**Software Requirements Specification**

**for**

**<Project>**

**Version 1.0 approved**

**Prepared by <authors>**

**<organization>**

**<date created>**

**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** |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |

# **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**

*<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.>*

A diagram of the UI mockup can be found here: <https://miro.com/app/board/uXjVPT8jhWE=/>

The standard buttons which will appear on each screen along the top of the UI are:

* **Memento:** The name of the app and serves as the home button, taking the user to the initial screen where they can create new cards.
  + It's function as a home button is suggested by being written with larger font than the other buttons while being positioned alongside them.
* **Library**: Takes the user to a list of their sets. From here they can add existing cards to a set.
* **Quiz**: Takes the user to the list of their card sets where they can select one to begin studying.
* **Share**: Lets the user produce a text document storing their cards and associated metadata to an email address or save it locally.

Eventually UIs may be needed for the following software components:

* Adding new cards
  + Selecting sets to add the cards to
* Viewing existing sets
  + Adding a new set
  + Moving cards from the collection to a chosen set
* Quizzing cards
  + Before and after flipping the card over
  + Rating confidence on a card
* Sharing cards
  + By email
  + Locally

## **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.>*

## Importing Notes

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).>*

Ensure users can upload or paste in their notes to the app and have them converted to flash cards. This is high priority since it’s required for the app to function.

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.>*

When the user opens the app, the main screen shows the input field for notes and a button which the user may press to convert any entered text into flash cards.

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.>*

*<Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.>*

REQ-1: The flashcard creation system shall be able to parse text documents given some text delimiter(s)

REQ-2: The flashcard creation system shall allow users to use certain flashcard templates

REQ-3:The flashcard creation system shall default to present users with an empty notes field

REQ-4: The flashcard creation system will allow the user to set custom delimiters to differentiate the front and rear of a flashcard

## Creating Decks

4.2.1 Description and Priority

Allow the user to assign created cards to decks so that they can organize information belonging to different subjects. As this improves the studying experience but isn’t necessary for baseline use, it’s medium priority.

4.2.2 Stimulus/Response Sequences

After cards are created the user is given the option to assign them to sets. Choosing to do so takes them to an interface which allows assigning the new cards to sets. This can also be done by going to the library menu and selecting a set.

4.2.3 Functional Requirements

REQ-5: The flashcard creation system will allow users to add new or existing cards to one or more sets of cards, or decks, for organization.

REQ-6: The flashcard creation system will automatically create new sets as required when importing shared flashcards, based on recorded information of what previous decks they belonged to.

## Reviewing Cards

4.3.1 Description and Priority

Allow the user to review cards in a selected set. This is a required feature for the baseline functions of the app, so it should have high priority.

4.3.2 Stimulus/Response Sequences

When the user presses the menu button “Quiz”, theyre shown a selection of decks. By picking a deck, the user begins the cycle of being shown a card, flipping it to see the answer, and progressing through sequential cards in the set. This may include a step where they report their confidence on the card to influence card scheduling.

4.3.3 Functional Requirements

REQ-7: The flashcard creation system shall allow users to rate their comfort level with individual cards

REQ-8: The flashcard creation system shall allow users to make notes or comments on cards or decks

REQ-9: The flashcard creation system shall be able to email the user reminders for studying

REQ-10: The flashcard creation system shall track statistics about user performance for individual cards and decks

REQ-11: The flashcard creation system shall implement scheduling of individual cards based on performance or user confidence level

## Saving Cards

4.4.1 Description and Priority

The user can save their cards and associated metadata, such as scheduling information or notes and associated decks, to the local directory. This allows cards to be reimported at a later date or to serve as a backup. This is a quality-of-life feature, and should have low priority.

4.4.2 Stimulus/Response Sequences

When the user clicks the “Share” button, they will be given the option to have this file emailed to a provided address or saved locally.

4.4.3 Functional Requirements

REQ-12: The data synchronization system shall allow users to access flashcards from a variety of computers.

REQ-13: The flashcard creation system shall allow users to use the app without the need of signing up

REQ-14: The flashcard creation system will allow the user to save flashcards to their computer to use later

REQ-15: The flashcard creation system will include metadata such as associated sets and notes when saving or sharing cards.

# **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.>*

1. The flashcard creation system shall allow users to create decks of up to 100 cards
   1. Rationale: Decks need to be able to store enough information that users can’t simply memorize all of the cards and to increase the intervals between repetition.
2. The flashcard creation system shall allow users to create up to 20 decks
   1. Rationale: Users may need to study material from different subjects, each represented by it's own set of cards.
3. The flashcard creation system shall take no more than 1 second to load and display the next card
   1. Rationale: The fast/smooth transition between subsequent cards makes the studying process faster and less frustrating
4. The app shall crash no more than once per 10 hours of use.
   1. Rationale: The app crashing may disrupt study sessions and dissuade use of the app. Minimizing it would help avoid these issues.
5. The flashcard creation system shall be capable of loading the home screen within 4 seconds of start up
   1. Rationale: Quick startup times serves as a good first impression for the app and helps ensure users aren’t frustrated when starting their study sessions.
6. The flashcard creation system shall be capable of storing 250 flashcards per user
   1. Rationale: The user should be able to store enough cards to make multiple decks of maximum size.
7. The flashcard creation system shall be capable of functioning without internet access
   1. Rationale: Studying cards offline increases the amount of time that the app is available to the user, also providing them a productive way to spend time when unable to access the internet.
8. The flashcard creation system will take no longer than 1 minute to save flashcard information to the computer once the user has clicked "save"
   1. Rationale: Serves as a quality of life requirement limiting wait times
9. The flashcard creation system shall allow users to write up to 100 words words per flashcards
10. Rationale: Larger word allowances allow more flexibility in what information the user chooses to use when studying.

## **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.>*

1. The flashcard creation system will be robust to any input text from notes, such that they don’t crash the app.
   1. Safeguards: Include error handling in the code, consider and account for various common errors with text parsing such as unexpected end-of-file issues.
2. The flashcard creation system will not write to any directories other than the one created for it.
   1. Safeguards: Set up any directory pathways within the code to start with a constant string, pointing to the local directory. Ensure that when installed, and possible when run, that the app is in a directory created for its use.

## **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.>*

1. The flashcard creation system shall keep user data and created cards private by default
   1. Sharing cards should require direct action from the user, who is authorized by accessing the device. The user could potentially password-protect their data without the need of formal online account.
2. The flashcard creation system shall function without the need for users to create accounts
   1. User authentication could be done by local password protection, or simply by having access to the device.

## **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.>*