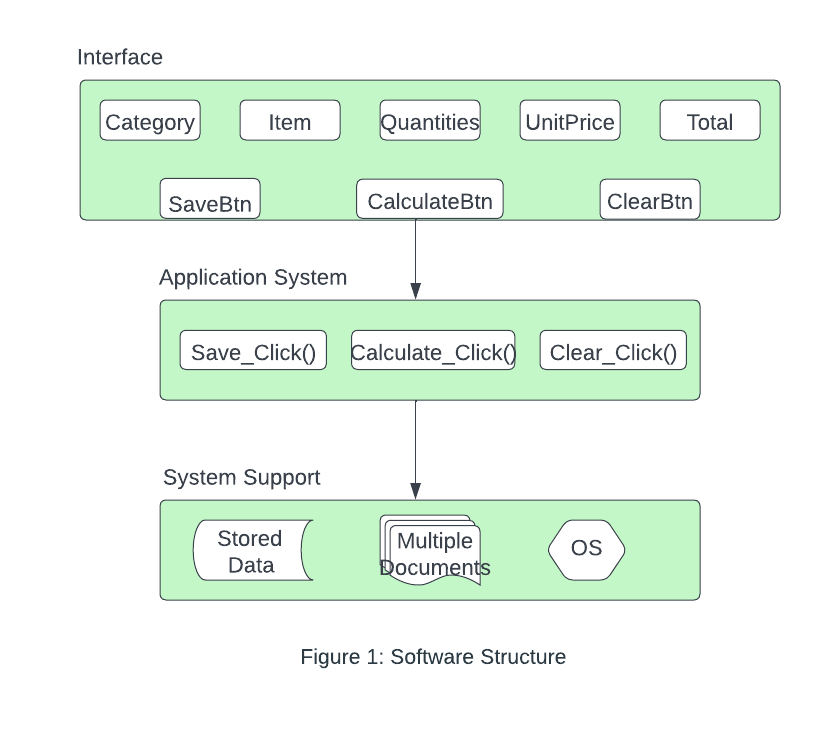
**Design Document**

* **Introduction** - The introduction should include information describing the intended software project (including the hardware and software that the software application will run on). This introduction should be at least 50 words in length.

This project will be a windows form application run on Visual Studio in C#. The project will be a Material List Estimator. It will allow the user to pick Items from a list and the quantity of that said item. There will also be a budget per category option letting the user set their own budget per category. Once the user is done with the list it will calculate the total amount spent and display a pie chart depicting all items bought and the most expensive items.

* **Architecture** - Another section should include the architecture you plan on using. For example, if you plan on using a layered approach, include a diagram such as seen in figure 6.8 with general information about what will be included in each layer.

Our software architecture will be a Layered Architecture pattern. The structure is shown in Figure 1 below.



* **Use Cases** - There should be a section that includes use cases for the software development project. There should be at least one use case for each requirement that you identified on the requirements analysis document for your software project.
* Adds up all the totals for each row and displays the correct total at the bottom.

Calculating the totals for each row is beneficial to the user because it will show the total cost of their project. This can help them budget effectively if the cost exceeds their budget.

* Calculates the order quantity times the unit price and displays the correct total at the far right of each row.

Calculating per unit price x quantity allows the user to budget effectively depending on the unit price and the quantity of said items.

* The clear button will clear the entire table and begin a new materials list.

The clear button will let the user erase the whole spreadsheet. This is a useful button because rather than having the user erase item by item it will let the user completely erase the spreadsheet.

* Save button will save the information to a file

The save button will let the user save their spreadsheet allowing the user to come back to the sheet without losing all of their progress.

* Calculate button will display the total estimated cost, budget exceeded, and pie chart.

The calculate button will display the total estimated cost, budget exceeded, and pie charts. These displays will help the user with budgeting their project and learning the most expensive items purchased which can help the user decrease cost overall.

* Text boxes will allow the user to enter type letters and numbers into the item description and item. However, it will only allow the user to enter numbers into quantity, unit price, and budget. If the user tries to type letters into the text boxes, it will inform the user that letters are not allowed via a prompt.

