Deliverable 3

Table of Contents

Requirements Implemented

Mandatory requirement 2（Add Line Charts）

ID 02

Description

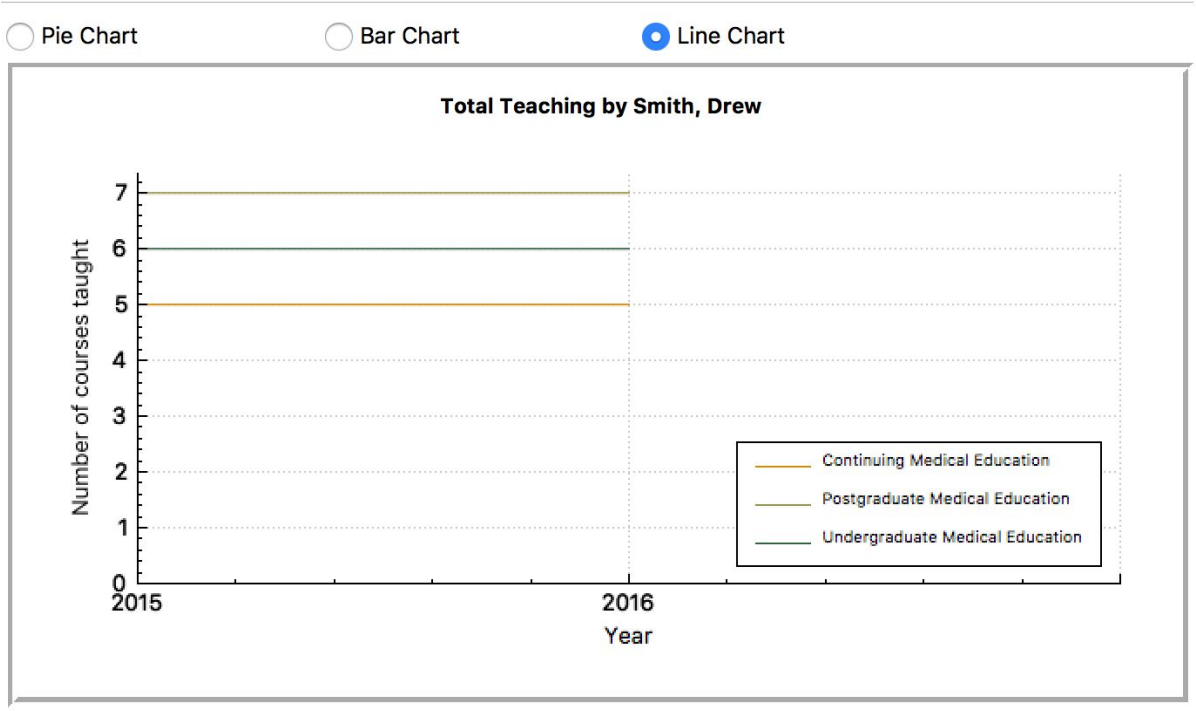
We implemented a line chart representation of data to the reporting options (Mandatory Requirement). In Phase I of the project, Team Peach implemented two different kinds of visualization for the data: a pie chart and a bar chart. In Phase II, we have added another option for users: a line chart. We added GUI elements in the same style as Phase I uses allowing the user to check another box to change their view. This required learning how to make user of the QT Designer to add widgets and bind functionality to the 'slots' of GUI elements. We added a method to the 'MainWindow' class that parses the list representation of the data to render it in the correct visual form.

