**Software Requirements Specification**

**for**

Money Gremlin

**Version 1.5 approved**

**Prepared by Eric Jestel, Saurav Pokhrel, Joshua Wilbur, Siddhartha Bajracharya**

**COS 420, Fall 2024**

**October 3rd, 2024**

**Table of Contents**

**Table of Contents ii**

**Revision History ii**

**1. Introduction 1**

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 1

1.5 References 1

**2. Overall Description 2**

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Classes and Characteristics 2

2.4 Operating Environment 2

2.5 Design and Implementation Constraints 2

2.6 User Documentation 2

2.7 Assumptions and Dependencies 3

**3. External Interface Requirements 3**

3.1 User Interfaces 3

3.2 Hardware Interfaces 3

3.3 Software Interfaces 3

3.4 Communications Interfaces 3

**4. System Features 4**

4.1 System Feature 1 4

4.2 System Feature 2 (and so on) 4

**5. Other Nonfunctional Requirements 4**

5.1 Performance Requirements 4

5.2 Safety Requirements 5

5.3 Security Requirements 5

5.4 Software Quality Attributes 5

5.5 Business Rules 5

**6. Other Requirements 5**

**Appendix A: Glossary 5**

**Appendix B: Analysis Models 5**

**Appendix C: To Be Determined List 6**

**Revision History**

| **Name** | **Date** | **Reason For Changes** | **Version** |
| --- | --- | --- | --- |
| Eric | 10/2 | Added functional and non-functional requirements | 1.0 |
| Josh | 10/3 | Added 7 functional requirements and 3 nonfunctional requirements. | 1.1 |
| Saurav | 10/3 | Added 7 functional requirements and 3 nonfunctional requirements. | 1.2 |
| Eric | 10/8 | Added link to mockups in 3.1 | 1.3 |
| Siddhartha | 10/8 | Added 7 functional requirements and 3 nonfunctional requirements. | 1.4 |
| Team | 10/10 | Revised document and added title page | 1.5 |

# **Introduction**

## **Purpose**

*<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.>*

## **Document Conventions**

*<Describe any standards or typographical conventions that were followed when writing this SRS, such as fonts or highlighting that have special significance. For example, state whether priorities for higher-level requirements are assumed to be inherited by detailed requirements, or whether every requirement statement is to have its own priority.>*

## **Intended Audience and Reading Suggestions**

*<Describe the different types of reader that the document is intended for, such as developers, project managers, marketing staff, users, testers, and documentation writers. 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.>*

## **Product Scope**

*<Provide a short description of the software being specified and its purpose, including relevant benefits, objectives, and goals. Relate the software to corporate goals or business strategies. If a separate vision and scope document is available, refer to it rather than duplicating its contents here.>*

## **References**

*<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. Provide enough information so that the reader could access a copy of each reference, including title, author, version number, date, and source or location.>*

# **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. A simple diagram that shows the major components of the overall system, subsystem interconnections, and external interfaces can be helpful.>*

## **Product Functions**

*<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 (such as a bullet list) 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, is often effective.>*

## **User Classes and Characteristics**

*<Identify the various user classes that you anticipate will use this product. User classes may be differentiated based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience. Describe the pertinent characteristics of each user class. Certain requirements may pertain only to certain user classes. Distinguish the most important user classes for this product from those who are less important to satisfy.>*

## **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.>*

## **Design and Implementation Constraints**

*<Describe any items or issues that will limit the options available to the developers. These might include: corporate or regulatory policies; 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).>*

## **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.>*

## **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, unless they are already documented elsewhere (for example, in the vision and scope document or the project plan).>*

# **External Interface Requirements**

## **User Interfaces**

<https://docs.google.com/presentation/d/1yvxT47wots3ccU64k7fqQTBp045WM9lHmpsLHmnby3c/edit#slide=id.p>

*<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., help) that will appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.>*

## **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, and communication protocols to be used.>*

## **Software Interfaces**

*<Describe the connections between this product and other specific software components (name and version), including databases, operating systems, 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. Refer to documents that describe detailed application programming interface protocols. 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.>*

## **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.>*

# **System Features**

*<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>*

## **System Feature 1**

*<Don’t really say “System Feature 1.” State the feature name in just a few words.>*

4.1.1 Description and Priority

*<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>*

4.1.2 Stimulus/Response Sequences

*<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>*

4.1.3 Functional Requirements

*<Itemize the detailed functional requirements associated with this feature. These are the*

*software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use “TBD” as a placeholder to indicate when necessary information is not yet available.>*

REQ-1: The system shall allow the user to create spending categories.

