**Function Design**

Subtitle

Author: Your Name

Date: 2023-12-18

Contents

[Function Design 3](#_Toc256000000)

[Overview 3](#_Toc256000001)

[Purpose 3](#_Toc256000002)

[Significance 3](#_Toc256000003)

[Functionality 3](#_Toc256000004)

[Function Requirement 4](#_Toc256000005)

[Functional Requirements 4](#_Toc256000006)

[Inputs, Outputs, and Side Effects 5](#_Toc256000007)

[Boundaries and Constraints 5](#_Toc256000008)

[User Interface 5](#_Toc256000009)

[Key Elements of UI 5](#_Toc256000010)

[Data Definitions 7](#_Toc256000011)

[Input and Output Specification 7](#_Toc256000012)

[Generate Requirement Document Function 7](#_Toc256000013)

[List History Function 8](#_Toc256000014)

[Algorithm 9](#_Toc256000015)

[Error Handling 9](#_Toc256000016)

[Overview 9](#_Toc256000017)

[Error Types and Responses 9](#_Toc256000018)

[Handling Mechanisms 11](#_Toc256000019)

[Performance Consideration 11](#_Toc256000020)

[Time Complexity 11](#_Toc256000021)

[Space Complexity 12](#_Toc256000022)

[Potential Bottlenecks 12](#_Toc256000023)

[Optimization Strategies 12](#_Toc256000024)

[Security Consideration 13](#_Toc256000025)

[Authentication and Authorization 13](#_Toc256000026)

[Data Encryption 13](#_Toc256000027)

[Audit Logging 13](#_Toc256000028)

[Data Minimization and Anonymization 13](#_Toc256000029)

[Access Controls for Document History 13](#_Toc256000030)

[Integrity Checks 14](#_Toc256000031)

[Confidentiality and Data Retention 14](#_Toc256000032)

[Regular Security Assessments 14](#_Toc256000033)

[Test Cases 14](#_Toc256000034)

[System Engineer inputs a description to generate a Requirement Document 14](#_Toc256000035)

[System Engineer tries to generate a document with an empty description 15](#_Toc256000036)

[User lists all generated documents by clicking the “History” button 15](#_Toc256000037)

[User accesses "History" with no generated documents 15](#_Toc256000038)

[User accesses "History" when there is a very large number of generated documents 16](#_Toc256000039)

# Function Design

This is a SAAS product for the System Engineer. The System Engineer will input some description. The product will use the latest AI technology to analysis the description. Then the product will generate one Requirement Document according to AI analysis data. Now, there is a new function that use can list all the generated documents. Users click the “History” button. The product will list all the generated files include the file name, generated time etc.

## Overview

### Purpose

The function aims to streamline the requirement documentation process for System Engineers by utilizing the latest AI technology to create Requirement Documents based on input descriptions. Additionally, the function provides a history feature that enables users to view and access a list of all previously generated documents.

### Significance

* **Enhancement of Efficiency**: Automating the process of Requirement Document generation from descriptions saves time and reduces manual effort for System Engineers.
* **AI-Driven Analysis**: Leveraging cutting-edge AI technology ensures that the analysis of the input descriptions is comprehensive, leading to high-quality Requirement Documents.
* **Document Management**: The history feature simplifies document tracking and organization, allowing users to quickly find and reference past requirement documents.

### Functionality

1. **Requirement Document Generation**
   * Input: System Engineer provides the description of the system or feature.
   * Process: The AI technology conducts an analysis of the input description.
   * Output: A Requirement Document is generated based on AI analysis data.
2. **History Listing**
   * Interaction: User clicks the "History" button within the application interface.
   * Display: A list of all previously generated Requirement Documents is presented, including details such as file name, generation time, and other relevant metadata.

## Function Requirement

### Functional Requirements

#### 1. Description Input and Analysis

* **Requirement 1.1:** The system must provide the user with an interface to input textual descriptions.
* **Requirement 1.2:** The system must use AI technology to analyze the input descriptions.
* **Requirement 1.3:** The system must ensure the AI analysis process adheres to current technological standards for accuracy and performance.

#### 2. Document Generation

* **Requirement 2.1:** The system must generate a Requirement Document based on the AI analysis data.
* **Requirement 2.2:** The system must save the Requirement Document with a unique reference for future retrieval.
* **Requirement 2.3:** The system must provide the user the option to download or directly view the generated Requirement Document.