Origin

Implemented by Chris and Yuchen of Team Orion

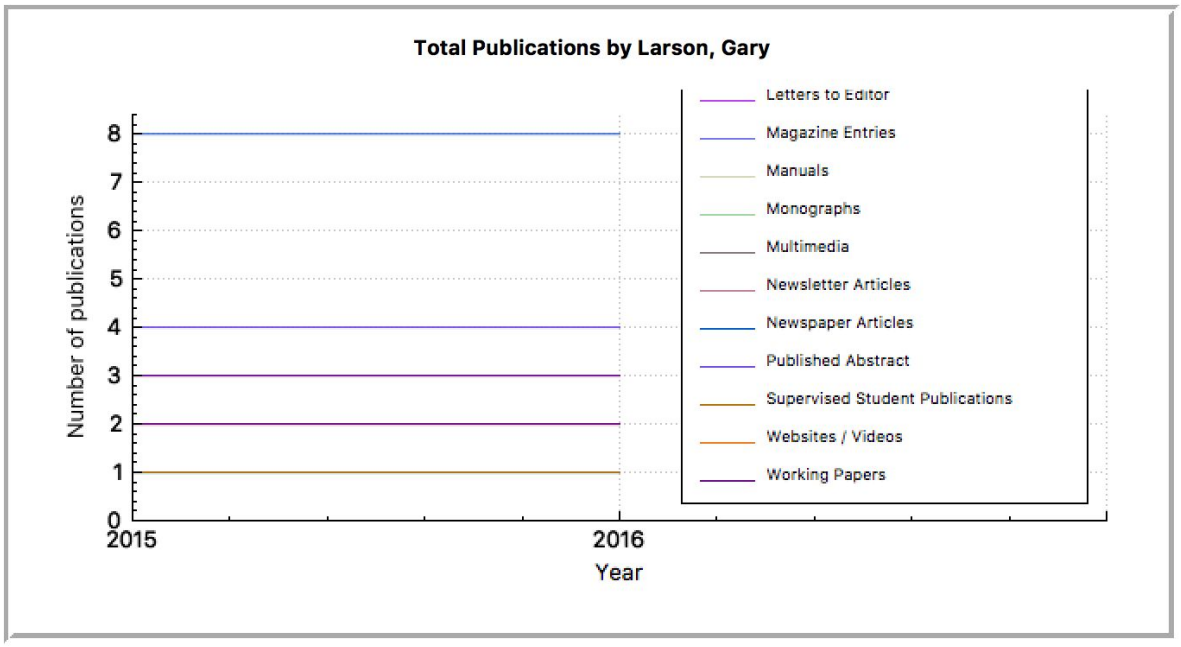
Examples

Line Chart example for Teaching



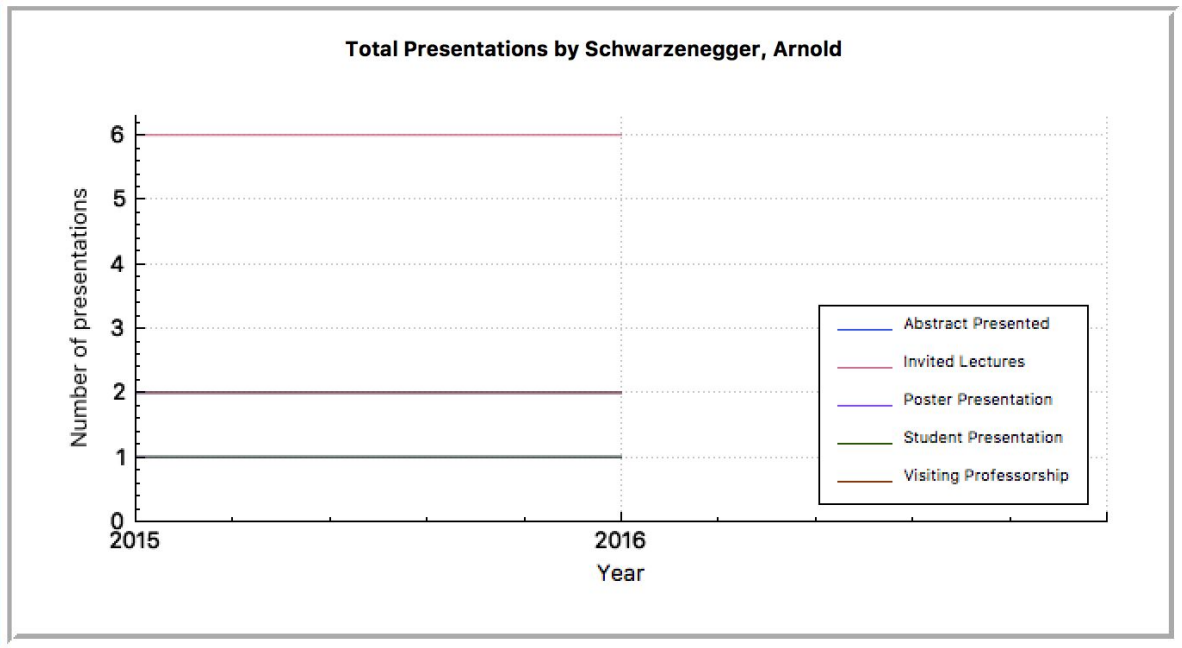
Line Chart example for Publications

Pie Chart Bar Chart Line Chart



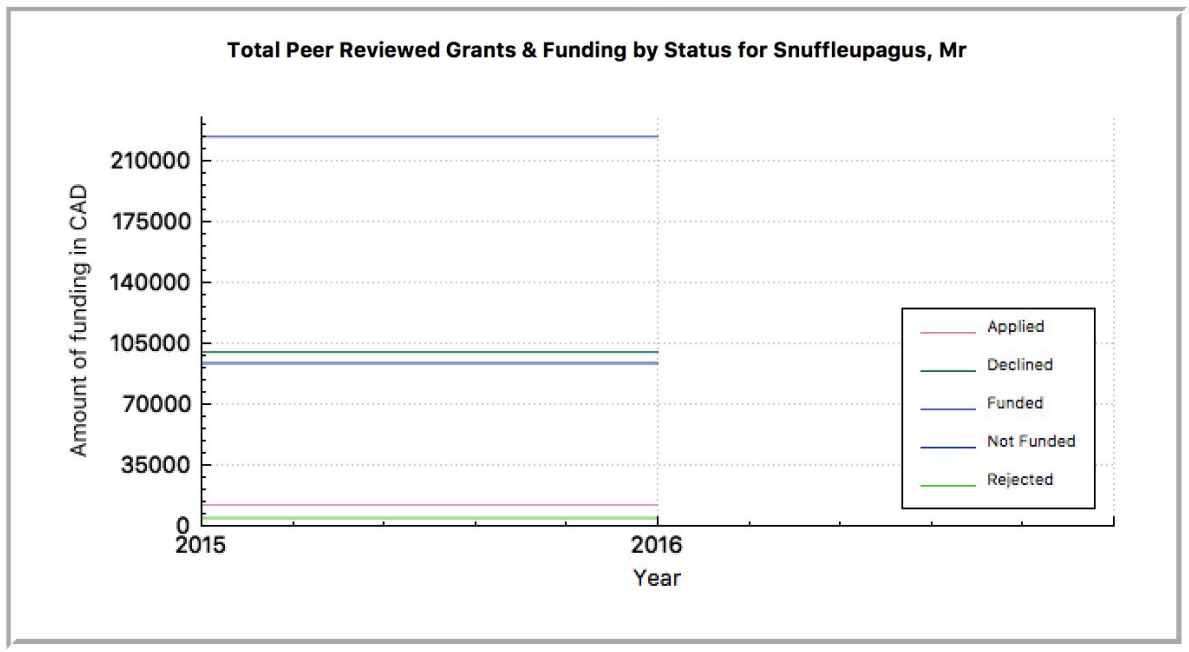
Line chart example for Presentations

Pie Chart Bar Chart Line Chart



Line Chart example for Grants and Clinical Funding

Pie Chart Bar Chart Line Chart



Mandatory Requirement 3 (Add Save-Session)

ID 03

Description:

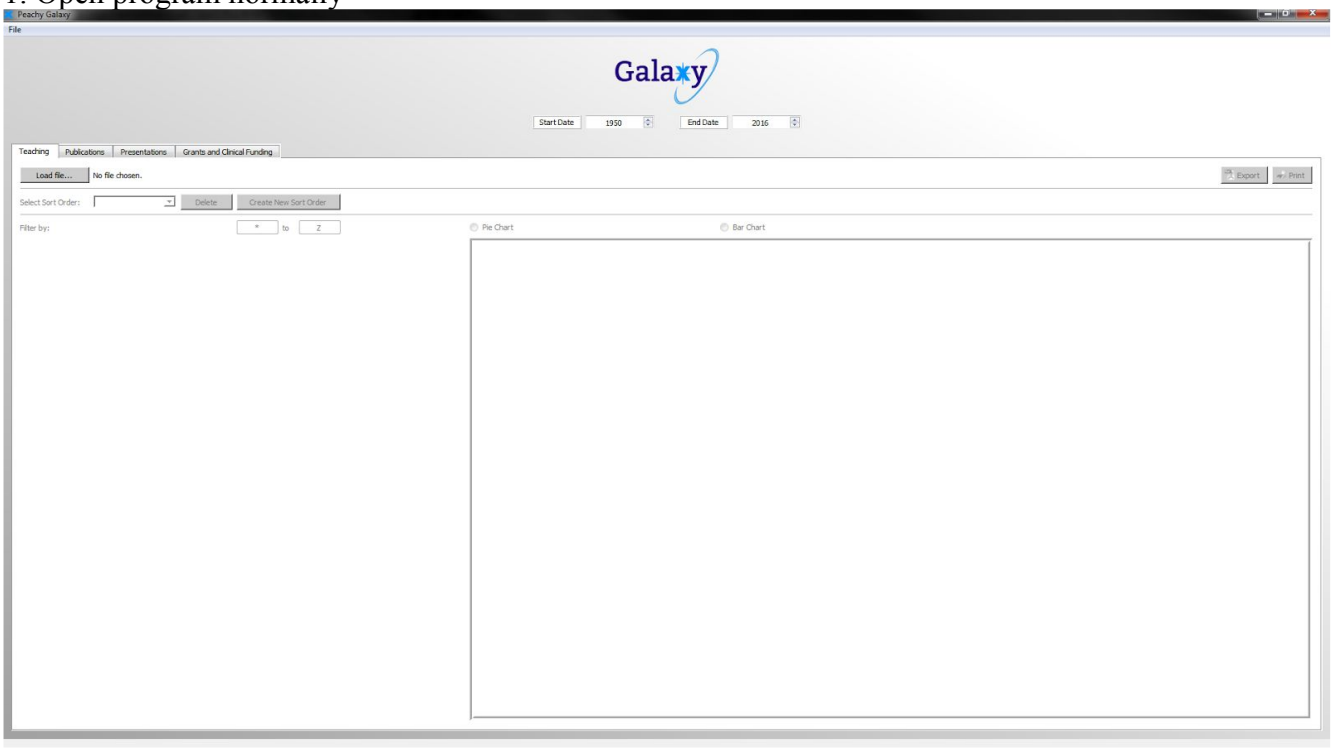
One of the requirements specified by the client is for the program to have the ability to save the current session state so the user can resume what they were last doing with it. Currently we have implemented methods that remember which files have been loaded during the previous session. We were able to retool the class QSortListIO from Phase 1 into QFileIO which accepts a single string containing the path to a file that was loaded instead of a list of strings containing methods to sort its data. QFileIO takes a QString of the file path of each type of .csv loaded during a session and serializes it to a .dat file in the folder the program is running from, and is converted back into a QString in the mainwindow class when the program is re-opened. The mainwindow class searches for .dat files and loads the path stored inside if such files are found. In the event that no .dat file is located for a field (teaching, publications, etc.), the program assumes that there was no such file loaded during the previous session and leaves that field empty. Once the program locates an existing file it is loaded automatically and declines the error handling prompt which requests the user to fill in missing mandatory fields. For all files loaded manually, the prompt will still appear as usual. While this may not be the most ideal way of dealing with this situation, it is unclear whether the client wants any fields filled by the error handling prompt to persist between sessions and once this is clarified a more elegant solution will be implemented for the next prototype.

