

**Software Design Document Specification Template**

The Software Design Specification (SDS) sections provide you with guidelines related to the structure and the contents of SDS document. The Software Design Specification document includes at least these sections.

For the project, your team may have good reasons for wanting to deviate from this proposed outline. If a section is not applicable in your case, do not delete it; instead, give the topic heading and write "Not applicable".

You will note that there is some overlap in the content between different documents (i.e. the User Requirements Specification Document and the Software Design Specification Document.) This redundancy allows each document to stand on its own.

***ONLY THE SECTION TITLES COLORED IN ORANGE ARE REQUIRED TO BE COMPLETED.***

***DO NOT DELETE THE SECTIONS YOU ARE NOT COMPLETING AS THEY ARE A PART OF THE DOCUMENT***

Contents

[1 Introduction 3](#_heading=h.gjdgxs)

[1.1 Purpose of this document 3](#_heading=h.30j0zll)

[1.2 Scope of the development project 3](#_heading=h.1fob9te)

[1.3 Definitions, acronyms, and abbreviations 3](#_heading=h.3znysh7)

[1.4 References 3](#_heading=h.2et92p0)

[1.5 Overview of document 3](#_heading=h.tyjcwt)

[2 System architecture description 3](#_heading=h.3dy6vkm)

[2.1 Overview of modules / components 3](#_heading=h.1t3h5sf)

[2.2 Structure and relationships 3](#_heading=h.4d34og8)

[2.3 User interface 3](#_heading=h.17dp8vu)

[2.4 User interface issues 3](#_heading=h.3rdcrjn)

[3 Detailed description of components (ONLY 2 ARE REQUIRED) 3](#_heading=h.26in1rg)

[3.1 X Component (or Class or Function ...) 3](#_heading=h.lnxbz9)

[3.2 Y Component (or Class or Function ...) 4](#_heading=h.35nkun2)

[4 Reuse and relationships to other products 4](#_heading=h.1ksv4uv)

[5 Design decisions and tradeoffs 4](#_heading=h.44sinio)

[6 Pseudocode for components 4](#_heading=h.2jxsxqh)

[7 Appendices (if any) 4](#_heading=h.z337ya)

[Software component template for section 3 4](#_heading=h.3j2qqm3)

# Introduction

## Purpose of this document

Full description of the main objectives of the SDS document.

## Scope of the development project

<This subsection should:

(1) Identify the software product(s) to be produced by name; for example, Host DBMS, Report Generator, etc.

1. Back Against Wall Street Desktop Application
2. Stock Analysis Algorithm

(2) Explain what the software product(s) will, and, if necessary, will not do **--** Back Against Wall Street is a program that

1. The Software Product will be able to log into an account
2. The Software Product will allow the user to save the stocks that they like
3. The Software Product will have a light/dark feature
4. The Software Product will allow the user to add stocks that they specifically want to be graded to the list that they want to be graded by the algorithm.
5. Users can add their credit card information
6. The Software will provide recommendations of reliable stocks that have a high chance of future growth to the users.
7. The Software will have a settings tab where the user can tailor the indicators to their liking for various purposes
8. The Software will implement a subscription service to receive stronger analysis using more complex indicators.
9. The Software will have a customer service tab
10. It will be possible to give feedback to Back Against The Wall Street through a neatly formatted feedback form.

(3) Describe the application of the software being specified. As a portion of this, it should:

(a) Describe all relevant benefits, objectives, and goals as precisely as possible. For example, to say that one goal is to provide effective reporting capabilities is not as good as saying parameter-driven, user-definable reports with a 2 hour turnaround and on-line entry of user parameters.

1. Use Yahoo Financials package to get stock data
2. Test data against 10-day,50-day, and 200-day moving average
3. Calculate support and resistance
4. Compare relative strength index (RSI)
5. Detect upper and lower Bollinger band walks
6. Take daily volume into account for analysis
7. Make prediction on which stock will perform the best
8. Create visualizations

(b) Be consistent with similar statements in higher-level specifications (for example, the System Requirement Specification), if they exist. What is the scope of this software product?>

The project will analyze a predetermined list of stocks to determine which one has the highest potential for growth in the short term. The analysis will begin with a few basic tests, and we will continue to try to add tests as long as we have time to. The first functionalities that will be implemented are a 10-day, 50-day and 200-day moving average test.