#### 3. Document History Retrieval

* **Requirement 3.1:** The system must provide a "History" button or interface element for users to view the document generation history.
* **Requirement 3.2:** The system must list all previously generated documents upon request.
* **Requirement 3.3:** For each document listed, the system must display the file name, generated time, and other relevant metadata.
* **Requirement 3.4:** The system must ensure that the history data is up-to-date and reflects the most recent document generation activities.

### Inputs, Outputs, and Side Effects

#### Inputs

* User-provided textual descriptions.
* User command to trigger the document generation process.
* User request to view document generation history.

#### Expected Outputs

* A generated Requirement Document based on AI analysis.
* A list of generated documents, including file names, generated times, and other metadata.

#### Side Effects

* Storage of generated documents and their associated metadata.
* Potential long-term impact on storage depending on the number of documents and frequency of generation.

### Boundaries and Constraints

* The AI analysis must complete within a reasonable time frame to ensure user satisfaction.
* The system should be able to handle a high volume of documents while maintaining performance.
* Document generation history must comply with data privacy and protection regulations.
* The user interface elements must be accessible and follow the best usability practices.

## User Interface

### Key Elements of UI

#### 1. History Button

* **Element Type:** Button
* **Label:** "History"
* **Location:** Accessible from the main dashboard or toolbar
* **Function:** When clicked, it triggers the display of the Document History pane.

#### 2. Document History Pane

* **Element Type:** Pane/Modal/Section
* **Function:** Displays a list of all previously generated Requirement Documents.

#### 3. Document List

* **Element Type:** List/Table
* **Data Definition:**
  + File Name: String - The name of the generated document.
  + Generated Time: DateTime - The exact date and time when the document was generated.
  + Actions: Array of Buttons - Available actions for each document (e.g., View, Download, Delete).

#### 4. View Button (Part of Actions in Document List)

* **Element Type:** Button
* **Label:** "View"
* **Function:** Opens the selected document for viewing.

#### 5. Download Button (Part of Actions in Document List)

* **Element Type:** Button
* **Label:** "Download"
* **Function:** Allows the user to download the selected document.

#### 6. Delete Button (Part of Actions in Document List)

* **Element Type:** Button
* **Label:** "Delete"
* **Function:** Removes the selected document from history after confirmation.

#### 7. Refresh List Button

* **Element Type:** Button
* **Label:** "Refresh"
* **Function:** Refreshes the Document List to show the most recent changes.

#### 8. Close Pane Button

* **Element Type:** Button
* **Label:** "Close" or "X"
* **Function:** Closes the Document History pane.

#### 9. Empty State Message

* **Element Type:** Text
* **Function:** Displays a friendly message when no documents are found in history.
* **Message:** "No historical documents found. Start by generating a new Requirement Document!"

### Data Definitions

| **Element** | **Type** | **Description** |
| --- | --- | --- |
| History Button | Button | Triggers display of the Document History pane. |
| Document List | List/Table | Shows all generated documents with details. |
| File Name | String | The name of the Requirement Document. |
| Generated Time | DateTime | Timestamp of when the document was created. |
| View Button | Button | Opens the document for viewing. |
| Download Button | Button | Initiates download of the document. |
| Delete Button | Button | Initiates deletion process for a document. |
| Refresh List Button | Button | Refreshes the list of documents. |
| Close Pane Button | Button | Closes the Document History pane. |
| Empty State Message | Text | Message when no documents are in history. |

## Input and Output Specification

### Generate Requirement Document Function

#### Input Specification

* **DescriptionText** (String): A textual description provided by the system engineer that the SAAS product will analyze. The text should follow these constraints:
  + UTF-8 encoding
  + Maximum length: Defined by the system limitations or user subscription tier
  + It may include technical terminology that the AI should understand to properly analyze.

#### Output Specification

* **RequirementDocument** (Object):
  + **filename** (String): The name of the generated requirement document file.
    - Format: 'Requirement\_<Timestamp>.pdf' where <Timestamp> is replaced by the document generation timestamp.
  + **content** (Binary): The content of the document in PDF format.
  + **metadata** (Object):
    - **generatedTime** (DateTime): The UTC timestamp when the document was generated.
      * Format: ISO 8601 (YYYY-MM-DDTHH:mm:ssZ)
    - **wordCount** (Integer): The number of words in the generated document.
    - **pageCount** (Integer): The number of pages in the generated document.

### List History Function

#### Input Specification

* **HistoryButton** (Event): A user interface event triggered by the user clicking the “History” button.

