

Software Requirements Specification

for

Expense Tracker

Version 1.0

Prepared by

Group Name: <*place your group name here*>

|  |  |  |
| --- | --- | --- |
| Chayse Jorgenson | 11678623 | [Chayse.jorgenson@wsu.edu](mailto:Chayse.jorgenson@wsu.edu) |
| Kevin Tong | <student #> | [Kevin.tong@wsu.edu](mailto:Kevin.tong@wsu.edu) |
| Vitaliy Kudrik | 11603449 | [Vitaliy.kudrik@wsu.edu](mailto:Vitaliy.kudrik@wsu.edu) |

|  |  |
| --- | --- |
|  |  |
| Date: | 10/25/2019 |
|  |  |
|  |  |
|  |  |

Contents

Revisions iii

1 Introduction 1

1.1 Document Purpose 1

1.2 Product Scope 1

1.3 Intended Audience and Document Overview 1

1.4 Definitions, Acronyms and Abbreviations 2

1.5 Document Conventions 2

1.6 References and Acknowledgments 2

2 Overall Description 3

2.1 Product Perspective 3

2.2 Product Functionality 3

2.3 Users and Characteristics 3

2.4 Operating Environment 3

2.5 Design and Implementation Constraints 4

2.6 User Documentation 4

2.7 Assumptions and Dependencies 4

3 Specific Requirements 5

3.1 External Interface Requirements 5

3.2 Functional Requirements 6

3.3 Behaviour Requirements 6

4 Other Non-functional Requirements 7

4.1 Performance Requirements 7

4.2 Safety and Security Requirements 7

4.3 Software Quality Attributes 7

5 Other Requirements 8

Appendix A – Data Dictionary 9

Appendix B - Group Log 10

Revisions

| Version | Primary Author(s) | Description of Version | Date Completed |
| --- | --- | --- | --- |
| 1.0 | Chayse Jorgenson  Kevin Tong  Vitaliy Kudrik | Document Creation | 10/25/19 |

# 

# *<In this template you will find text bounded by the “<>” symbols. This text appears in italics and is intended to provide explanations and guide you through the document. There are two types of comments in this document. The comments that are in black are intended specifically for the course. The comments that are in blue are more general and apply to any SRS. Please make sure to delete all of the comments before submitting the document**.>*

# Introduction

The Expense Tracker 1.0 is a web application to help users analyze their spending habits and set saving goals. The tracker will take user input and analyze that input to display their spending habits for them to reflect on.

*<TO DO: Please provide a brief introduction to your project and a brief overview of what the reader will find in this section.>*

## Document Purpose

This Document will describe in detail, the functional and nonfunctional requirements of the Expense Tracker 1.0. This product will track and anaylze expense data as well as earnings data for the user. This Document will also serve as log to track changes to the project as it evolves.

This document will provide a concrete resource for the developers to look back on as they continue to work on the project and allow them to keep track of the requirements related conversatiosn that occur throughout devlopment. It will also allow for the client and professor to evaluate how well the developers designed a product to match the specifications.

. <Identify the product whose software requirements are specified in this document, including the revision or release number. Describe the scope of the product that is covered by this SRS, particularly if this SRS describes only part of the system or a single subsystem.

TO DO: Write 1-2 paragraphs describing the purpose of this document as explained above.>

## Product Scope

The Expense Tracker 1.0 will take expense data from the user and use the data to provide the user with analytics. The data the user will provide will be expense value, expense category, and date of expense. The program will give analytics such as weekly expenses, monthly expenses, yearly expenses, expenses based on category etc.

The program will also allow for the user to input their earnings and use that to show how much they are making or losing within specified time frames. This will allow for the user to better understand their financial situation and learn how to better reach their financial goals.

<Provide a short description of the software being specified and its purpose, including relevant benefits, objectives, and goals.

TO DO: 1-2 paragraphs describing the scope of the product. Make sure to describe the benefits associated with the product.>

## Intended Audience and Document Overview

This document is intended for the client, the devlopers, and the professor. The client will use this document to ensure that the developers are creating a product that fits their specification. The devlopers will use this document as a reference to ensure that they are creating a qualtiy product that follows the specifications. The Professor will use this document to analyze the students SRS writing abilities and to understand the scope of the project that they will be working on for the rest of the semester.

This document will go over the functional and nonfunction requirements of the the Expense Tracker 1.0 as well as work a as log to keep track of changes to said requirements and the project grows and evolves.