Origin

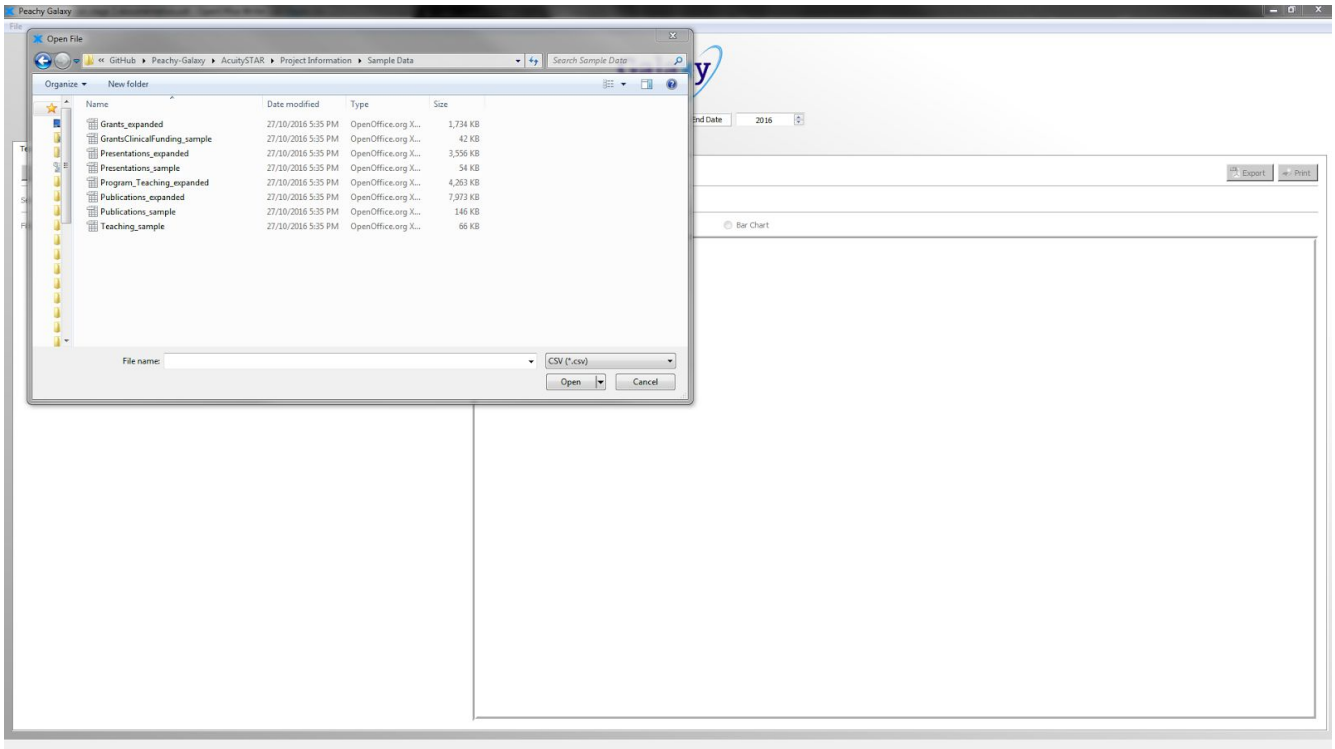
Implemented by Jeffrey and Paul of Team Orion

How it works:

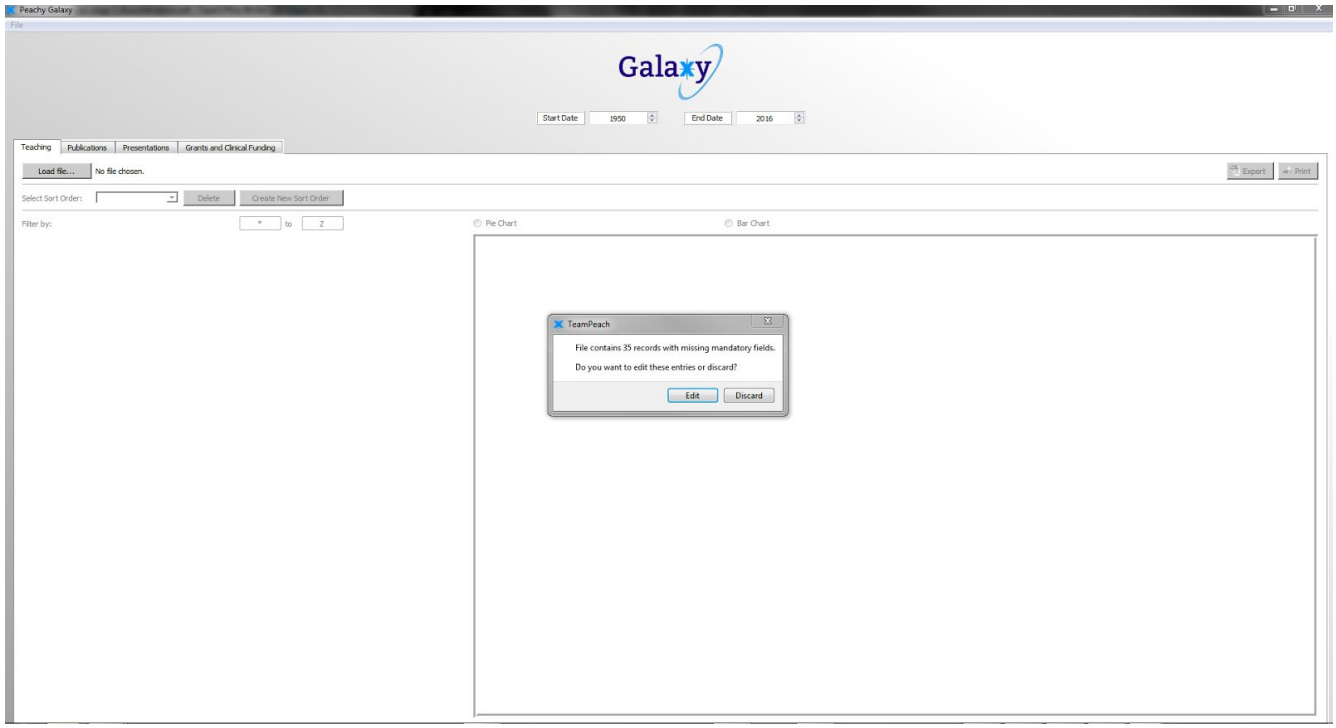
1. Open program normally



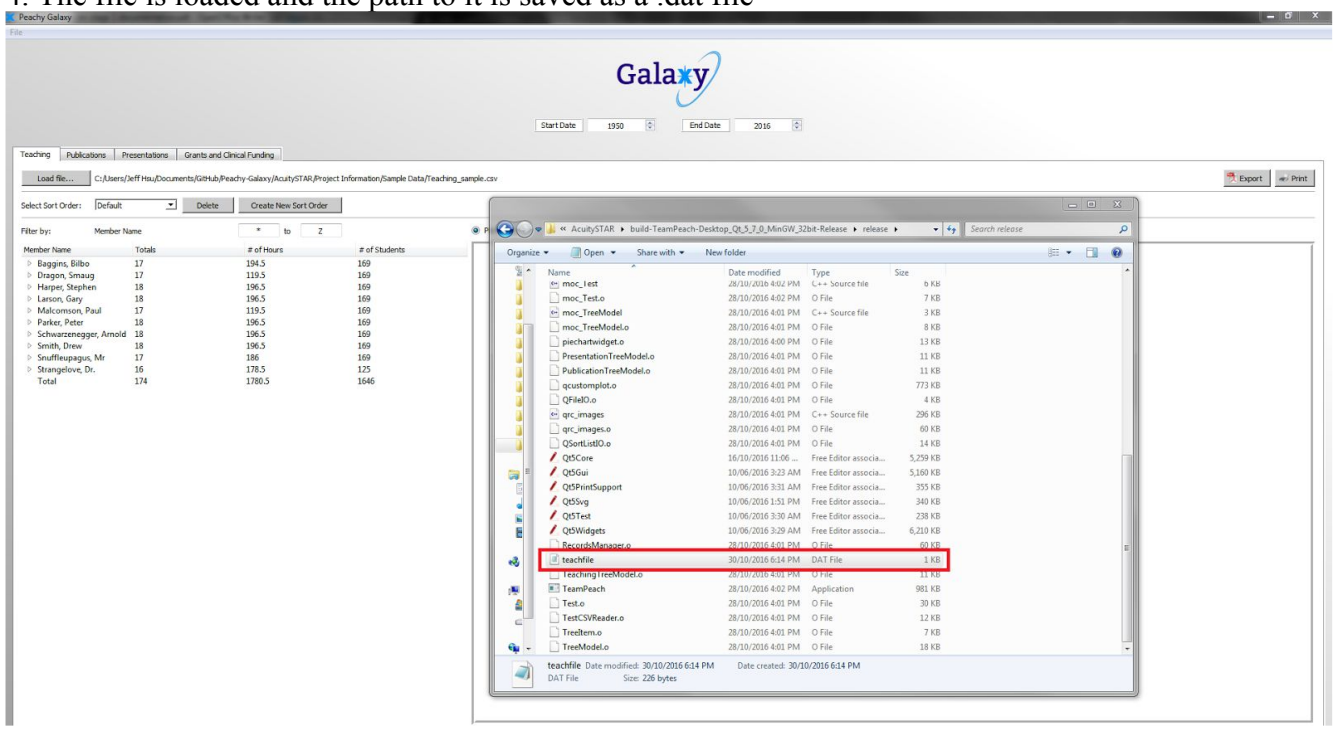
2. Load an appropriate file



3. Loading files manually will cause this prompt to appear if there are missing mandatory fields

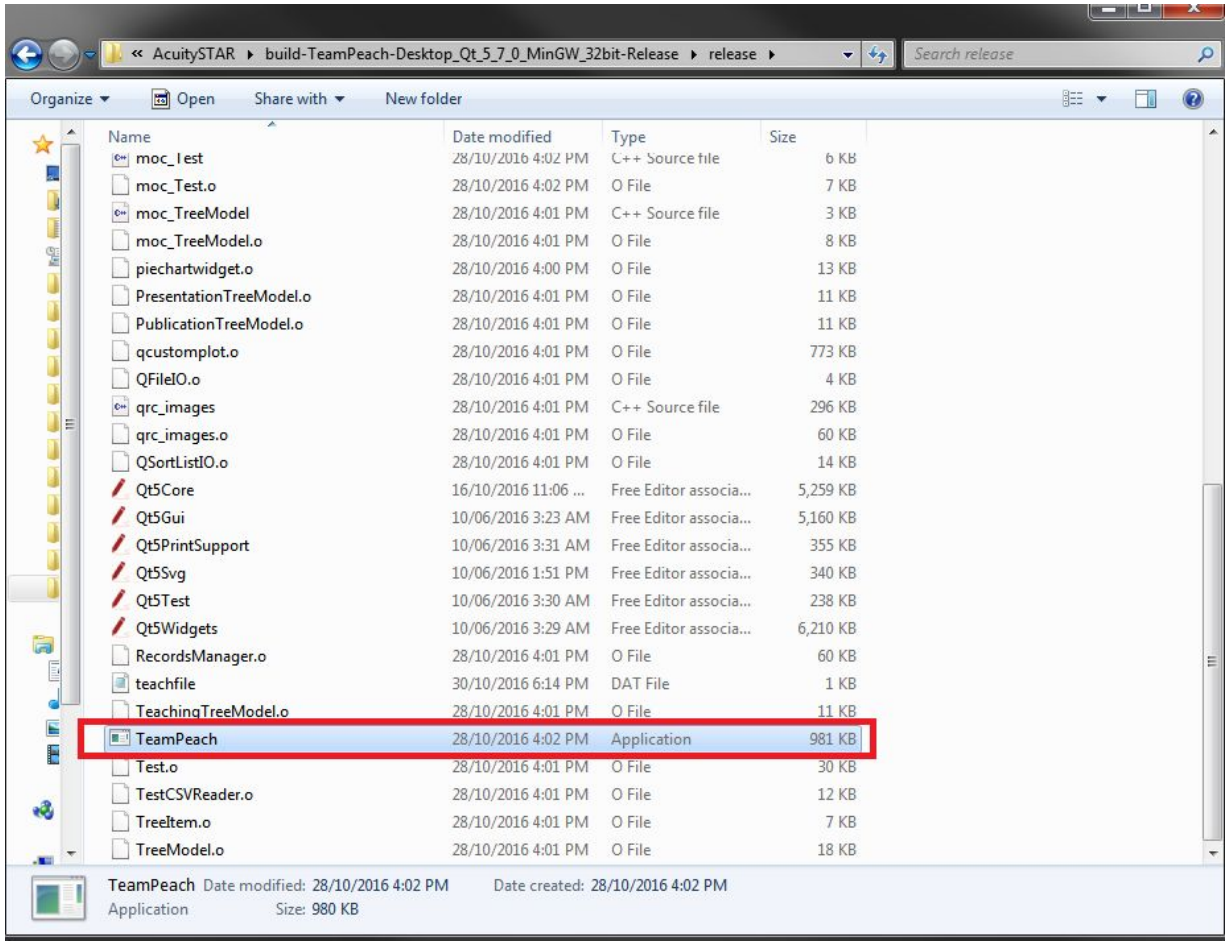


4. The file is loaded and the path to it is saved as a .dat file

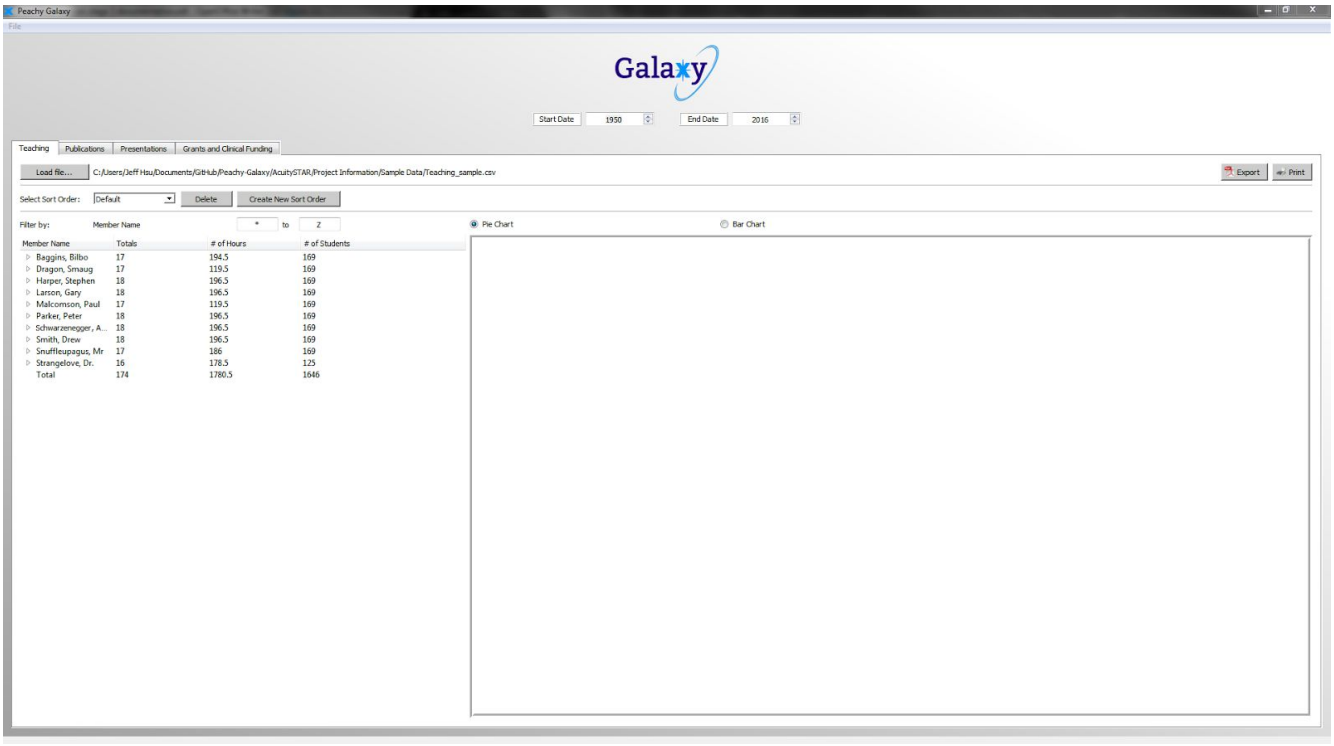


Contents of the .dat file

5. Close and reopen the program



6. You will see that on startup the file opened during the previous session will be loaded automatically.



Mandatory Requirement 4 (Expand Phase I Project by adding the ability to sort by the member's division.)

ID 04

Description