## Definitions, acronyms, and abbreviations

* **52s** – New 52 week high.
* **BAWS** – (Back Against Wall Street).
* **B/O** = Breakout.
* **Green** – Price is above previous day's close.
* **Red** – Price is below previous day's close.
* **SS or S/S** – Short Sell.

## References

This section will include technical books and documents related to design issues. Be certain that the references you give are complete and in the appropriate format.

## Overview of document

A short description of how the rest of the SDS is organized and what can be found in the rest of the document. This is not simply a table of contents. Motivate and briefly describe the various parts!

# System architecture description

## Overview of modules / components

This subsection will introduce the various components and subsystems.

## Structure and relationships

Make clear the interrelationships and dependencies among the various components. Structure charts can be useful here. A simple finite state machine can be useful in demonstrating the operation of the product. Include explanatory text to help the reader understand any charts.

## User interface

Complete design with user interactive windows, error messages, etc. Include all windows and messages that will be viewed by the user and also the ones that the user will be interacting with.

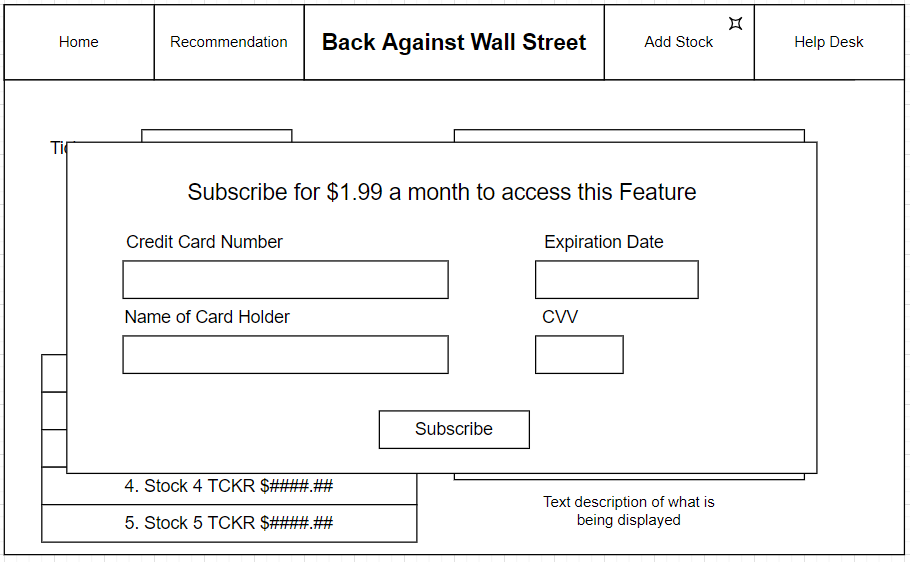
**2.3.1** Login Page



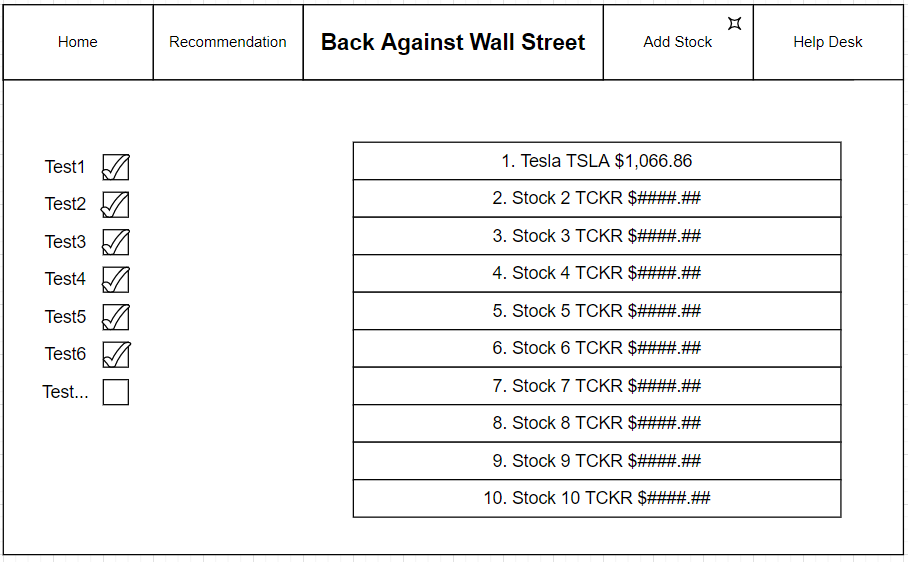
**2.3.2** Sign Up page



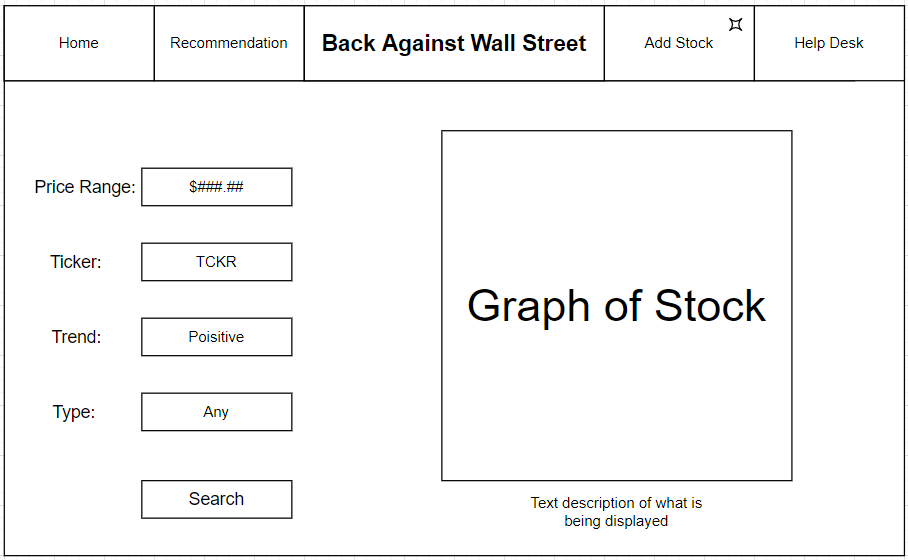
**2.3.3** Credit card information entry page