The text boxes will let the user input their own list information. An error message will appear when typing a letter in the quantity box will help make sure there aren't any errors when using the spreadsheet.

* Our entire application will be built in a grid in a similar manner to how excel’s grid looks.

Having the spreadsheet look similar to the excel grid will let the user feel familiar when first opening our program, letting the user feel more comfortable using the spreadsheet.

* The pie chart will use the total of each category and divide it by the total estimated cost determining the percent sliver of each slice on the pie graph.

The pie chart will depict a percentage of the cost of all purchased items, allowing the user to see what items are cost effective and what categories/items they can spend more on or less than.

* Allow the user to input a budget for each category and then a budget for overall and if the user exceeds the budget when they hit calculate, it will highlight the category that exceeds the budget in the red.

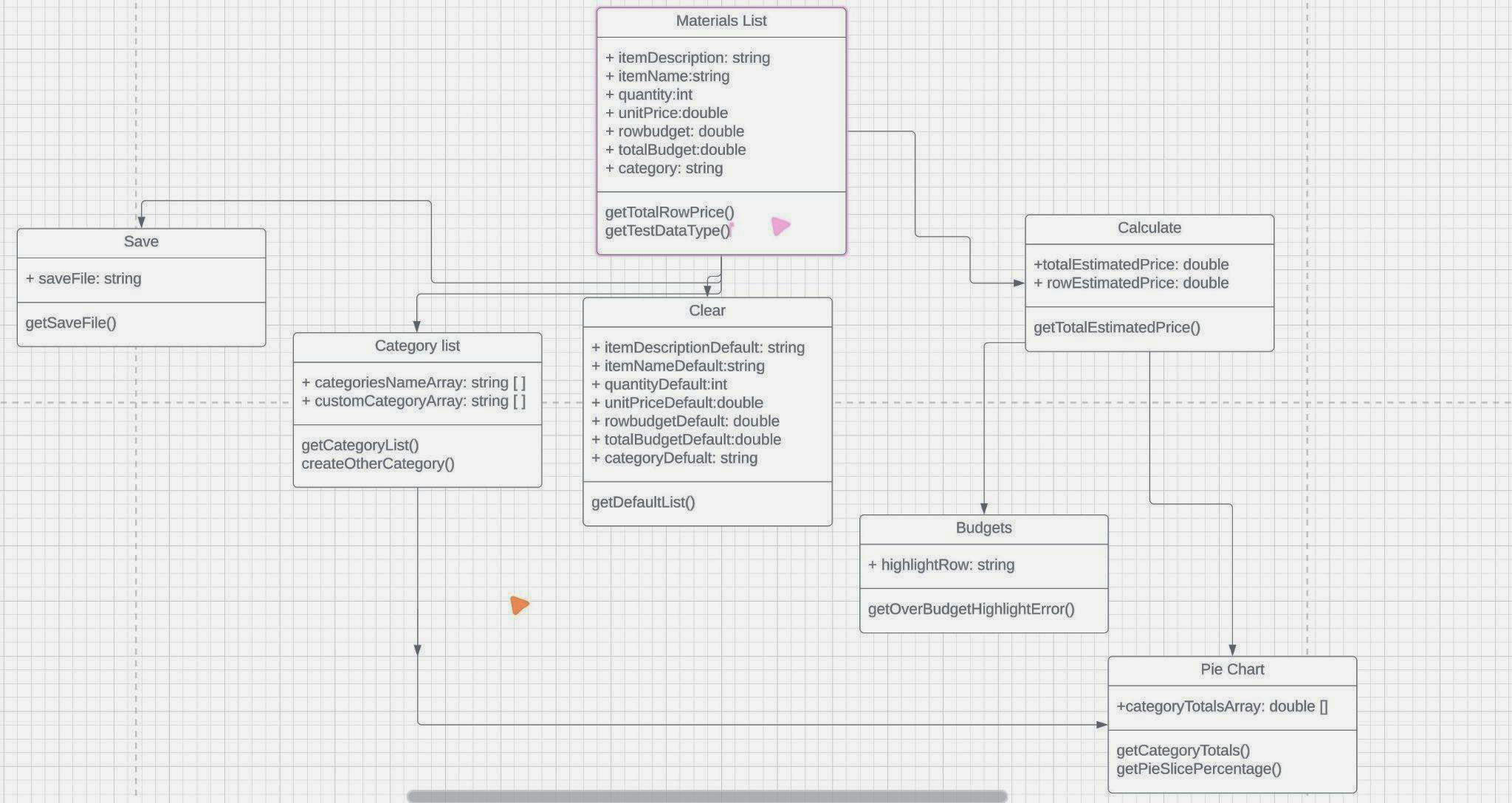
The budgeting option for each category will help the user pick items and the amount of them without going over their budget. The budgeting tool will also help the user know if they still have room in their budget.