REQ-2: The system shall let the user define a monthly spending limit.

REQ-3: The system shall provide a visualization of the user’s spending.

REQ-4: The system shall allow the user to make a new account with an email and

password.

REQ-5: The system shall let the user update their email and password.

REQ-6: The system shall allow the user to input transactions.

REQ-7: The system shall allow the user to export and download their transaction log.

REQ-8: The system shall allow the user to log in using their account credentials.

REQ-9: The system shall allow the user to set custom notification settings for when they approach the spending limit.

REQ-10: The system shall allow the user to categorize each transaction. (ie. food, gas)

REQ-11: The system shall provide budgeting recommendations based on the user’s spending trends.

REQ-12: The system shall allow the user to undo a transaction for accidental transaction entries.

REQ-13: The system shall notify the user when the user has exceeded their budget.

REQ-14: The system shall allow the user to edit transaction amounts.

REQ-15: The system shall allow the user to assign a transaction to only 1 spending category

REQ-16: The system shall track the date, amount, and category of each transaction.

REQ-17: The system shall predict a rate of spending based on the transaction history.

REQ-18: The system shall report the predicted rate and the predicted date when spending will exceed budget to the user.

REQ-19: The system shall allow the user to set a spending limit for each category.

REQ-20: The system shall allow the user to set spending warnings at different dollar values or percentages of the spending limit.

REQ-21: The user can attach a memo/note to a transaction

REQ-22: The system shall allow the user to reset their password using the email associated with the account.

REQ-23: The system shall allow the user to log out of the system.

REQ-24: The system shall allow the user to delete their account.

REQ-25: The system shall allow the user to view transactions by date, category, or amount.

REQ-26: The system shall allow the user to schedule recurring transactions such as monthly bills, subscriptions, or income and automatically apply them to the budget.

REQ-27: The user can select a profile picture

REQ-28: The system shall provide a financial summary that includes their total income, total expense, and their expenditure breakdown.

REQ-29: The system shall allow the user to change their profile picture.

REQ-30: The system shall provide a search bar to find specific transactions

## **System Feature 2 (and so on)**

# **Other Nonfunctional 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.>*

PERF-1: The system shall respond to user input within 1 second, 99% of the time.

PERF-2: The system shall update the transaction log, within 2 seconds of the user inputting a new transaction, 95% of the time.

PERF-3: The system shall decrease the available monthly spending limit after the user inputs a new transaction within 2 seconds, 95% of the time.

PERF-4: The system shall remain active 99% of the time between 8am and midnight.

PERF-5: The system shall remain active 95% of the time between midnight and 8am.

PEFR-6: The system shall display visual alerts or error messages for failed operations within 1 second of detecting the issue, 99% of the time.

PEFR-7: The system shall load the user’s profile page within 2 seconds, 95% of the time.

## **Safety 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.>*

SAFE-1: The system shall encrypt all sensitive user data.

SAFE-2: The system shall incorporate a “forgot password” feature.

SAFE-3: The system shall require an additional authentication to access sensitive financial information.

## **Security Requirements**

*<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. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.>*

SEC-1: The system shall operate within GDPR and CCPA regulations to safeguard the user’s data.

SEC-2: The system shall store only the hash of the password.

SEC-3: The system shall securely encrypt the hash table and salt it.

SEC-4: The system shall comply with the OWASP Top 10 Security Standards to mitigate common security vulnerabilities and ensure secure handling of user data.

SEC-5: The system shall comply with PCI DSS for processing, storing, and transmitting credit card information to ensure data protection and regulatory compliance.

## **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.>*

## **Business Rules**

*<List any operating principles about the product, such as which individuals or roles can perform which functions under specific circumstances. These are not functional requirements in themselves, but they may imply certain functional requirements to enforce the rules.>*

# **Other Requirements**

*<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: Glossary**

*<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.>*

**Appendix B: Analysis Models**

*<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams*.>

**Appendix C: To Be Determined List**

*<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>*