We expanded Phase I Project by adding the ability to sort by the member's division. In Phase I Project, the users are only able to sort teaching data by member name, start date and program. In Phase II, we have added the ability to sort teaching data by a member's division. Now the users have four sort options to choose by creating their new sort order (Member Name, Start Date, Program, & Division). They also can create a new sort order, and only sort data by the member's division.

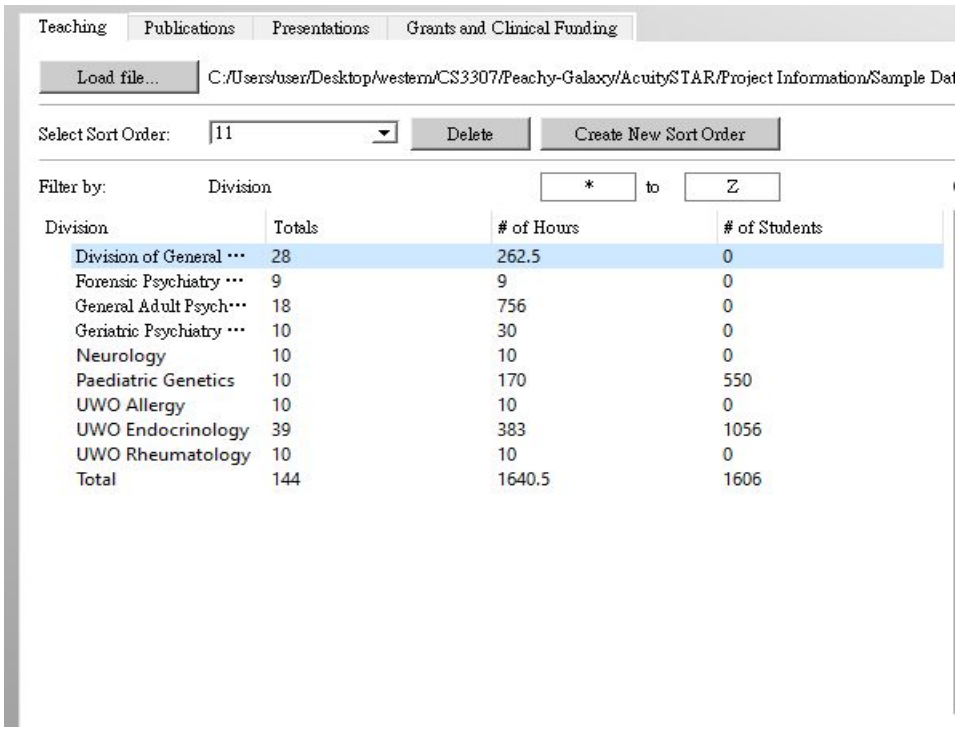
We added a new variable and a judgement statement in the "checkFile" function in the mainwindow.cpp file. So the users are able to sort data by the member's division if they choose division in the Create New Sort Order (CustomSort.cpp) dialog widget.

Origin

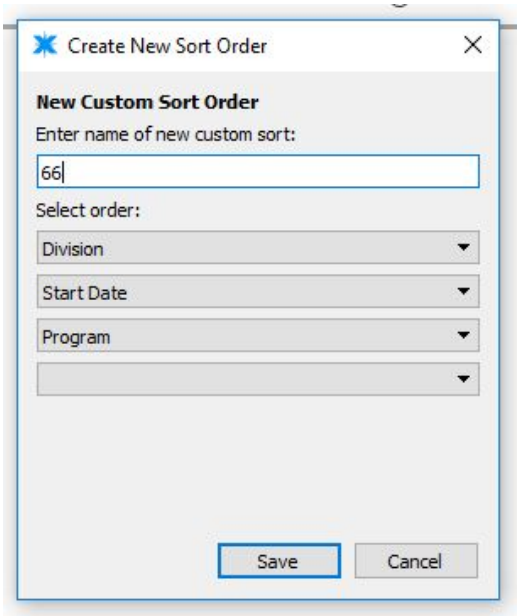
Implemented by James and Ming of Team Orion