<Describe the different types of reader that the document is intended for, such as developers, project managers, marketing staff, users, testers, and documentation writers (In your case it would probably be the “client” and the professor). Describe what the rest of this SRS contains and how it is organized. Suggest a sequence for reading the document, beginning with the overview sections and proceeding through the sections that are most pertinent to each reader type.>

## Definitions, Acronyms and Abbreviations

This Section does not yet apply because we are not using any acronyms or abbreviations at this time. If acronyms or abbreviations are added throughout the course of development, then we describe them here.

<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.

TO DO: Please provide a list of all abbreviations and acronyms used in this document sorted in alphabetical order.>

## Document Conventions

This Document follows IEEE formatting requirements. It uses Arial font size 11 through the document for text. It uses italacs to denote comments. The document is single spaced with 1” margins. It uses bold font size 14 for the subsection titles and bold font size 18 for the section titles.

<In general this document follows the IEEE formatting requirements. Use Arial font size 11, or 12 throughout the document for text. Use italics for comments. Document text should be single spaced and maintain the 1” margins found in this template. For Section and Subsection titles please follow the template.

TO DO: Describe any standards or typographical conventions that were followed when writing this SRS, such as fonts or highlighting that have special significance. Sometimes, it is useful to divide this section to several sections, e.g., Formatting Conventions, Naming Conventions, etc.>

## References and Acknowledgments

This document does not currently reference any other outside sources.

<List any other documents or Web addresses to which this SRS refers. These may include user interface style guides, contracts, standards, system requirements specifications, use case documents, or a vision and scope document.

TO DO: Use the standard IEEE citation guide (attached) for this section.>

# Overall Description

## Product Perspective

<Describe the context and origin of the product being specified in this SRS. For example, state whether this product is a follow-on member of a product family, a replacement for certain existing systems, or a new, self-contained product. If the SRS defines a component of a larger system, relate the requirements of the larger system to the functionality of this software and identify interfaces between the two. In this part, make sure to include a simple diagram that shows the major components of the overall system, subsystem interconnections, and external interface. In this section it is crucial that you will be creative and provide as much information as possible.

TO DO: Provide at least one paragraph describing product perspective. Provide a general diagram that will illustrate how your product interacts with the environment and in what context it is being used, i.e., context diagram.>

 This product is a brand new webpage driven expense tracker. It has no previous components and is not a follow-on member of any other product family. This expense tracker will be compromised of a website that will allow the user to navigate all the features. These features will be coded using the Javascript language and the webpage will be setup using HTML. The webpage will feature a landing page for the user, so that they can traverse the different features that we will have on separate pages. The user will input income and expense information

on another webpage, and that webpage will also be connected to the

rest of the pages, allowing for easier access to the data they

wish to chart/graph.

## Product Functionality

<Summarize the major functions the product must perform or must let the user perform. Details will be provided in Section 3, so only a high level summary is needed here. Organize the functions to make them understandable to any reader of the SRS. A picture of the major groups of related requirements and how they relate, such as a top level data flow diagram or object class diagram, will be effective.

TO DO:

1. Provide a bulleted list of all the major functions of the system

2. **(Optional)** Provide a Data Flow Diagram of the system to show how these functions relate to each other. This is useful when there is a clear sequence for the functions being performed.>

* Landing page that lets them find the other features
* Income/Expense data reciever
* Income tracker that accepts both the income and expenses and sees if they’re making/losing money and other little things
* Expense tracking function which can show expenses for weekly/monthy/yearly and category specific expenses.

- The landing page allows the user to have ease of access to the other functions without having to traverse a bunch of webages to find exactly what they are looking for.

- The Income/Expense reciever is there so that the user can have a place to input their data, which is used later by the other functions of this product, so that we can provide proper data.

- The income tracker is a function that will add together the income and expenses of the user and will show them visually how their finances are looking in general.

- The expense tracking function will be more detailed and focus primarily on expense data. This function will have the ability to show expenses for multiple time frames and categorical expenses.

## Users and Characteristics

<Identify the various users that you anticipate will use this product. Users may be differentiated based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience.

TO DO:

1. Describe the pertinent characteristics of each user. Certain requirements may pertain only to certain users.

3. Distinguish the most important users for this product from those who are less important to satisfy.>

Our product is geared towards any person/entity that is looking to keep track of their finances and potentially use our software to make a change in that behavior. Our product is geared towards simplicity and ease of access, there are not any functions for accepting giant loads of information and different kinds of incomes, for example derivative options that some might be holding.

Our main user will be the average person who is not part of a huge corporation, that just needs to see where his finances are going wrong and where they can improve. This user has the priority when it comes to our functional design, so that he might be able to access every feature without any expertise on his part. There will be those who use our product for corporations and bigger entity’s, however they are not given priority when it comes to having functionality geared specifically for them, rather they must use our simple functions in their own creative ways to adapt for themselves.