* When the user hits the calculate button, it will highlight any category that exceeds their budget or which category that is most expensive in red.

This feature just helps the user more when budgeting using the spreadsheet. Letting the user know where to cut back on item quantities helps the user not go over their budget.

* There will be a textbox that will allow the user to add a new category to the drop list if they don’t like the current categories given.

This just lets the user add more and more to the spreadsheet allowing them to calculate the total of many more products.

* **UML** - there should be a section that includes the currently identified classes that you intend on using (see chapter 7). The UML should also include the relationships between the various classes.
* **Test Cases** - There should be a section that includes test cases for the software development project. There should be at least one test case for each requirement that you identified on the requirements analysis document for your software project.

REQUIREMENTS

1. Adds up all the totals for each row and displays the correct total at the bottom.

To test if this section is working one of the group members can test multiple times per row and total.

1. Calculates the order quantity times the unit price and displays the correct total at the far right of each row.

Like the first activity a member of this group can manually calculate the totals multiple times to ensure the math is correct.

1. The clear button will clear the entire table and begin a new materials list.

Using the button will be the best way to test if it works, making sure the spreadsheet is fully cleared.

1. Save button will save the information to a file

To test the save button we will type into the spreadsheet and save it. Then closing the program and re-opening the program to make sure the saved data is still there.

1. Calculate button will display the total estimated cost, budget exceeded, and pie chart.

To test the calculate button we can click the button and see the output and make sure it matched up with our own physical calculations. This test will be done countless times to ensure the correct totals every time.

1. Text boxes will allow the user to enter type letters and numbers into the item description and item. However, it will only allow the user to enter numbers into quantity, unit price, and budget. If the user tries to type letters into the text boxes, it will inform the user that letters are not allowed via a prompt.

The text boxes will be checked when running the program to make sure they are working properly. To check the quantity error code when typing a letter instead of a number we will just type into the text box to make sure it is working.

1. The drop-down list will be enabled on the far left of the grid and will allow the user to select from a variety of categories successfully.

To test this function we will manually click the drop down list and make sure it successfully selects a category.

1. Our entire application will be built in a grid in a similar manner to how excel’s grid looks.

We will make sure the spreadsheet looks and functions similar to Excel.

1. The pie chart will use the total of each category and divide it by the total estimated cost determining the percent sliver of each slice on the pie graph.

When displaying the pie chart we will manually calculate the totals per percentage to make sure they match with the pie chart. We will also test to make sure each slice of the pie represents the percentage correctly.

1. Allow the user to input a budget for each category and then a budget for overall and if the user exceeds the budget when they hit calculate, it will highlight the category that exceeds the budget in the red.

We will make a mock spreadsheet with budgets per category and test if the budget system does turn red when over.

1. When the user hits the calculate button, it will highlight any category that exceeds their budget or which category that is most expensive in red.

We will make a mock spreadsheet with budgets per category and test if the budget system does turn when the calculate button is pressed.

1. There will be a textbox that will allow the user to add a new category to the drop list if they don’t like the current categories given.

This will be tested by clicking on the textbox and seeing if a category and items appear in the correct spot and are fully functional.

* **Summary** - The design document should conclude with a summarization of the project. The summary should be at least 50 words in length.

This project will be a Material List Estimator, it will be designed and coded in Visual studio in C#. This project will allow a user to enter items into a spreadsheet and pick the quantities of the said items. Once the user is finished with the spreadsheet it can calculate the total cost of items and display a chart showing most expensive to least expensive items that were picked by the user.