#### Output Specification

* **DocumentList** (Array of Object):
  + Each Object in the array represents a generated document and contains the following properties:
    - **filename** (String): The name of the generated requirement document file.
      * Format: Same as the format described in the RequirementDocument output.
    - **generatedTime** (DateTime): The timestamp when the document was generated.
      * Format: Same as the format described in the RequirementDocument output.
    - Additional file properties can be specified as needed (e.g., fileSize, creator, etc.).

## Algorithm

### Algorithm:

1. Initialize the function ListGeneratedDocuments which will be triggered upon the user clicking the "History" button.
2. Retrieve the current user's identity to ensure that the listing contains only the documents that the user has permission to see.
3. Fetch the document metadata from the database where generated documents are stored. This metadata includes file names, generation timestamps, user identifiers, and any other relevant data typically associated with generated documents.
4. Sort the retrieved metadata list by generation timestamp, with the most recent documents appearing first.
5. Format the list into a human-readable structure including necessary details such as file name and generated time.
6. Send the formatted list back to the user's interface.
7. Display the list in a clear and navigable format on the user interface, allowing the user to select and potentially open any document from the history list.

### Illustration (DOT Language):

digraph ListGeneratedDocuments {

node [shape=box];

Start -> "User Clicks 'History'" [label="Trigger"];

"User Clicks 'History'" -> "Retrieve UserID" [label="User Action"];

"Retrieve UserID" -> FetchDocuments [label="Get User ID"];

FetchDocuments -> SortDocuments [label="Fetch Metadata from DB"];

SortDocuments -> FormatList [label="Sort by Time"];

FormatList -> SendList [label="Format Data"];

SendList -> "Display List" [label="Send to UI"];

"Display List" -> End [label="Render on Screen"];

}

## Error Handling

### Overview

The Document History Retrieval function allows System Engineers using the SAAS product to list all generated Requirement Documents by clicking the “History” button. It displays metadata such as file names, generated times, etc.

### Error Types and Responses

The following are the anticipated errors that the function may encounter and the corresponding error responses:

1. **Network Error**
   * Description: Error while trying to retrieve document history due to a network issue.
   * Error Message: "Failed to load history. Please check your network connection and try again."
2. **Authentication Error**
   * Description: User session is not authenticated or the session has expired.
   * Error Message: "You have been logged out. Please log in again to view document history."
3. **Authorization Error**
   * Description: User does not have permission to access the history of documents.
   * Error Message: "You do not have permissions to view the history. Contact your administrator."
4. **No History Found**
   * Description: No generated documents are present in the user's history.
   * Error Message: "No document history found. Start by creating new documents."
5. **Service Unavailable**
   * Description: The SAAS product's service or specific functionality is temporarily unavailable.
   * Error Message: "The history service is currently unavailable. Please try again later."
6. **Timeout Error**
   * Description: The request to retrieve document history took too long to respond.
   * Error Message: "Request timed out. Please try retrieving the documents again."
7. **Internal Server Error**
   * Description: A generic error message for any unexpected server-side error.
   * Error Message: "An error occurred while processing your request. Please try again."
8. **Data Fetch Error**
   * Description: An error occurred while fetching the document data from the database.
   * Error Message: "An error occurred while accessing document data. Please refresh or try again later."
9. **Unexpected Client-Side Error**
   * Description: General client-side error for any unspecified issues.
   * Error Message: "An unexpected error occurred. If the problem persists, contact support."

### Handling Mechanisms

* All errors will log detailed information for debugging purposes, but only a user-friendly error message will be presented to the end user.
* Network and timeout errors will offer a retry option to the user alongside the error message.
* Authentication and authorization errors will prompt the user to log in again or contact their administrator respectively, with a redirect provided to the login page or help center.
* In the case of service unavailability, internal server, and unexpected errors, the user will be encouraged to try again later, and additional support contact information may be displayed.
* For no history found, the system should guide the user to start creating documents or provide a link to help them begin.

## Performance Consideration

### Time Complexity

* **AI Analysis**: The time complexity of the AI analysis process should be considered as it can vary depending on the complexity of the algorithms used and the length of the user's input description. It is essential to use efficient AI models to minimize response times.
* **Requirement Document Generation**: The process of generating a requirements document from the analyzed data should be optimized for speed, especially if templating systems are used.
* **Document Listing (History)**: The function to list all generated documents should be optimized for quick retrieval. The time complexity can be dependent on the data structure used for storing document metadata. Ideally, retrieving the document history should be O(1) if indexed correctly or O(n) when a full scan is required if indices are not used.