## Operating Environment

<Describe the environment in which the software will operate, including the hardware platform, operating system and versions, and any other software components or applications with which it must peacefully coexist. In this part, make sure to include a simple diagram that shows the major components of the overall system, subsystem interconnections, and external interface

TO DO: As stated above, in at least one paragraph, describe the environment your system will have to operate in. Make sure to include the minimum platform requirements for your system. >

Our product will be running on the computer that contains the files for the webpage. This means that it is portable and although it is not online, it is a website that is usable on the local machine. This means that our product will work in any environment that has access to a browser that access HTML and javascript, internet connection is not required. It also means that operating system will not be a conflict, it will work on Windows, Linux, Unix, etc. This product will basically be able to run as soon as someone clicks on the html file that we will provide.

## Design and Implementation Constraints

<Describe any items or issues that will limit the options available to the developers. These might include: hardware limitations (timing requirements, memory requirements); interfaces to other applications; specific technologies, tools, and databases to be used; parallel operations; language requirements; communications protocols; security considerations; design conventions or programming standards (for example, if the customer’s organization will be responsible for maintaining the delivered software).

TO DO: In this section you need to consider all of the information you gathered so far, analyze it and correctly identify relevant constraints.>

There are no hardware issues that will limit us from producing our product. Our product has no security considerations because it is a locally based product which only the computer user themselves will have access to. There are also no programming language requirements, because this is a webpage based tool, you can include other little scripts and functions from languages like Python or Java, as long as you know how to integrate them with HTML. The only issue would be a language barrier between us and the entire world. This is due to the fact that our product will be produced in English only, and we do not have any support to switch between multiple languages. The users will be responsible for maintaining and upgrading the software as they see fit, we only provide the base version.

## User Documentation

<List the user documentation components (such as user manuals, on-line help, and tutorials) that will be delivered along with the software. Identify any known user documentation delivery formats or standards.

TO DO: You will not actually develop any user-manuals, but you need to describe what kind of manuals and what kind of help is needed for the software you will be developing. One paragraph should be sufficient for this section.>

Our user manual would be rather simple because our product is self-explanitory. The user manual would show how to navigate the webpages. It would show the user how to input their income/expense data, and also how to remove income/expense data they no longer want. It will also show the user how to navigate the different charting/graphing tools that our product provides. All of these functions would be self explanitory due to their naming and general use conventions by the majority of people. A tutorial showing how each function works and how they fit together would also be approriate as seeing someone use the software is a lot easier then reading about how to use it.

## Assumptions and Dependencies

<List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. These could include third-party or commercial components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors, such as software components that you intend to reuse from another project.

TO DO: Provide a short list of some major assumptions that might significantly affect your design. For example, you can assume that your client will have 1, 2 or at most 50 Automated Banking Machines. Every number has a significant effect on the design of your system. >

Our product is intended to be used on a local machine and as such there are no major operating constraints. If a client has multiple machines that they want to run our product on, it would not matter, because our product doesn’t hold any information on a database or is linked in anyway to other versions of itself. Rather it keeps all of the data that is inputted and all the charts it creates in the local folder that it is held in. As for assuming the worse case scenario, it is possible that a customer might have a personal computer without access to any browsers, or they might not be able to use html or javascript based products. The only solution to these issues would be for the user to get the software necessary to run our products, by getting a browser and html/javascript off the web or even obtain them from friends/collegues who would be willing to give them the installation files. If that is not possible, then they would not be able to run our product without a new personal computer or machine that can run html/javascript webpages.

# Specific Requirements

## External Interface Requirements

### User Interfaces

<Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., Cancel) that will appear on every screen, error message display standards, and so on. Define the software components for which a user interface is needed.

TO DO: The least you can do for this section is to describe in words the different User Interfaces and the different screens that will be available to the user. Optional: You may also provide an initial Graphical User Interface design (does not have to be final).>

### Hardware Interfaces

<Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware. You are not required to specify what protocols you will be using to communicate with the hardware, but it will be usually included in this part as well.

TO DO: Please provide a short description of the different hardware interfaces. If you will be using some special libraries to communicate with your software mention them here. In case you have more than one hardware interface divide this section into subsections.>

### Software Interfaces

<Describe the connections between this product and other specific software components (name and version), including databases, operating systems (Windows? Linux? Etc…), tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.

TO DO: The previous part illustrates some of the information you would usually include in this part of the SRS document. To make things simpler, you are only required to describe the specific interface with the operating system.>

### Communications Interfaces

<Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.