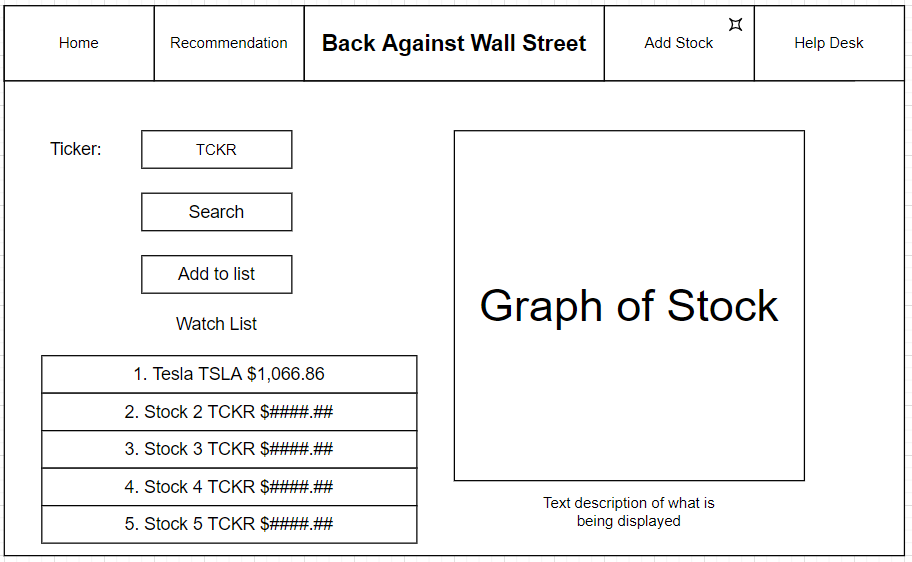
**2.3.4** Home Page



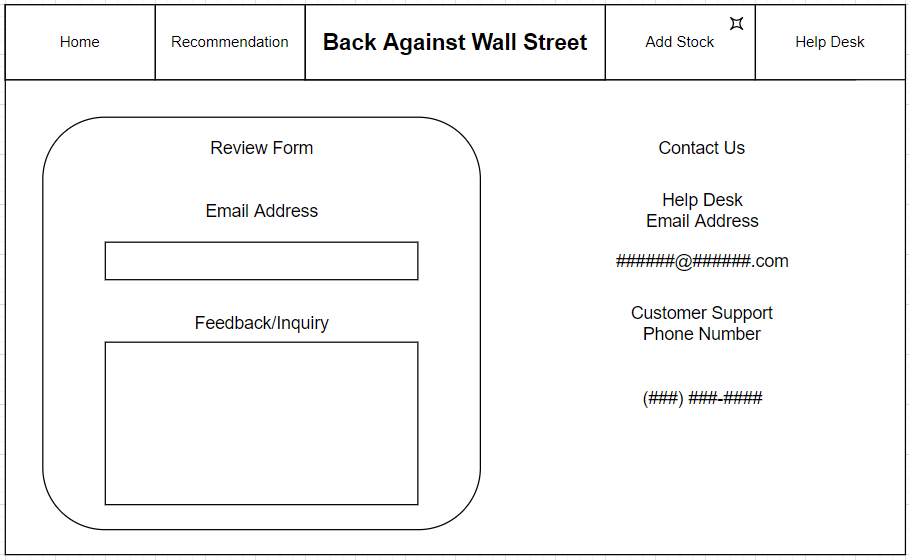
**2.3.5** Stock Recommendation Data Entry Tab



**2.3.6** Add Stock Page



**2.3.7** Help Desk Page



User interface issues

This section will present the main principles of the product's user interface. Use the personas/use cases defined in the SRS to make specific examples. This section should not touch on technical details. You may want to include sketches and specific text messages.

For the Login Page, if the user does not have an account or inputs the wrong information, they will receive an error. Similarly, in the Sign Up Page, if the user inputs the wrong information, they will receive an error. On the Credit Card Information Entry Page, if the credit card the user is signing up with does not have enough funds on it or does not exist, they will receive an error message. On the Stock Addition Tab, if the user inputs a stock that does not exist or that we cannot receive info on, they will receive an error.

# Detailed description of components (ONLY 2 ARE REQUIRED)

## Stock Database

| Identification | Stock Database. This will use pandas to store data in a feather database. |
| --- | --- |
| Type | A database. |
| Purpose | The purpose of the stock database is to store stock data that will be used in calculations and displays in order to give the user an idea of what stock to purchase. |
| Function | The function of the stock database is to store the stock data that will be used in calculations and displays in order to give the user an idea of what stock to purchase. The analyzed stock data will be stored here as well. This includes the price of stock, list of stock, other indicators for each stock, the results of the tests we will run on each stock. |
| Subordinates |  |
| Dependencies | The stock database dependencies are Login page and Signup page for storing the user’s email and password. Credit card information page for storing the user’s card information. home page to display the user’s name. Stock addition tab to add stocks to the database. Stock recommendation data entry tab to store stock price and rating criteria. |
| Interfaces | The stock database will store error messages to show the user if they used the incorrect username/password for logging in. The database will store the user’s information, and display the analyzed stock data, price of stock, list of stock, other indicators for each stock, and the results of the tests we will run on each stock. |
| Resources |  |
| Processing |  |
| Data |  |

