AP Computer Science Final Project - README Template

Instructions:

The first step in creating an excellent APCS final project is to write up a README. At this stage, this README file acts as your **project proposal**. Once you’ve filled in all components, Shelby will read through it and suggest edits. Ultimately, you need a document that adequately describes your project idea and **we must agree on this plan**.

Have one member of your group **make a copy of this Google Doc**. Then, they should share it with all other members **and with Mr. Shelby** so that every group member has edit permissions, and Shelby can add comments on your ideas.

There’s a lot of parts of this document that you might not have full answers for yet. Because you haven’t written the program yet, it’s difficult to think about the **instructions** or **which group members will do which parts**. Even though this is hard to think about, you must have something in these sections that acts as your current plan. However, during the course of the project, you’ll **continuously update this document**. This means that you will not be *held* to exactly what you put here - components of this document can change (and it’s pretty common!).

There is one exception: the **Features List** section. Once Shelby OKs your README, the Features List section **cannot be modified**. For this reason, it is most important that you get a solid idea of what you want to make and the primary features it will have *now*.

Talk with your group. Consider drawing some pictures of what you think your project might look like. Be precise. When you’re ready, fill this out together. Each component in brackets below ( [these things] ) should be replaced with your ideas. Note that there are several sample READMEs posted on this assignment for you to use as guidance.

**-------------------When README is finalized, remove everything above this line--------------------**

Stock Chart

**Authors**: Ryan Xu, Leor Porat

**Revision**: 4/15/22

**Introduction**:

[In a few paragraphs totaling about ½ page, introduce the high-level concept of your program. What this looks like depends a lot on what type of thing you are making. An introduction for an *application* will look different than one for a *game*. In general, your introduction should address questions like these:

What does your program do?

* It draws a stock chart and calculates a price target based on extracted data

What problem does it solve? Why did you write it?

* It saves a lot of time for invesors who want to calculate a precise price target
* I wrote it so I can use it to expedite and improve my investments

What is the story?

* n/a

What are the rules? What is the goal?

* n/a

Who would want to use your program?

* Investors

What are the primary features of your program?

* Stock chart, price target (DCF) calculator

**Instructions**:

[Explain how to use the program. This needs to be **specific**:

Which keyboard keys will do what? n/a

Where will you need to click? Anywhere, will draw line

Will you have menus that need to be navigated? What will they look like? no

Do actions need to be taken in a certain order? must first upload txt version of earnings data to generate price target

**Features List (THE ONLY SECTION THAT CANNOT CHANGE LATER)**:

**Must-have Features**:

[These are features that we agree you will *definitely* have by the project due date. A good final project would have all of these completed. At least 5 are required. Each feature should be fully described (at least a few full sentences for each)]

* Extraction of data sets for at least 5 stocks (close price)
* Visualization of data (line chart)
* User interface
* Earnings report extractor
  + Discounted cash flow automatic calculation
    - Calculated price target visualization

**Want-to-have Features**:

[These are features that you would like to have by the project due date, but you’re unsure whether you’ll hit all of them. A good final project would have perhaps half of these completed. At least 5 are required. Again, fully describe each.]

* expanding/contracting
* drawing lines
* Moving averages
* Candle chart
* MACD, RSI

**Stretch Features**:

[These are features that we agree a *fully complete version of this program would have, but that you probably will not have time to implement*. A good final project does not necessarily need to have any of these completed at all. At least 3 are required. Again, fully describe each.]

* Automated technical analysis (support & resistance at least)
* Fully automated trading algorithm
* Automatic earnings report extractor
* Portfolio analysis (alpha, beta)

**Class List**:

[This section lists the Java classes that make up the program and very briefly describes what each represents. It’s totally fine to put this section in list format and not to use full sentences.]

* AlphaVantageConnector.java: connects to alphavantage api
  + ApiConnector.java: represents a connection to an api endpoint
* TimeSeries.java: provides real time stock data (focus on intraday and daily)

**Credits**:

[Gives credit for project components. This includes both internal credit (your group members) and external credit (other people, websites, libraries). To do this:

* Ryan: Brainstormed project idea, completed README,
  + Will figure out how to extract the data using Maven and Alpha Vantage, design the DCF formula, and incorporate it into the chart
* Leor:
  + Will program the user interface, data extractor
* Credits
  + AlphaVantage
  + Patriques82 on github
  + Gradle