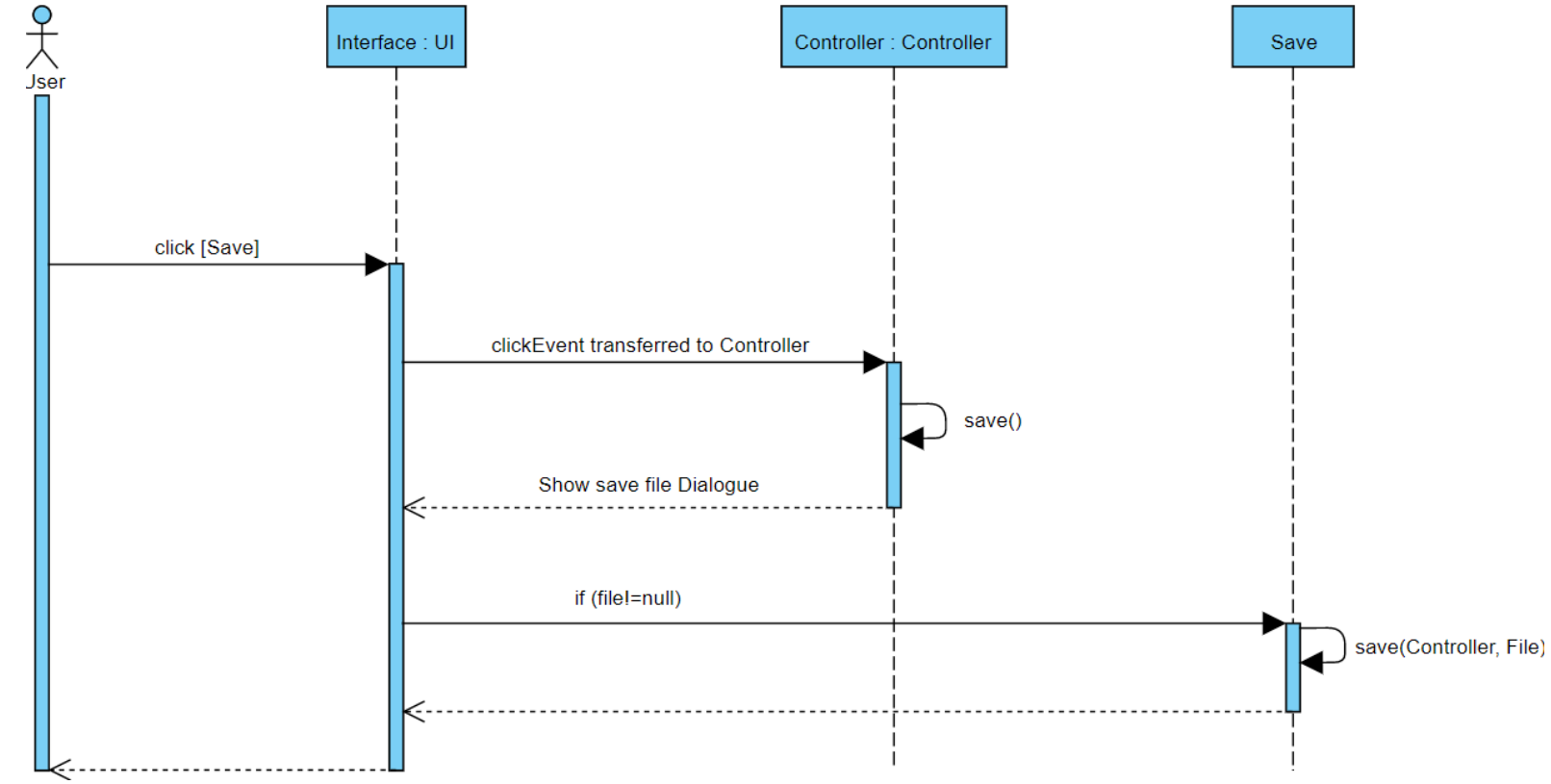


Figure 11: Sequence Diagram for Saving the Venn Diagram Application File

The sequence diagram below shows the relationship the User, UI Interface, and Controller class when loading the Venn diagram application (figure 12).

