

EECS 2311 W20

Venn Diagram

Design Document

GRP - #11

MEMBERS

Tianpei Liao

Arda Temel

Lynn Al Agilly

Jeyaprashanth Sivasubramaniam

York University

Table of Contents

Introduction	3
System Functions	3
Model Classes	4
Controller Classes.....	5
System Class Diagram.....	6
Sequence Diagrams.....	7
Maintenance Scenarios.....	12
Appendix: Complete System Class Diagram	13

Introduction

This design document is one of four documents that serve as draft deliverables for the Venn Diagram application. It contains the overall design hierarchy of the system through the class diagram, and portray system flow and class interaction through the sequence diagram. We highlight.

The UML diagrams show the heavy reliance on components with their core functionalities. We only focus on functions and their interaction between each other up to current development. Details about individual classes, their respective description, and roles of classes are explained detail in the Testing Document. This document provides complete interaction between classes and details.

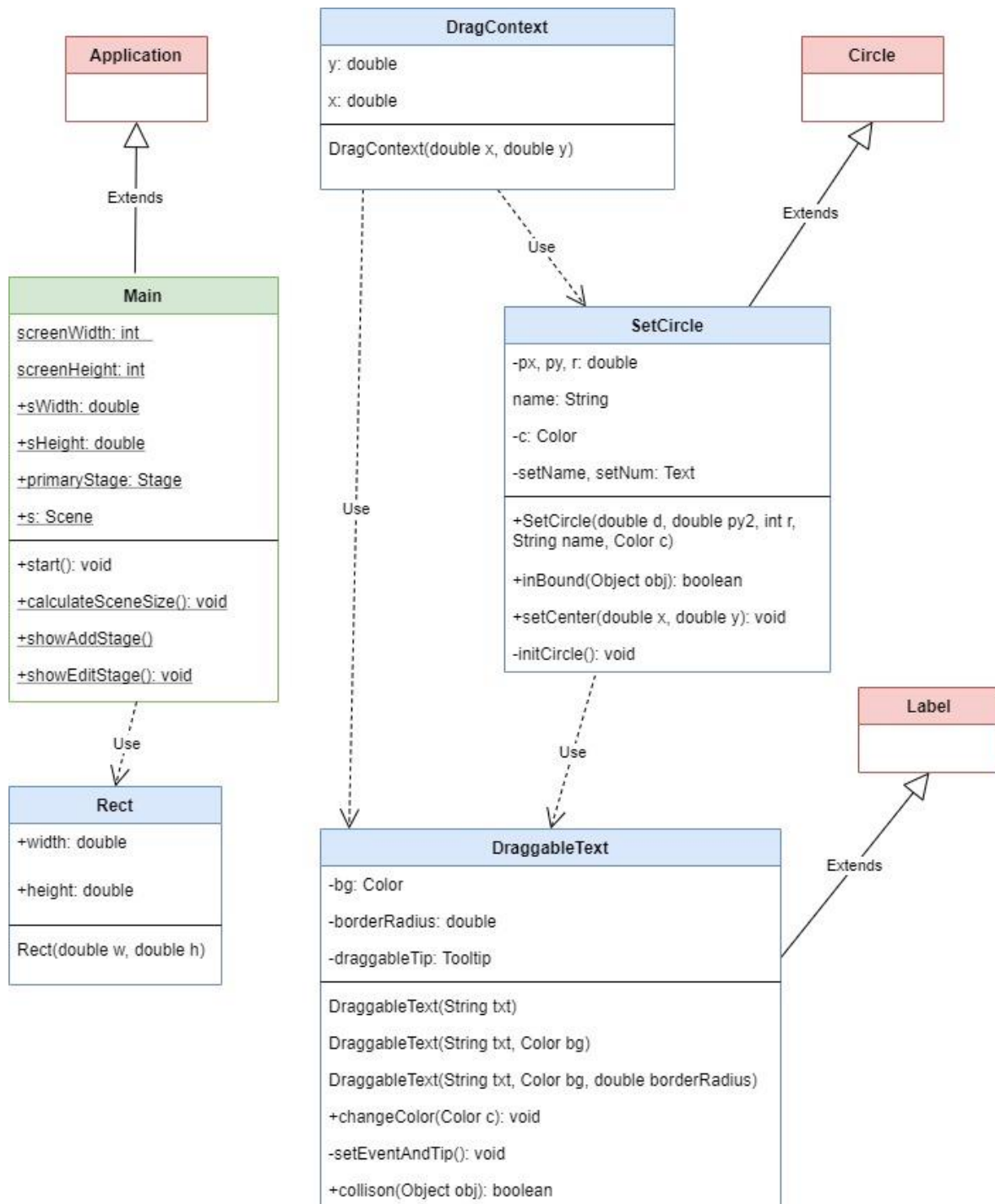
System Functions

An overview of the main functionality of the application. Following functions are completely implemented and fully functioning.