### Space Complexity

* **Stored Documents**: Depending on the number of documents generated, the system should be designed to efficiently handle and store a potentially large amount of data without significantly impacting performance. Consideration should be given to the growth of the dataset and how it can be managed, possibly using techniques such as pagination or lazy loading.
* **AI Models**: The space complexity of AI models should be examined, as large models may require significant memory and storage resources.

### Potential Bottlenecks

* **Database Access**: Frequent read/write operations to the database when generating and listing documents can become a bottleneck. Caching frequently accessed data can alleviate database load.
* **AI Processing Overhead**: The AI analysis task could become a bottleneck if not properly managed, especially if the system has a high volume of concurrent users. Asynchronous processing and load balancing might be required to maintain performance.
* **Network Latency**: For a SaaS product, network latency can impact the responsiveness of the system, particularly when transferring large documents or when the service relies on external APIs for AI processing.

### Optimization Strategies

* Implement caching mechanisms to reduce database load and speed up document listing.
* Use efficient data structures and algorithms to manage and retrieve documents, favoring those with lower time complexities for search and retrieval operations.
* Minimize the size of AI models while ensuring acceptable accuracy, to reduce memory footprint and storage requirements.
* Consider implementing a queuing system for AI analysis tasks to manage load and ensure scalability.
* Use content delivery networks (CDNs) or geographically distributed services to minimize network latency for users.

## Security Consideration

### Authentication and Authorization

* Implement multi-factor authentication (MFA) to ensure that only authorized system engineers can access their document history.
* Define strict role-based access controls (RBAC) to prevent unauthorized users from accessing document generation history that they do not own.

### Data Encryption

* Ensure that all data transfer between the client and server is encrypted using TLS/SSL to prevent man-in-the-middle (MITM) attacks.
* Stored documents and any temporary data should be encrypted at rest using industry-standard encryption algorithms to protect against data breaches.

### Audit Logging

* Maintain detailed audit logs for all user actions related to the document generation and access history, which includes file names, generated time, user actions, timestamps, and IP addresses.
* Regularly review audit logs to detect and respond to any unauthorized access or suspicious activities.

### Data Minimization and Anonymization

* Display minimal, necessary details in the history view to avoid exposing sensitive data. Anonymize any data when possible.
* Implement automatic redaction of sensitive information from Requirement Documents before listing them in history if the AI identifies any such data during analysis.

### Access Controls for Document History

* Implement granular access control for viewing generated documents, ensuring users can only list the history of the documents that they have generated or have been given permission to view.
* Set up automatic session timeouts to reduce the risk of unauthorized access in case of user inactivity.

### Integrity Checks

* Include checksums or hash functions to verify the integrity of documents when listed in the history to ensure they have not been tampered with.
* Verify the integrity of the generated Requirement Document before presenting it to users to prevent data corruption or tampering.

### Confidentiality and Data Retention

* Define a strict data retention policy to determine how long generated documents are kept and when they should be securely deleted.
* Provide users the ability to manually delete their generated documents from the history, ensuring confidentiality and control over their data.

### Regular Security Assessments

* Conduct regular security assessments and penetration testing to evaluate the effectiveness of security measures in place, focusing on the document history-listing feature.
* Update the security measures based on the findings of security assessments to deal with emerging threats and vulnerabilities.

## Test Cases

### System Engineer inputs a description to generate a Requirement Document

Acceptance Criteria:

Given: the System Engineer is logged into the SAAS product

When: the System Engineer inputs a valid description into the system

Then: the SAAS product uses AI technology to analyze the description

And: the SAAS product generates a Requirement Document based on the AI analysis data

### System Engineer tries to generate a document with an empty description

Acceptance Criteria:

Given: the System Engineer is logged into the SAAS product

When: the System Engineer attempts to input an empty description

Then: the SAAS product does not proceed with AI analysis

And: the SAAS product displays an error message indicating that the description cannot be empty

### User lists all generated documents by clicking the “History” button

Acceptance Criteria:

Given: the User is logged into the SAAS product

When: the User clicks the “History” button

Then: the SAAS product lists all generated documents

And: each entry in the list shows the file name and generated time

### User accesses "History" with no generated documents

Acceptance Criteria:

Given: the User is logged into the SAAS product

And: there are no generated documents in the User's history

When: the User clicks the “History” button

Then: the SAAS product displays a message indicating there are no generated documents to list

### User accesses "History" when there is a very large number of generated documents

Acceptance Criteria:

Given: the User is logged into the SAAS product

And: there is a very large number of generated documents in the User's history

When: the User clicks the “History” button

Then: the SAAS product lists the generated documents in a paginated view

And: the User can navigate through different pages to view all documents