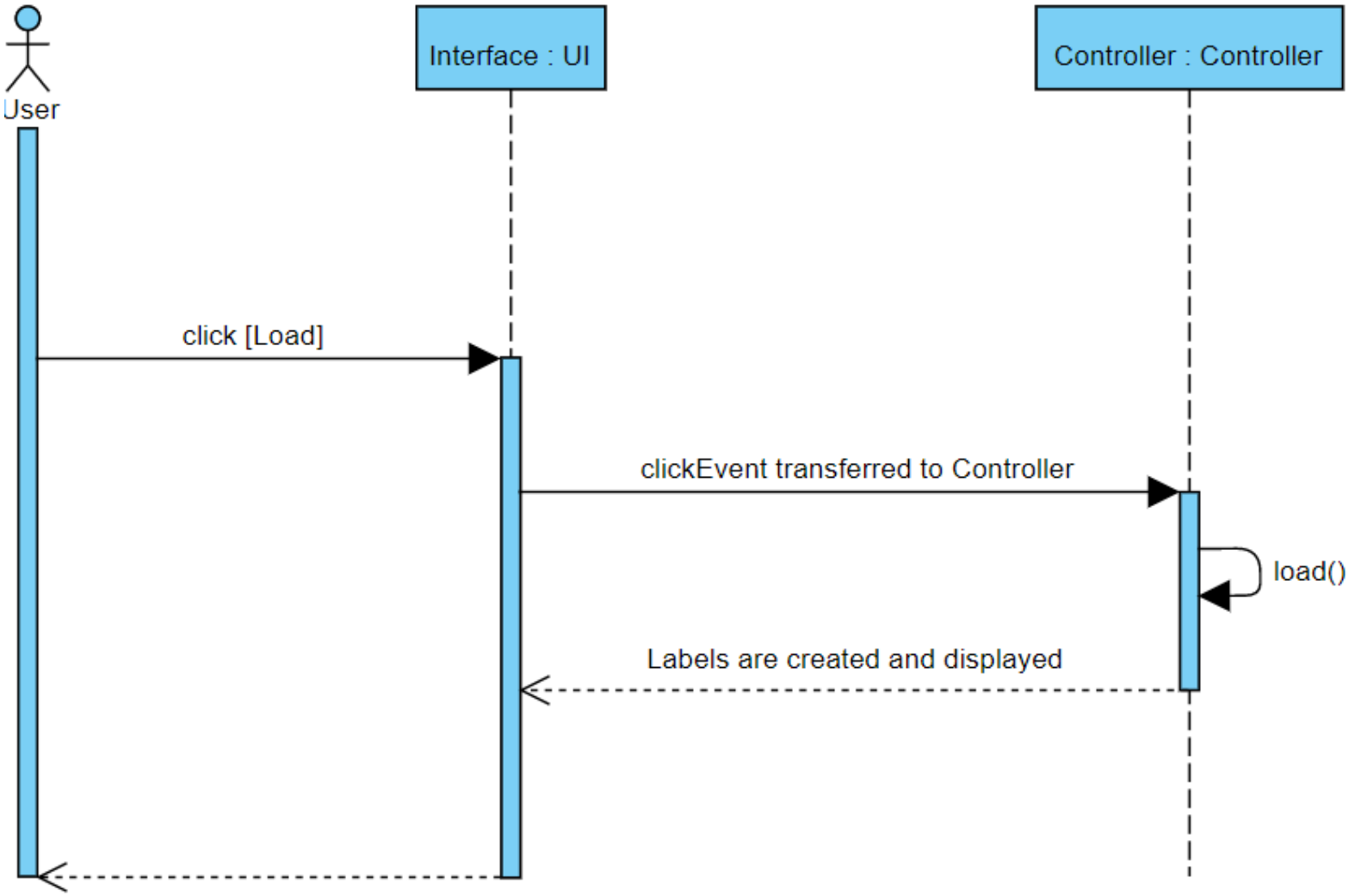


Figure 12: Sequence Diagram for Loading the Venn Diagram Application file

4. Maintenance Scenarios

In this section, the following Maintenance Scenarios will be discussed,

- Swap Texts
- Auto Scaling Circles
- Animated Texts
- Data Mode
- Extra Sets

What must be done to allow users to swap texts?

To swap texts, we can create a method that while the text is being dragged if more than 65% of the currently dragged text is on top of another text, it will cause the two texts to swap. This also means that the method must save the original position of the text being dragged.

Is it possible to auto scale the circles and its respective circles to degree of proficiency where the user does not see the need to change their positions?

The scenario would be around the alignment design tab. Aligning the texts should cause the circle selected to scale based on the size of texts it encompasses. Auto scaling circles would also mean that there will always be an intersection between the circles capable of storing at least one text. To do this, we can create a method that calculates that maximum height and width the texts takes up and resize the corresponding circle based on the output of that method, while also keeping the other circles within intersection. Meaning, we would also need to create a method that also causes the position of the circle to change based on their size, allowing the circles to always have an intersection point.

Presentation quality is often impacted by animations. How would a user be able to change the animation of a text? Or any other part of the diagram?

The scenario would cause the texts to appear based on the sequence the user defined it to be. And gives the user different animations for the text to be displayed. To achieve this, it is possible that we need to make a new class system for animations and sequences. Importing animations from JavaFX and redesigning the design and home tab to allow the user to change the sequencing of the texts. When it comes to sequencing, we can allow the user to have customize it based on the follow: order added, alphabetical order, or custom.