- Add, Edit, Delete entry labels
- Import multiple labels from text file
- Export current system state as image file (.png format)
- Drag and drop one or more labels across the application window
- Entry categorization – drag drop the entries within the Venn diagram visually
- Delete more multiple entries at once
- Save current system state as text file
- Load existing saved file to update
- Import answer sets and start Test mode
- Submit answers for evaluation in Test mode
- Delete answer existing answer sets and exit from the Test mode
- Undo and redo (This function only works for delete operation)
- Adding additional answer sets

Model Classes

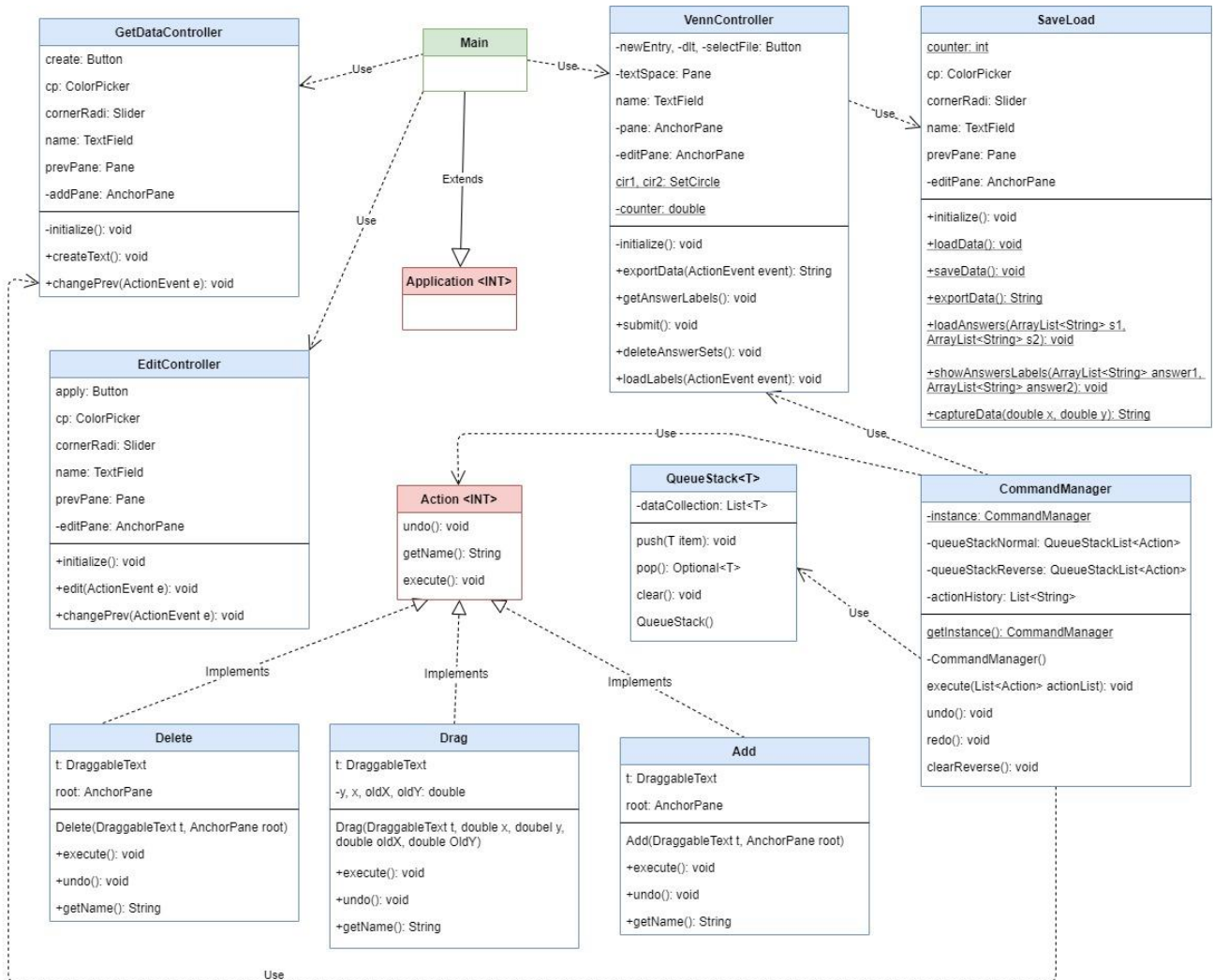
The application's main starting point is the start method in Main.java class, which is responsible for displaying the main GUI and using events to call other methods and pages. The following diagram represents the systems model classes and their relationship with other core classes.



