**Library Name**: Financial Reporting Library

**Description**:  The financial reporting library allows developers to summarize, display and link personal and corporate financial information. The core functionality is as follows:

Function 1: Provide customizable formats for basic financial reports such as a balance sheet, income statement, cash flow statement and statement of equity. The library will allow users to customize the format, change the statement structure, allow fields to be editable or disabled, determine the level of subtotalling and customize the UI.

Function 2: Developers will be able to link specific line items in the financial statements (such as cash) to a more detailed tabular break down of cash (ex. savings, investments, RRSPs, etc) or provide time series data of cash changing over time. The library will provide validation checking as well so that the balances match, but the primary feature will include linking to more granular information. The format will be customizable such as linking to a new table or collapsing an existing table for example.

Function 3: Minor data manipulation such as subtotalling, totalling data and cross-referencing information across statements such as comparing cash from the balance sheet to the cash flow statement.

The use cases are : a) Personal financial companies like mint, wave and quickbooks provide financial information for individuals and corporations in financial statement form such as balance sheets, income statements and cash flow statements. There are many ways that users can input, display and summarize financial information. There is a need for developers to summarize, link and display and input this information for users of their applications.

b) Companies that post their financial information on their wbsites on the public relations section of the website usually display the contents of a PDF which can be downloaded or it is displayed statically on the website. Providing dynamic, linked financial information will allow web users to better analyze the company financial information if it is easier to consume and click through. While this library provides a niche function, I believe the case is broad enough to allow for extensive customizability and great utility for financially oriented web applications. I may also add budgeting customizable statements to broaden the library if the above is too limited.

**Implementation so far**: I have implemented a basic version of function 1, which contains basic reports for the balance sheet, income statement, cash flow and equity statements. At this time, there are only 3 basic customizations. First, the user can include subtotal categories. The user can also specify if the field is editable or the user can specify if the field is fillable in which case, users can input financial information directly into the web page.

I have a implemented a version of function 2 whereby a developer can include collapsible sub tables under a line in a financial report that provide granular financial information about a specific line item. A button that collapses or expands the line item demonstrates this feature.

I have also implemented a simple version of function 3 where there is minor validation on the balance sheet such as assets = liabilities + equity. Subtotals also automatically updated based on edited or inputted financial data.

**Link to website**: <https://powerful-waters-30511.herokuapp.com/examples.html>

**Structure of JS objects**: The core objects in my library are the financial report, balance sheet, income statement, cash flow statement and equity statement classes. Balance sheet, income statement, cash flow statement and equity statement extend financial report. The basic component of financial report is the header data, which typically includes the entity name, report name and year. This can be found in the form. This is the required input form into this object.

{

name: “Fraud Co”,

report: “Income Statement”,

year: “2020”

}

The balance sheet, income statement, etc contains the header data, the data which consists of financial information, assets, liabilities, equity, assetSize, liabilitySize, equitySize. This information in each objects is of the form as follows. This is the required input form into this object.t

{

header: //This will be in the form shown above.

data = {

assets: {

"Cash": 100,

"Accounts Receivable": 200,

"PPE": 300,

"Other Assets": 200

},

liabilities: {

"Accounts Payable": 200,

"Long term Debt": 400

},

equity: {

"Retained Earnings": 100,

"Outstanding Shares": 100

}

}

//Assets, liabilities, equity, etc subdivide the data.

}

This data is used to populate the financial report line items. For example this data is intended for the balance sheet financial report, which is a tabular form subdivided into each section – assets, liabilities and equity. Each section shows the line items and the value of that line item in the DOM.

In each financial report, there are methods that construct the type of report required by the developer. See below for API functionality.

**API functionality:**

genereateHeader(selector)

The selector is the id, class, html component where the header form will be displayed in the DOM. This method will generate a tabular form containing the header data of the financial report.

generateFillableBody(selector, subtotal)

The selector is the id, class, html component where the form will be displayed in the DOM. The subtotal argument is a Boolean value. If true, then subtotals will be computed and displayed in the report. This function is called on a report object such as a balance sheet. It will manipulate the DOM to generate a fillable tabular report. If the data is not empty, then the fillable report will show the existing values as placeholders in the input form.

generateFormBody(selector, subtotal, edit)

The selector and subtotal arguments are the same as the arguments in the generateFillableBody method above. The edit argument is a Boolean. If true, then an edit button will be displayed beside each value in the report.

**Features to be implemented**:

Reporting linking

Cross-report checks

Horizontally collapsible budget data

CSS to improve UI experience and offer customization