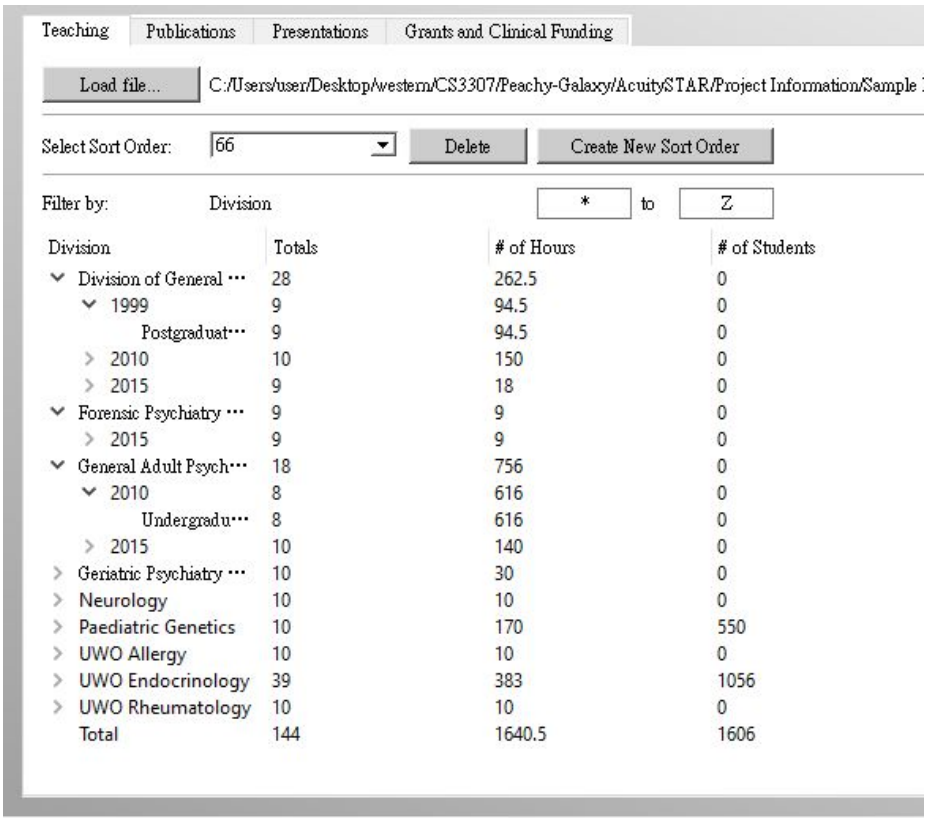
Examples

Sort by the member's division individually in Teaching



Create new sort order, sort by division, start date, then program





Division	Totals	# of Hours	# of Students
Division of General ···	28	262.5	0
1999	9	94.5	0
Postgraduat···	9	94.5	0
2010	10	150	0
2015	9	18	0
Forensic Psychiatry ···	9	9	0
2015	9	9	0
General Adult Psych···	18	756	0
2010	8	616	0
Undergradu···	8	616	0
2015	10	140	0
Geriatric Psychiatry ···	10	30	0
Neurology	10	10	0
Paediatric Genetics	10	170	550
UWO Allergy	10	10	0
UWO Endocrinology	39	383	1056
UWO Rheumatology	10	10	0
Total	144	1640.5	1606

Test Cases

Test case for Line Chart Requirement

In this requirement, we added 5 functions, and we wrote test cases for all of them.

```
//test on_teach_line_button_toggled function
void Test::test_on_teach_line_button_toggled(){
    w.on_teach_line_button_toggled();
    QCOMPARE(w.ui->teach_graph_stackedWidget->currentIndex(),2);
}
```

```
//test on_pres_line_button_toggled function
void Test::test_on_pres_line_button_toggled(){
    w.on_pres_line_button_toggled();
    QCOMPARE(w.ui->teach_graph_stackedWidget->currentIndex(),2);
}
```

```
//test on_pub_line_button_toggled function
void Test::test_on_pub_line_button_toggled(){
    w.on_pub_line_button_toggled();
    QCOMPARE(w.ui->teach_graph_stackedWidget->currentIndex(),2);
}
```

```
//test on_fund_line_button_toggled function
void Test::test_on_fund_line_button_toggled(){
    w.on_fund_line_button_toggled();
    QCOMPARE(w.ui->teach_graph_stackedWidget->currentIndex(),2);
}
```

```
//test setupLineChart function
void Test::test_setupLineChart() {
    int size = 5;
    std::vector<std::pair <std::string, double>> chartList;
    for (int i = 0; i < size; i++) {
        chartList.emplace_back("test", static_cast<double>(0.0));

    }
    w.setupLineChart(w.ui->teachLineChart,chartList);
    QCOMPARE(w.ui->teachLineChart->plottableCount(),(int) chartList.size());
}
```

Test result :

```
PASS   : Test::test_on_teach_line_button_toggled()
PASS   : Test::test_on_pres_line_button_toggled()
PASS   : Test::test_on_pub_line_button_toggled()
PASS   : Test::test_on_fund_line_button_toggled()
PASS   : Test::test_setupLineChart()
```

Test Case for sorting by division

We wrote test case for sorting by division

```
void Test::testSortByDivision()
{
    bool testPasses = false;
    //QString path = "../Project Information/Sample Data/Teaching_sample.csv";
    //w.load_teach(path, false);      //load teaching file

    QStringList newSortOrder = (QStringList() << "Division" << "Program"); //create new sort order (simulates w.on_teach_new_sort_clicked()

    w.allTeachOrders << newSortOrder;
    w.ui->teach_sort->addItem(newSortOrder.at(0));  //add new sort order to mainwindow attributes

    for (int i=0; i < w.allTeachOrders.size(); i++)  //check if sort orders contain a tier for "Division"
    {
        QStringList qsl = w.allTeachOrders.at(i);
        if (qsl.contains("Division"))
        {
            testPasses = true;
            //maybe could also be done by evaluating the tree model
        }
    }

    QVERIFY(testPasses);

}
```

Test result:

```
PASS   : Test::SaveTestPub()
PASS   : Test::SaveTestPres()
PASS   : Test::SaveTestFund()
PASS   : Test::testSortByDivision()
PASS   : Test::test_on_teach_line_button_toggled()
PASS   : Test::test_on_pres_line_button_toggled()
PASS   : Test::test_on_pub_line_button_toggled()
PASS   : Test::test_on_fund_line_button_toggled()
```