Controller Classes

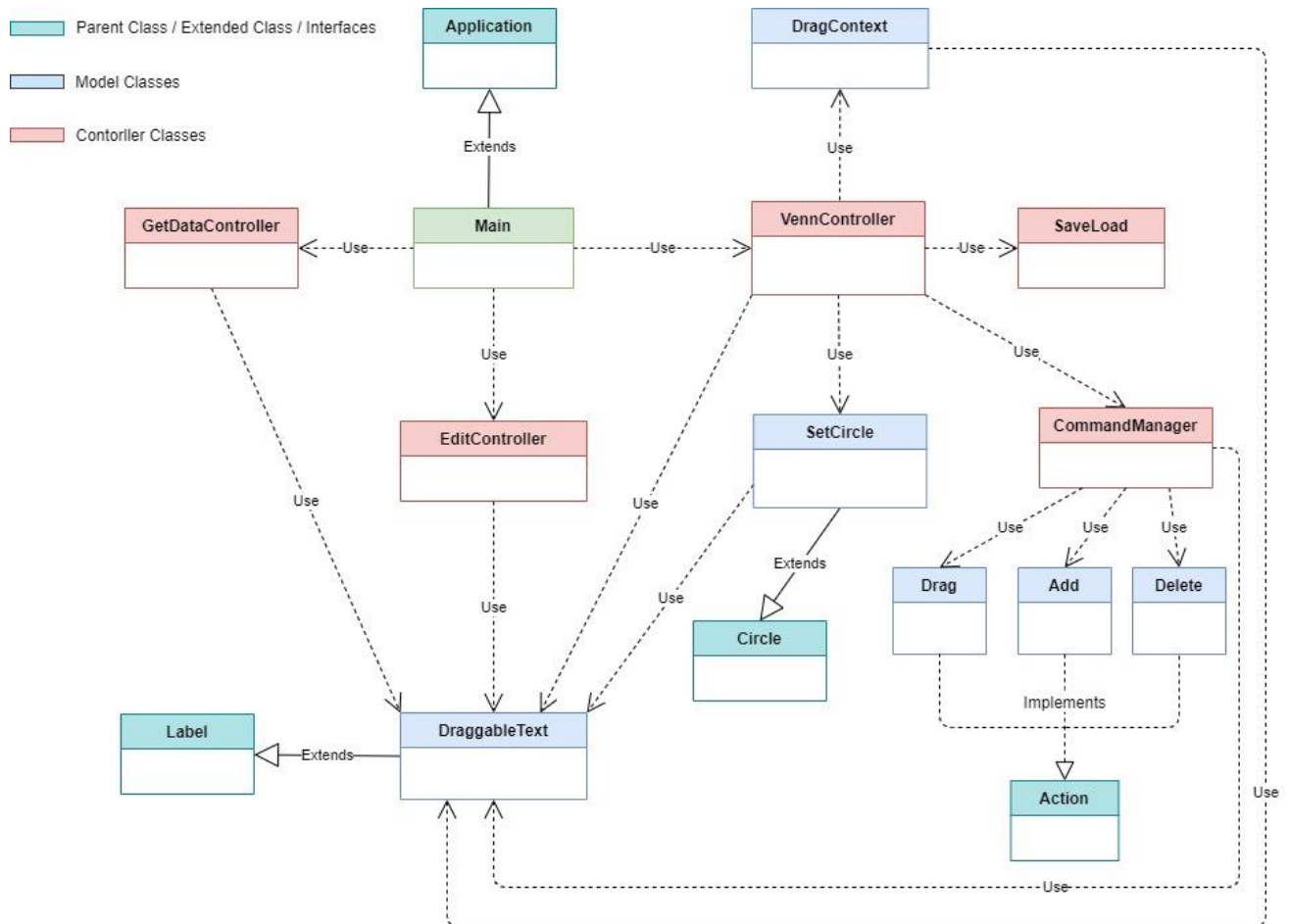
VennController.java is the main controller class in this system to interact with the main display GUI and call other methods and class functions. All user interaction with main GUI handle by this VennController class.

