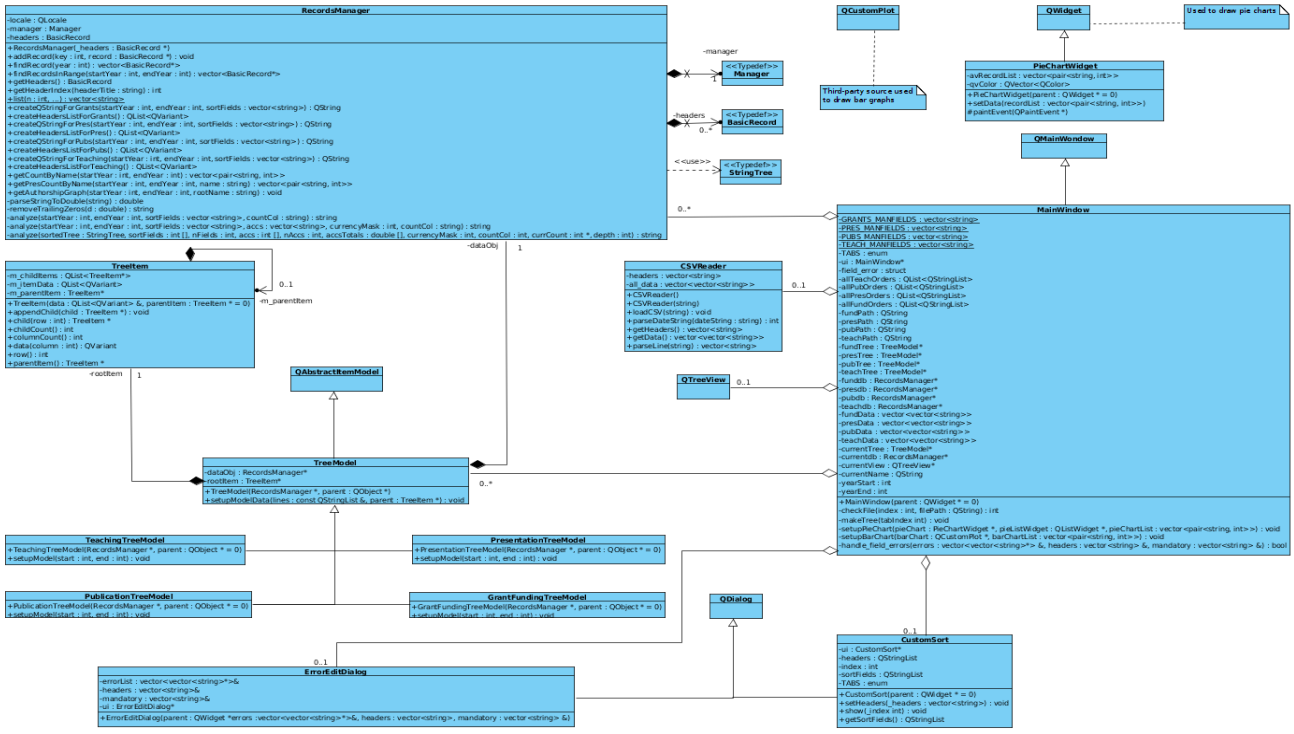
CS3307A – Acuity STAR Project – Team Peach Stage 2 – Class Diagram



Above is the class diagram describing the types of system objects and their static relationships for the Acuity STAR project application. Each class is divided into three components: the name of the class (in bold), its attributes, and its operations. There are two kinds of relationships between classes used in our diagram: associations, denoted by a line beginning at the source class and ending with a diamond at the target class, and generalizations, denoted by a line beginning at the source class and ending with a triangle at the target class. Associations are read "<target class> has a <source class>", while generalizations are read "<source class> is a <target class>". The clear diamonds indicate shared associations while the black diamonds represent composite associations. In addition, each relationship a multiplicity indicating how many objects may fill the property. Below is a list of the classes and their relationships.

CSVReader

CSVReader is used in MainWindow to read and parse the data from the CSV file. This class is designed to meet the requirement of loading of data from a CSV file and perform error processing.

RecordsManager

RecordsManager is used in MainWindow to create records from a CSV file for the various

summary types. These records are the bulk of the data manipulated by the system to create the required dashboard and visualizations. It is also used by TreeModel to sort the data and build the appropriate dashboard view.

TreeItem

TreeItem is used in TreeModel to store the records for the dashboard summary. Each TreeItem has a parent except for the rootItem, forming a linked list which facilitates and minimizes record storage time.

TreeModel

TreeModel is used in MainWindow to create and meet the dashboard summary requirement. TreeModel has a TreeItem which acts as a pointer to the first record in the list. It also has a RecordsManagers which it uses to build the data structure from the unsorted loaded data and passes on to the graphical user interface components. It is a generalization of the GrantFundingTreeModel, PresentationTreeModel, PublicationTreeModel, and TeachingTreeModel classes and is also a subclass of QabstractItemModel.

MainWindow

MainWindow contains the user interface and main program functionality for loading, filtering, and presenting data through the dashboard and other types of visualization. MainWindow is a subclass of QMainWindow, which allows it to access a vast amount of QT library features, standardizing and simplifying the user interface and visualizations implementation.

PieChartWidget

PieChartWidget created during the execution of MainWindow; it is its own temporary stand-alone class for visualizing the data kept in the records inside the TreeModel. PieChartWidget is a subclass of QWidget so that it may take full advantage of the QT library features and functionality.

CustomSort

CustomSort is also created during the execution of MainWindow, representing a dialog for the user to customize and filter the data they wish to select for visualization. CustomSort is a sublass of QDialog so as to facilitate integration with the other Q-type graphical user interface components.

ErrorEditDialog

ErrorEditDialog is also a subclass of QDialog and is created during the execution of MainWindow. It allows the user to either discard or edit missing mandatory fields to the loaded data.

QCustomPlot

QcustomPlot is a third-party source class used to plot bar graphs of the loaded data for various dashboard views. It is a stand-alone library and is built to be integrated with QT. It is created when MainWindow is executed once the desired data is loaded from the file.