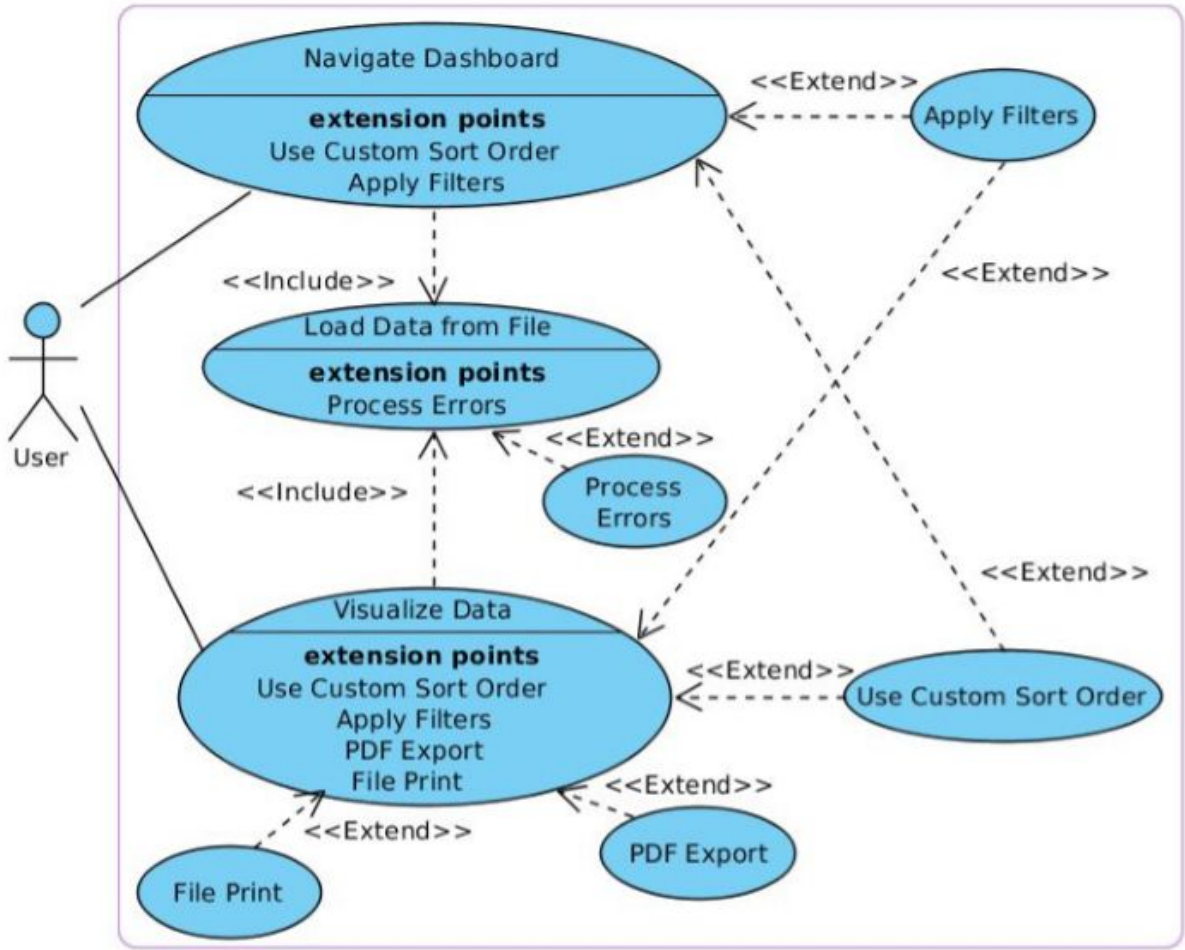
Testing matrix

	A	B	C	D	E	F	G	H	I	J	K
Line charts	X	X	X	X	X						
Save session						X	X	X	X	X	
Sort by member division											X

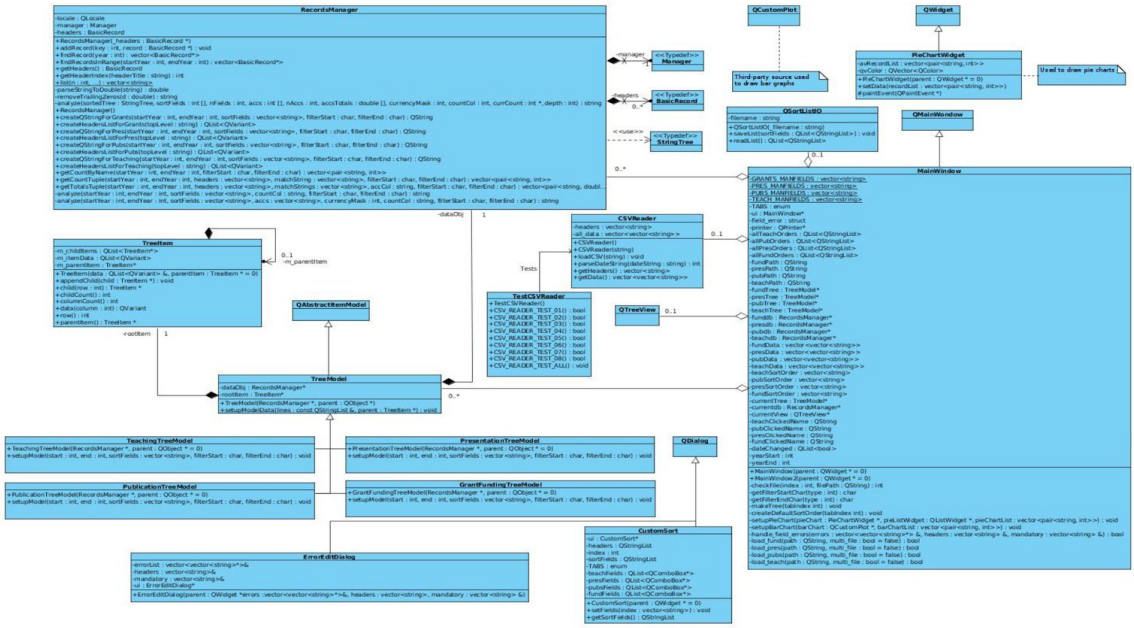
A = test_on_teach_line_button_toggled
B = test_on_pres_line_button_toggled
C = test_on_pub_line_button_toggled
D = test_on_fund_line_button_toggled
E = test_setupLineChart
F = NoSaveTest
G = SaveTestTeach
H = SaveTestPub
I = SaveTestPres
J = SaveTestFund
K = TestSortByDivision

System Design

Use Case Diagram



Original Class Diagram



Enhanced Class Diagram (pdf “delv 3 uml v3” also located in project folder)

- Added functions for line chart, sort by division and save state in mainwindow
- Implemented QtTests in Test.cpp

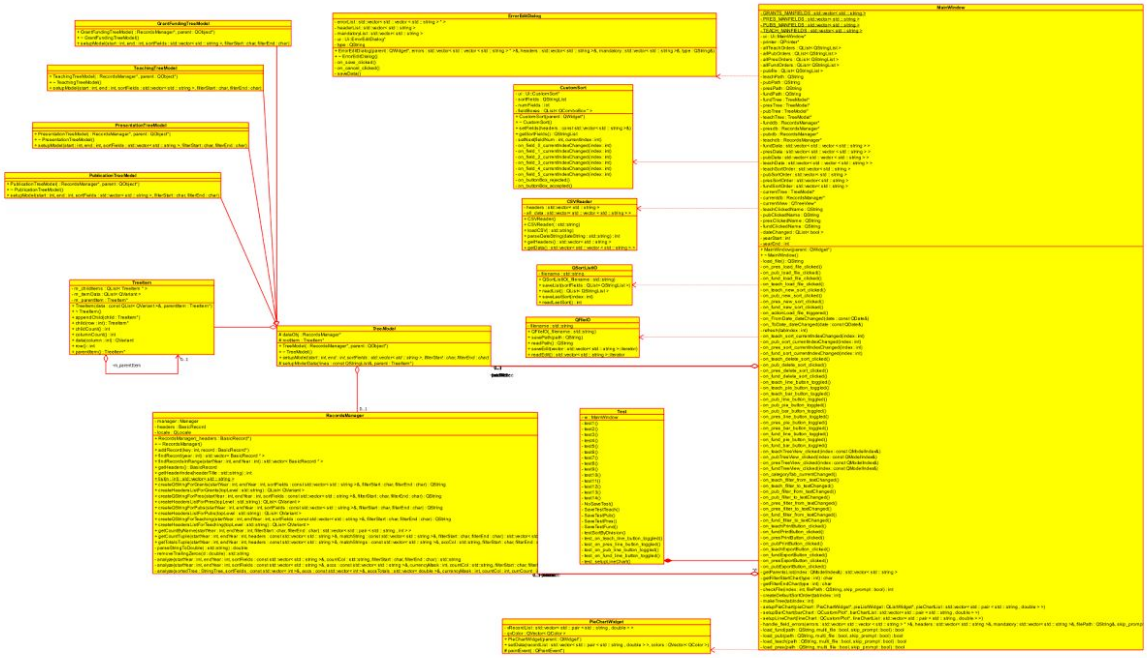


Diagram: Devlierable 3 UML Page 1

Development Plan

Development plan for team Orion's deliverables 2 & 3 were straightforward with little to no confusion or interference.