Following UML diagram shows all Controller classes and their interactions in the system.



System Class Diagram

Following diagram illustrate full system's view of class diagram and how the system classes interact with each other. **(without attributes and methods information)**

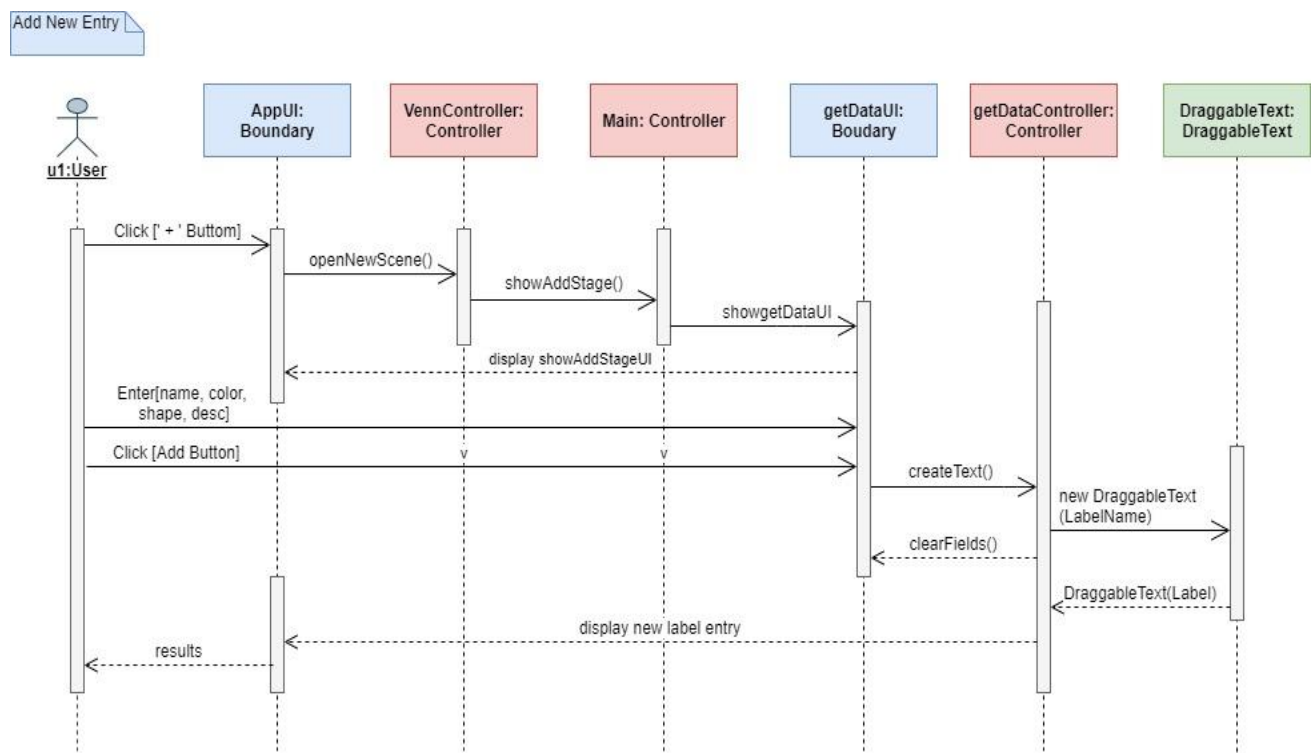


Sequence Diagrams

Add New Entry

Following diagram shows complete sequence diagram for adding new entry. User can add more than one label until close the Enter Data window.