TO DO: Do not go into too much detail, but provide 1-2 paragraphs were you will outline the major communication standards. For example, if you decide to use encryption there is no need to specify the exact encryption standards, but rather, specify the fact that the data will be encrypted and name what standards you consider using. >

## Functional Requirements

*< Functional requirements capture the intended behavior of the system. This behavior may be expressed as services, tasks or functions the system is required to perform. This section is the direct continuation of section 2.2 where you have specified the general functional requirements. Here, you should list in detail the different product functions with specific explanations regarding every function.*

*TO DO: Break the functional requirements to several functional areas and divide this section into subsections accordingly. Provide a detailed list of all product operations related to these functional areas.*

## Behaviour Requirements

### Use Case View

<A use case defines a goal-oriented set of interactions between external actors and the system under consideration.

TO DO: Provide a use case diagram which shows the entire system and all possible actors. Do not include detailed use case descriptions (these will be needed when you will be working on the Test Plan), but make sure to include a short description of what every use-case is, who are the actors in your diagram.>

# Other Non-functional Requirements

## Performance Requirements

<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.

TODO: Provide relevant performance requirements based on the information you collected from the client. For example you can say “1. Any transaction will not take more than 10 seconds, etc…>

Our performance requirements are limited to the speed of the users personal computer. This is a lightweight website application that does not require much speed or memory in order to run. There is also no storage in any databases either, rather we just create little files in the repository where the main webpage file is held. There is no connection to a server or the internet, thus there is no lag to be worried about. The only requirements again are hardware requirements on the user’s side, and any PC that can run a browser will be able to run our functions. The only thing the user would need to account for is the memory required to hold a few graphs/charts that are created by our tool.

## Safety and Security Requirements

<Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product’s design or use. Define any safety certifications that must be satisfied. Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements.

TODO:

* Provide relevant safety requirements based on your interview with the client or, on your expectation for the product.
* Describe briefly what level of security is expected from this product by your client and provide a bulleted (or numbered) list of the major security requirements.>

This product has no safety or security requirements from the creators. This is because our product is locally run and not connected to any databases. Our product also does not run on the internet, even though it is a webpage application. As such there is no risk of leaked income information as a result of using of our software. However, the only risk that must be considered is the fact that our files might be found and used by a hacker that might’ve compromised the users PC with means outside of our control. The simple way to avoid this issue is to make sure that the user understands that our tool is fully in their control and as such, all of the information they create using it will be their responsibility to protect and maintain.

## Software Quality Attributes

<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.

TODO: Use subsections (e.g., 4.3.1 Reliability, 4.3.2 Portability, etc…) provide requirements related to the different software quality attributes. Base the information you include in these subsections on the material you have learned in the class. Make sure, that you do not just write “This software shall be maintainable…” Indicate how you plan to achieve it, etc.>

Sections:

### 4.3.1 – Adaptabiltiy/Flexibility: Our product is a combination of HTML and javascript code. This means that the addition of more html webpages and more javascript code is always possible. Our code will be easily readable and well commented so that if someone were to change any of our functions, they could do so with ease.

### 4.3.2 – Availability/Portability: Our product is locally run and not connected to the internet or any databases. As such anyone with access to a browser with the ability to run HTML/Javascript code would have access to our product. This must always be the case, because we are geared towards the average person and not towards big corporations.

### 4.3.3 – Correctness/Reliability: Our product is based on using correct algorithms to provide our users with charts/graphs that show them how they are spending their money. This means that our algorithms have to be fool-proof and work within reasonable Big-O time complexity.

### 4.3.4 – Maintainability/Robustness: Our product is very simple and as such there is no need to have it maintained as the laws of addition and subtraction are not going to change. The only maintainace needed would come as a result of the user having made their own changes that might need to be revised.

### 4.3.5 – Usability: Usability is very important to our product. It is a product geared towards the average person and as such each function must come with a reasonable name and a clear description. If a function is unavoidably complex, then there must be a tutorial or some kind of informative tool that will allow the user to understand how to use the function without understanding the intricities of the function.

### 4.3.6 – Testabiltiy: Our code will be tested using minimalistic use cases as well as excessively large cases to make sure that our product can withstand anything our user will throw at it. The code itself is also in a separate javascript file, and as such, our users can go in and test each individual algorithm personally and as they see fit. Thus our testability is very diverse and doesn’t allow many bugs to come through.

# Other Requirements

<This section is **Optional.** Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>

Appendix A – Data Dictionary

*<Data dictionary is used to track all the different variables, states and functional requirements that you described in your document. Make sure to include the complete list of all constants, state variables (and their possible states), inputs and outputs in a table. In the table, include the description of these items as well as all related operations and requirements.>*

Appendix B - Group Log

<Please include here all the minutes from your group meetings, your group activities, and any other relevant information that will assist the Teaching Assistant to determine the effort put forth to produce this document>