## User Interface

| Identification | User Interface. Peruse will be using Flask to display the ui. |
| --- | --- |
| Type | A User Interface. |
| Purpose | The purpose of the user interface is to have the user interact with a clean and easy to understand application. |
| Function | The function of the User interface is so that the user can interact with our application. This can be done by having the user interact with a nice looking login/signup page, home page, Stock Addition page, and a Stock Recommendation Data Entry page. |
| Subordinates |  |
| Dependencies | The user interface dependencies are everything in the front end and having it connected to the backend. This consists of the login/signup pages, home page, Stock Addition page, and a Stock Recommendation Data Entry page. |
| Interfaces | The user interface will display error messages in the login page for misinput. The user interface will display the login/signup pages, home page, Stock Addition page, and a Stock Recommendation Data Entry page. |
| Resources |  |
| Processing |  |
| Data |  |

Examples of a component are:

1. Database
2. Server
3. Client application
4. Search Application
5. Etc.

# Reuse and relationships to other products

For teams doing enhancement work, reuse is an important issue. Most enhancement work should focus on extending, rather than replacing, the design and product development from earlier semesters. For teams doing new development, reuse can also be an important strategy. In some cases, there is freeware that could be incorporated. In other cases, there are existing modules or classes that could be adapted. Another possibility is the use of special tools that produce open source results and thus permissible under the terms of this course.

This section should include the following subsections as appropriate:

* How reuse is playing a role in your product design
* How reuse is playing a role in your product implementation (and the motivation for changes)
* If you are not reusing material that is available, then give motivation for why it is being thrown out.

# Design decisions and tradeoffs

Use this section to motivate any decisions that will help the reader understand the design that your team is using. This section can also capture good ideas that were abandoned and the reasons for leaving them out of the design.

This Design is built around an underlying list of predetermined stocks which will be analyzed instead of analyzing the entire stock market. This design decision was made to work around the constraint on the limit to the number of requests that can be sent to the Yahoo Finance API without invoking cost. This design allows for tabular and graphical representations of the stock data to give the user as many ways as possible to view the stocks and gain insight about them. The Add Stock feature was made to be locked behind a subscription wall. This decision was made to drive the sale of the subscription, and provide an added benefit to the purchase of it.

# Pseudocode for components

Utilize the use cases to create pseudocode for components.

# Appendices (if any)

# Software component template for section 3

The template given below suggests a reasonable structure for giving a thorough description of each component described in Part 3 of the SDS. The specific information depends in part on the design approach. Your team must adapt this template to your needs and describe it in section 3.1 of your SDS.

| Identification | The unique name for the component and the location of the component in the system. |
| --- | --- |
| Type | A module, a subprogram, a data file, a control procedure, a class, etc. |
| Purpose | Function and performance requirements implemented by the design component, including derived requirements. Derived requirements are not explicitly stated in the SRS, but are implied or adjunct to formally stated SDS requirements. |
| Function | What the component does, the transformation process, the specific inputs that are processed, the algorithms that are used, the outputs that are produced, where the data items are stored, and which data items are modified. |
| Subordinates | The internal structure of the component, the constituents of the component, and the functional requirements satisfied by each part. |
| Dependencies | How the component's function and performance relate to other components. How this component is used by other components. The other components that use this component. Interaction details such as timing, interaction conditions (such as order of execution and data sharing), and responsibility for creation, duplication, use, storage, and elimination of components. |
| Interfaces | Detailed descriptions of all external and internal interfaces as well as of any mechanisms for communicating through messages, parameters, or common data areas. All error messages and error codes should be identified. All screen formats, interactive messages, and other user interface components (originally defined in the SRS) should be given here. |
| Resources | A complete description of all resources (hardware or software) external to the component but required to carry out its functions. Some examples are CPU execution time, memory (primary, secondary, or archival), buffers, I/O channels, plotters, printers, math libraries, hardware registers, interrupt structures, and system services. |
| Processing | The full description of the functions presented in the Function subsection. Pseudocode can be used to document algorithms, equations, and logic. |
| Data | For the data internal to the component, describes the representation method, initial values, use, semantics, and format. This information will probably be recorded in the data dictionary. |