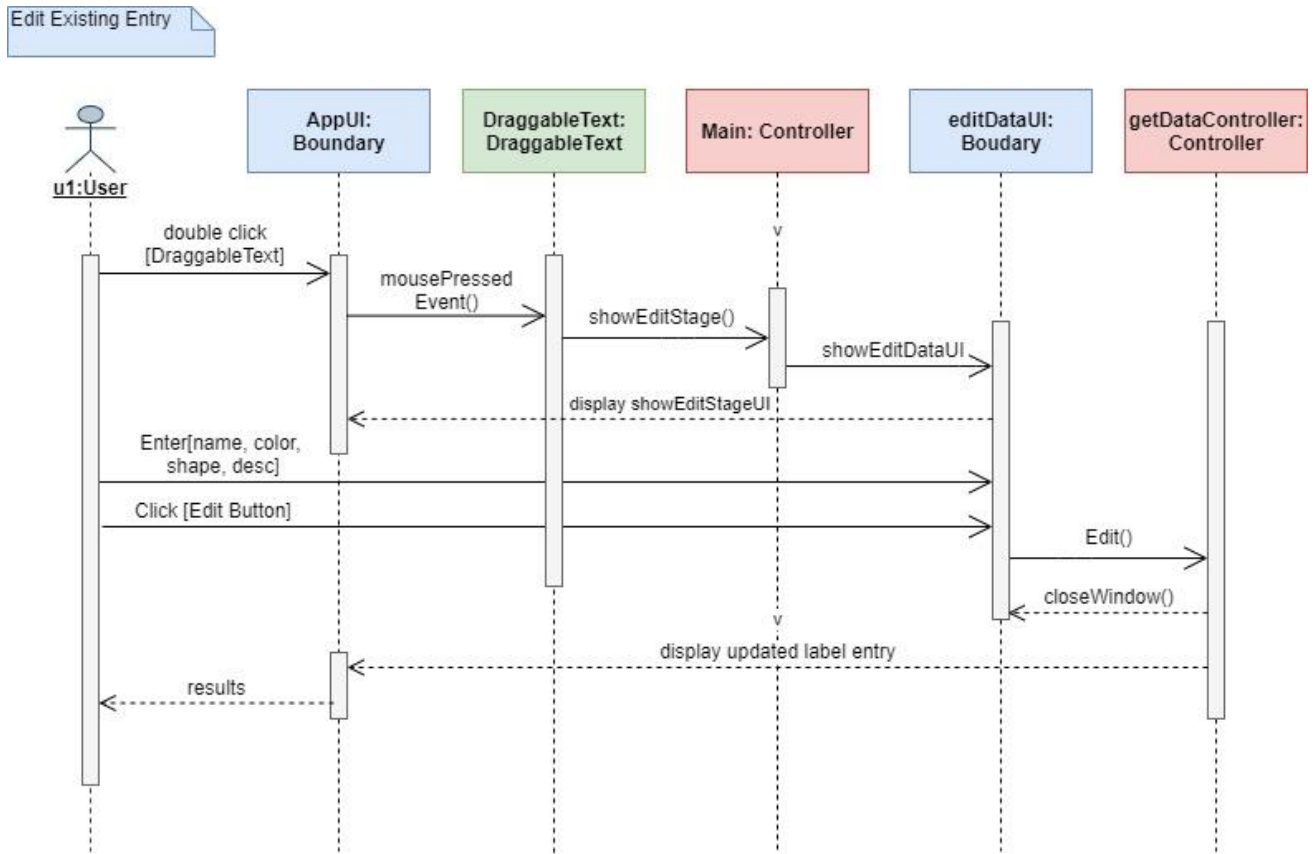
User can add more than one entry using enter key button click. Every time user adds new entry the getDataUI window fields (name, colour, description, etc.) will be cleared.