Venn Diagrams themselves compare and check what the differences and similarities are between two sets. Where the two sets are manually added. Is it possible to detect if all the sets are numbered values and from there, create the diagram such that the user no longer needs to manually input each value but read from a file and place the inputs in their respective places?

This scenario would be based on collecting data from two or more data sets. The user would set each circle to encompass a set defined by the user either from their input, or an import of values. The program will then find the intersections of the data sets, displaying them in their respective intersections. To do this, a new prompt must be created when the program is launched, prompting the user for the option of "Data Mode" or "Normal". The methods created for the program to work would revolve around finding values that is contained between all the sets, meaning we would need to create a deep copy of the input set, sort the data by quicksort, and then find the values that are present in both sets.

What does one need to change to support extra sets in the Venn diagram?

In order to support extra sets, the number of circles must increase to the number of sets placed. The model class must change in accordance to the number of sets,

currently, it is only possible to do calculation using two circles, thus, the model class must be rewritten in such a way that it changes its own code with respect to the number of sets. The saving and loading algorithm must also be tweaked to allow for the saving and loading of the nth subtitle, where n is the number of circles and n must be greater than or equal to 2. The design tab must also be changed to allow the user to change the nth circle's size, colour, and alignment. The autoscaling algorithm must also change with respect to the number of circles in the application.

5. Appendix A: Record of Changes

Version Number	Date	Author/Owner	Description of Change
1.0.0	*01/26/2020*	*Group 12*	*1st Version of Product*
1.1.0	*02/09/2020*	*Group 12*	*1st Version of Prototype*
1.2.0	*02/23/2020*	*Group 12*	*2nd Version of Prototype*
2.0.0	*03/08/2020*	*Group 12*	*Re-design and Functionality Upgrade*
2.0.1	*03/22/2020*	*Group 12*	*Some Bug Fixes*
3.0.0	*04/15/2020*	*Group 12*	*Final Version of Product*

Table 1 - Record of Changes