Team Orion - Peach Galaxy Phase 2

TimeLine

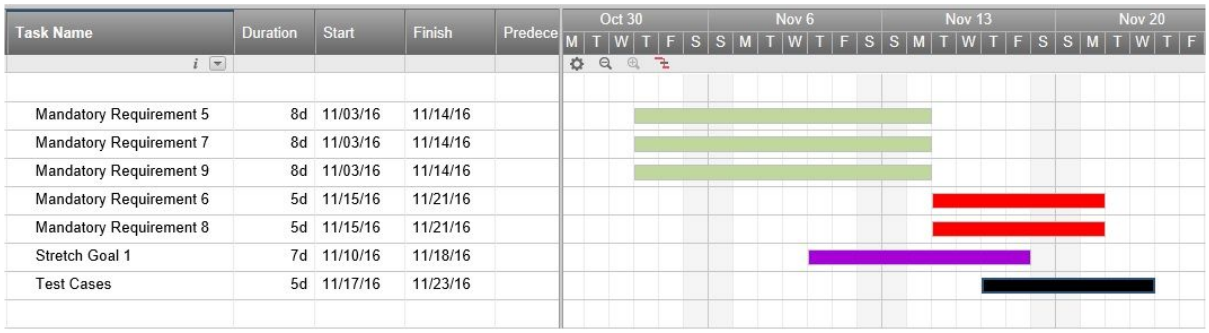
Achievements	Estimated Due Date	Date completed
Phase 1 system's Documentation analyzed	October 16, 2016	October 16, 2016
Phase 1 sustem's Documentation completion	October 20, 2016	October 21, 2016
Improvements to phase 1 system identified and logged	October 18, 2016	October 18, 2016
Test Cases created for Phase 1 system	October 20, 2016	October 21, 2016
Demo 1 requirements determined	October 27, 2016	October 27, 2016
Demo 1 requirements tasks allocated to members	October 27, 2016	October 27, 2016
Development infrastructure set up	October 29, 2016	October 30, 2016
Requirements completed	October 30, 2016	October 31, 2016
Test-cases completed	October 31, 2016	October 31, 2016
Documentation of deliverable 3	October 31, 2016	October 31, 2016

Team Orion Meetings: (From Deliverable 1 to end of Deliverable 2)

Date	Where	Discussed	Absent
11/10/2016	MC325	- Partial analysis of Phase 1 with team - Identified mutual time and date to meet - Created discord for communication	No One
14/10/2016	MC325	- Assigned topics for members to attend to for the Deliverable 2	Michael
21/10/2016	Online	- Merged all documents together into one PDF on the analysis of Phase 1 - Added test cases - Handed in Deliverable 2	Michael
27/10/2016	MC325	- Discussed what requirements should be tackled for the implementation of Phase 2 - Divided into groups to handle these operations	Michael

A Plan for Stage 2:

This Stage will require the remaining mandatory requirements to be completed. The due date for this deliverable is the 23rd of November.



Task Name	Duration	Start	Finish	Predece
Mandatory Requirement 5	8d	11/03/16	11/14/16	
Mandatory Requirement 7	8d	11/03/16	11/14/16	
Mandatory Requirement 9	8d	11/03/16	11/14/16	
Mandatory Requirement 6	5d	11/15/16	11/21/16	
Mandatory Requirement 8	5d	11/15/16	11/21/16	
Stretch Goal 1	7d	11/10/16	11/18/16	
Test Cases	5d	11/17/16	11/23/16	

Abstract Plan for Final Stage:

This is under the assumption that everything that is hoped to be accomplished in Stage 1 and 2 is completed. In the remaining 2 weeks left of the project, most of the time will be focused towards adding additional components along with a tutorial video displaying how to use the new feature. Team Orion also has a major concern with the installation of this software to make certain everything installs and runs properly, aergo a good portion of time will be spent testing this along with the software being utilized on other machines i.e. not just Windows 7 machines.

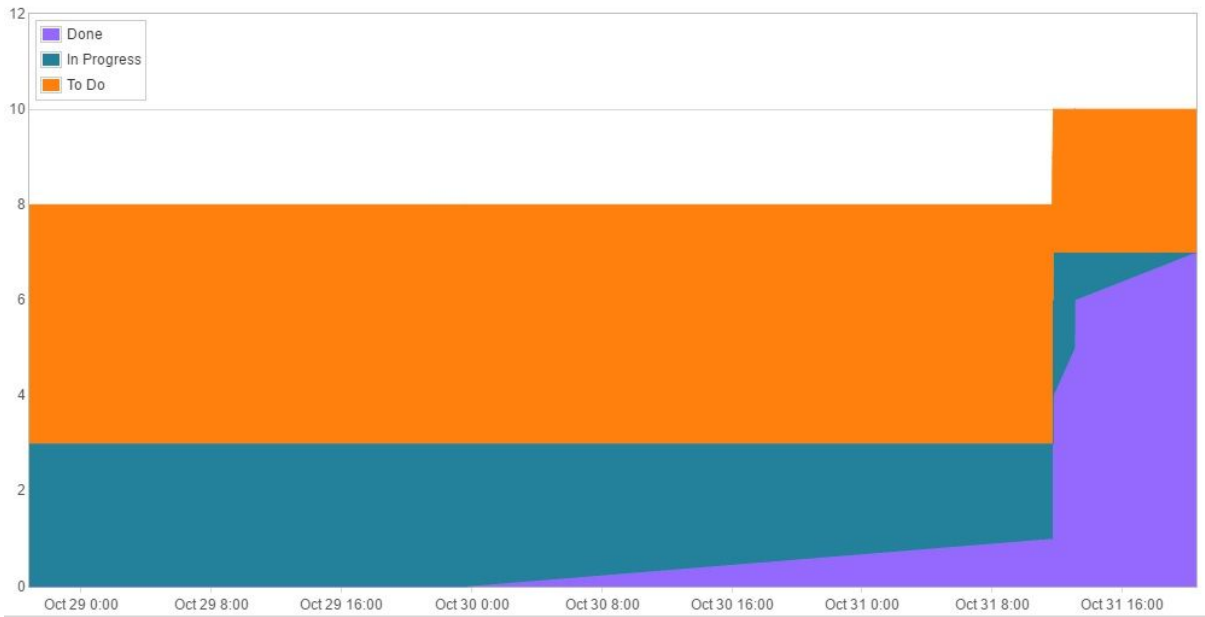
Task Association chart

Team Orion - Peach Galaxy Phase 2

Task Allocation

Role	Name	Assignments	Due Date	Priority
Project Manager	Paul Henderson	Create and Implement Development plan For stage 1 & 2.	31-10-2016	Medium
		Create Excel Spreadsheet Showcasing all assigned goals	31-10-2016	Low
		Scheduled meetings and conducting them professionally	31-10-2016	Low
		Assisted in creating "Save-Session" State for Stage 1	31-10-2016	Medium
Lead Designer Github Host	Chris Brown	Implemented design and back-end code for new "Graph" requirement	31-10-2016	High
		Created test-cases for that same requirement	31-10-2016	Medium
		Co-created the Documentation for this requirement	31-10-2016	Medium
Developer	Jeffrey Hsu	Implemented back-end code for creation of "Save-Session" state requirement	31-10-2016	High
		Created test-cases for that same requirement	31-10-2016	Medium
		Added Documentation for this requirement	31-10-2016	Medium
Designer	Yuchen Wang	Assisted with creation of "Grah" requirements	31-10-2016	High
		Assisted with creation of test-cases for those graphs	31-10-2016	Medium
		Co-created the Documentation for this requirement	31-10-2016	Medium
Quality Assurance Developer	Jackson Yang	Implementation of adding "Division" field for sorting section	31-10-2016	High
		Assisted with test-cases for "Division" field	31-10-2016	Medium
Developer/Designer Maven Lead	James Walsh	Assisted with implementation of "Division" field	31-10-2016	High
		Implementation of test-cases for this requirement	31-10-2016	Medium

Issues tracked



Tasks were evenly distributed between the group ensuring everyone was given the chance to do both coding and documentation. It is to be noted that a member of team Orion has not been present in any meetings or given any tasks. This member has not been seen in approximately a month for unknown reasons.