Edit Existing Data

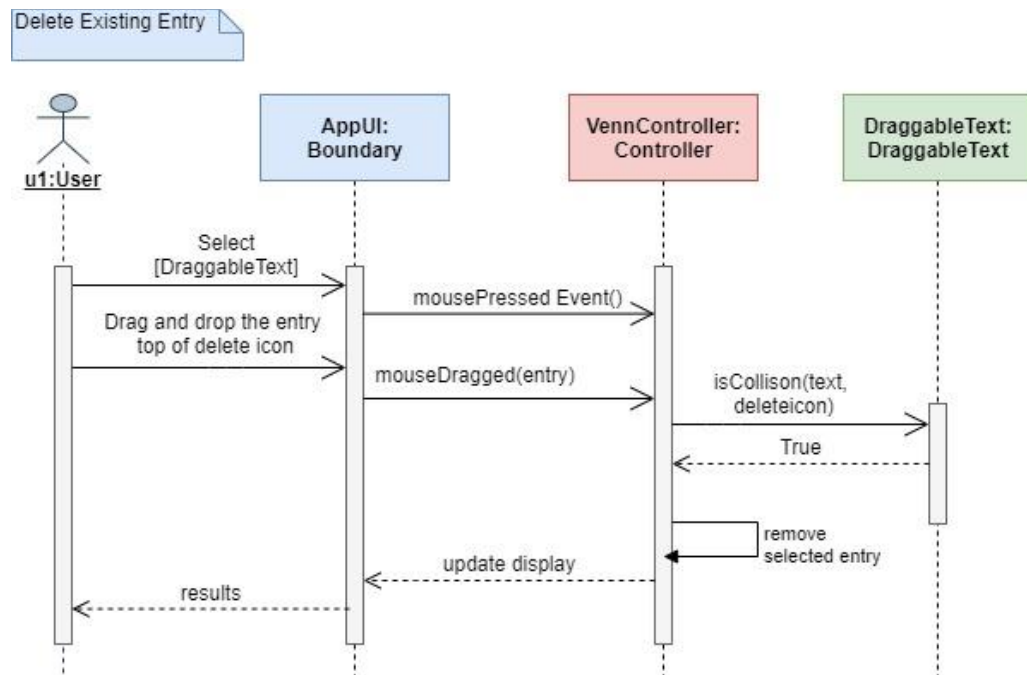
Following diagram shows complete sequence diagram for edit existing entry properties. User can edit once at a time once complete the update the edit stage window will be close itself.

System supports edit function to one label at a time because entries have multiple attributes.

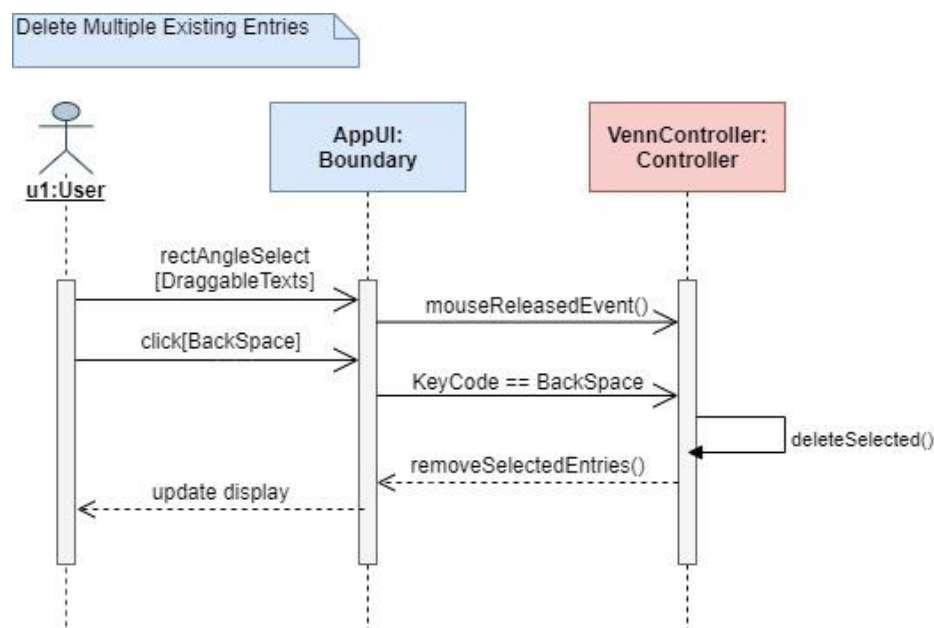


Delete Entry

Following diagram shows complete sequence diagram for delete existing entry using drag and drop method.

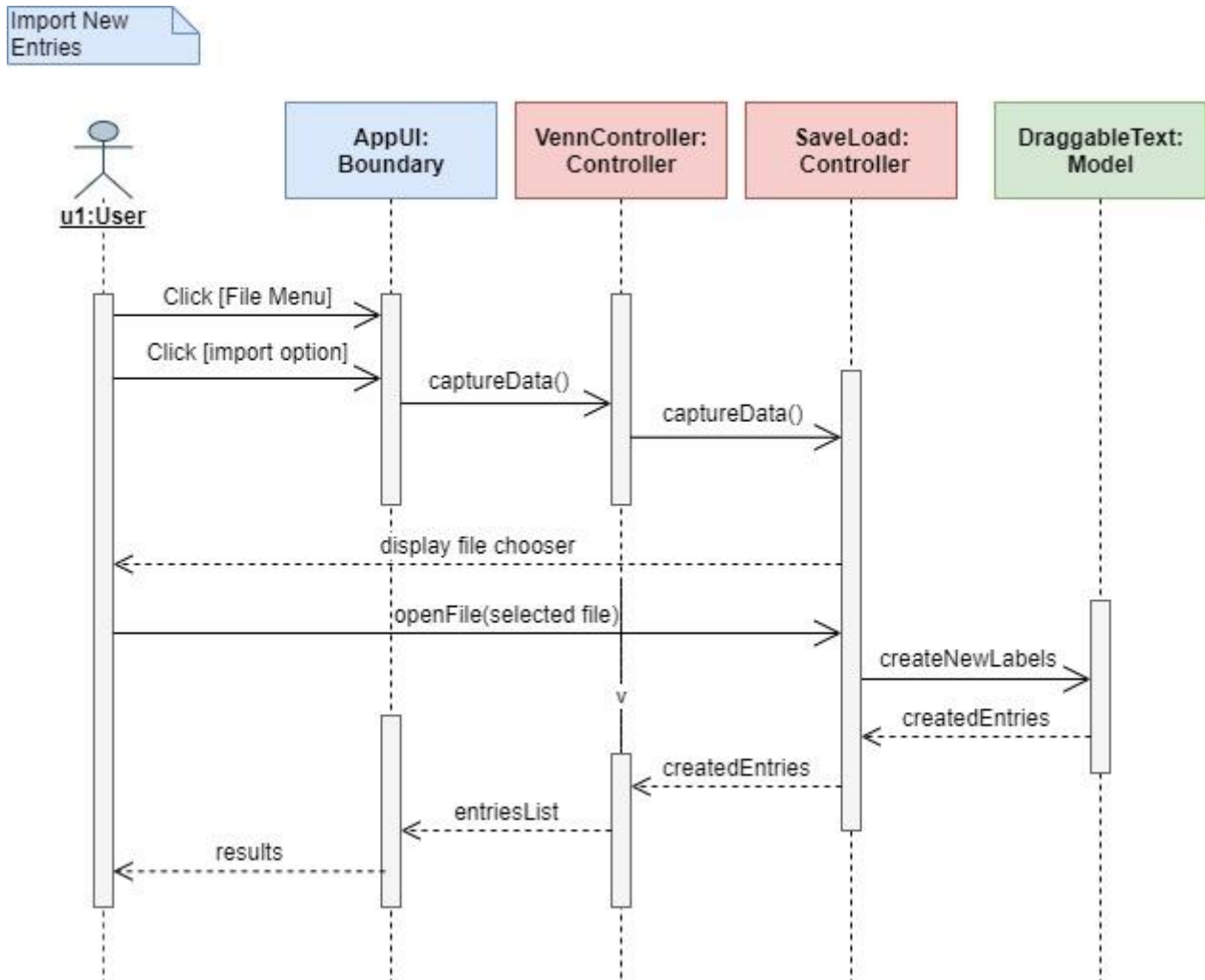


User can delete multiple entries by select multiple entries and press the backspace key.



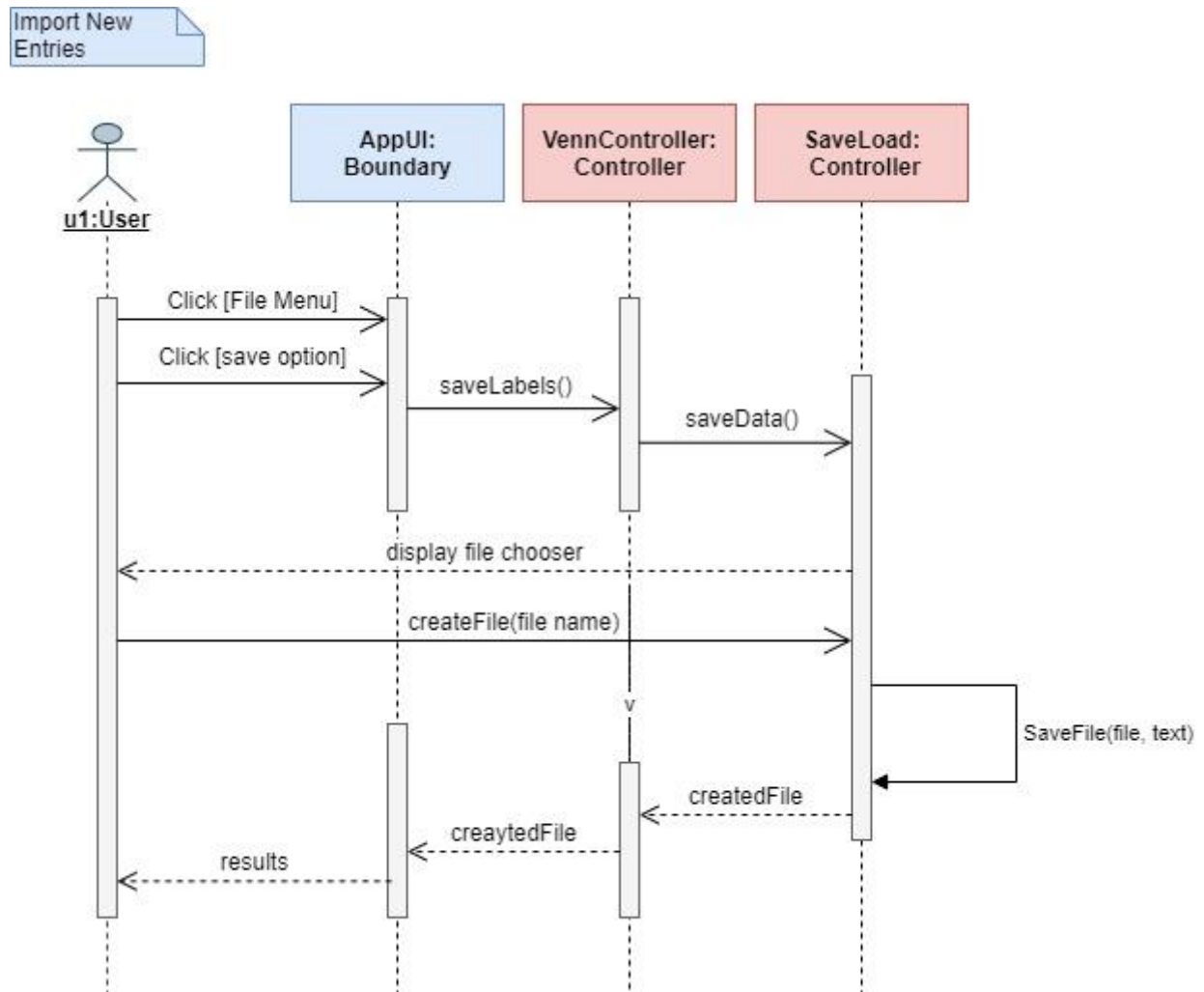
Import multiple entries

Following diagram shows complete sequence diagram for import multiple labels from text file.








Save File

Following diagram shows complete sequence diagram for save current state of the system in text file format.



Maintenance Scenarios

In this section we described the features which are extended version of the application. Following features will be implemented to improve the functionality of the product.

-  **Customize label font size,** User can customize the new entry label font size when adding or updating labels properties. DraggableText class model class maintains the properties of the label so include font as one of attribute and create font size selector as drop-down menu in UI (getData.fxml, editData.fxml). update the create () method in GetDataController class and edit () method in EditDataController class.
-  **Add additional circle,** SetCircle model class has implementation of the circle. Create another SetCircle static object and implement the colour, size, radius and position in VennController class initialize () method and implement other related events. Or implement this as additional feature to user through button click event.
-  **Auto scaling circles,** user can increase or decrease radius of the circles according to the label's size or number of labels that circle contains. To do that we can create a method for auto scaling in VennController class and also implement another method in SetCircle model class to maintain the properties of the circles because SetCircle class is the one which has implementation for circle properties. We have to set the maximum limit and center point for circle location to maintain the intersection of the circles.
-  **Colour changeable circles,** user able to change the circle colour like label colour. Diagram circles have default colour when we load the application and which is already set in SetCircle method in SetCircle model class. We already have method setNameEvents () to create title for each circle separately in SetCircle model class. When user double clicks the circle title a window will open for title update. So, we can additionally add colour picker in that window and allow user to update the colour of the circles and capture the new colour and title update.
-  **Fully functional Undo and Redo,** user able to undo and redo their operations completely. Current system supports this feature only for Add, Delete, Drag and Import / Load. System already have CommandManager class to save each user commands and maintain the action history. So, we only need to create some other model classes for remaining functions same as current model classes such as Delete, Add and Drag. And implement method call in VennController class.

Appendix: Complete System Class Diagram

