Software Requirements Specification

Version 1.1

<<Annotated Version>>

June 12, 2020

Virtual Assistant System

Bakri Alkhateeb

Muhammed Alkhattab

Submitted in fulfillment

Of the requirements of

Software Engineering Course

# Table of Contents

[Table of Contents ii](#_Toc42900773)

[List of Figures ii](#_Toc42900774)

[1.0. Introduction 1](#_Toc42900775)

[1.1. Purpose 1](#_Toc42900776)

[1.2. Scope of Project 1](#_Toc42900777)

[1.3. Glossary 2](#_Toc42900778)

[1.4. References 2](#_Toc42900779)

[1.5. Overview of Document 2](#_Toc42900780)

[2.0. Overall Description 3](#_Toc42900781)

[2.1 System Environment 3](#_Toc42900782)

[2.2 Functional Requirements Specification 4](#_Toc42900783)

[2.2.1 Use case: Fill Personal Information 4](#_Toc42900784)

[2.2.2 Use case: Voice Input 5](#_Toc42900785)

[2.2.3 **Use case: Alarm Queries** 6](#_Toc42900786)

[2.2.4 Map Queries 7](#_Toc42900787)

[2.2.5 Search Web 8](#_Toc42900788)

[2.3 User Characteristics 9](#_Toc42900789)

[2.4 Non-Functional Requirements 9](#_Toc42900790)

[3.0. Requirements Specification 10](#_Toc42900791)

[3.1 Functional Requirements 10](#_Toc42900792)

[3.2.1 Fill Personal Information 10](#_Toc42900793)

[3.2.2 Voice Input 11](#_Toc42900794)

[3.2.3 Alarm Queries 11](#_Toc42900795)

[3.2.4 Map Queries 12](#_Toc42900796)

[3.2.5 Search Web 12](#_Toc42900797)

[3.3 Detailed Non-Functional Requirements 13](#_Toc42900798)

[3.3.1 Logical Structure of the Data 13](#_Toc42900799)

[3.3.2 Security 15](#_Toc42900800)

# List of Figures

[Figure 1 - System Environment 3](#_Toc77487669)

[Figure 2 – Fill Personal Information Process 4](#_Toc77487670)

[Figure 3 – Voice Input Process 5](#_Toc77487670)

[Figure 4 - Editor Use Cases 8](#_Toc77487671)

[Figure 5 - Logical Structure of the Article Manager Data 23](#_Toc77487672)

# 1.0. Introduction

## 1.1. Purpose

The purpose of this document is to present a detailed description of the Virtual Assistant System. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate and how the system will react to external stimuli. This document is intended for both the stakeholders and the developers of the system and will be proposed to Mrs. Nahla Saad Eldeen for its approval.

## 1.2. Scope of Project

This software system will be a Virtual Assistant System for any user on a mobile phone. This system will be designed to maximize the user’s productivity by providing tools to assist in accomplishing daily tasks, which would otherwise have to be performed manually. By maximizing the user’s work efficiency and production the system will meet the user’s needs while remaining easy to understand and use.

## 1.3. Glossary

|  |  |
| --- | --- |
| **Term** | **Definition** |
| Database | Collection of all the information monitored by this system. |
| Software Requirements Specification | A document that completely describes all of the functions of a proposed system and the constraints under which it must operate. For example, this document. |
| Stakeholder | Any person with an interest in the project who is not a developer. |
| User | Mobile phone owner |

## 1.4. References

IEEE. *IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications.* IEEE Computer Society, 1998.

## 1.5. Overview of Document

The next chapter, the Overall Description section, of this document gives an overview of the functionality of the product. It describes the informal requirements and is used to establish a context for the technical requirements specification in the next chapter.

The third chapter, Requirements Specification section, of this document is written primarily for the developers and describes in technical terms the details of the functionality of the product.

Both sections of the document describe the same software product in its entirety, but are intended for different audiences and thus use different language.

# 2.0. Overall Description

## 2.1 System Environment

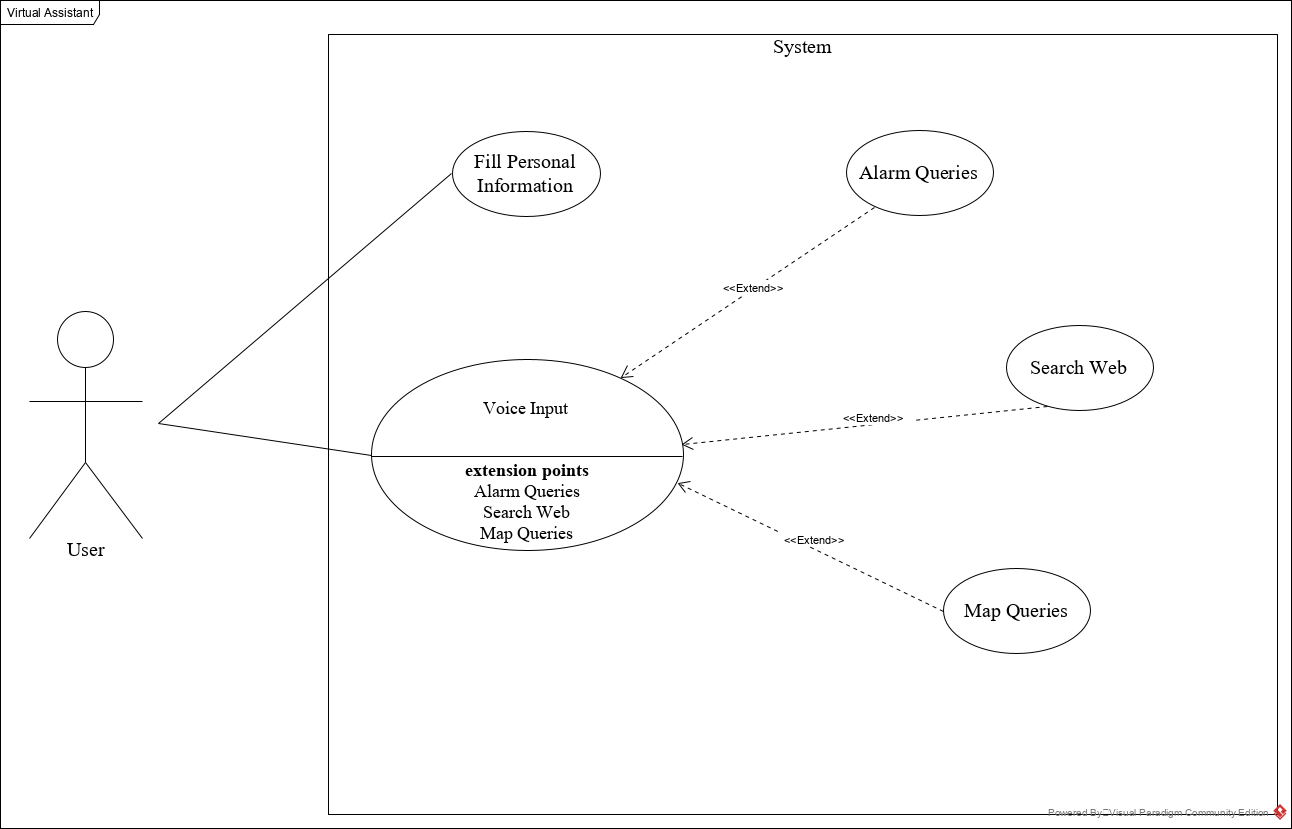


Figure - System Environment

The Virtual Assistant System has one active actor (the user).

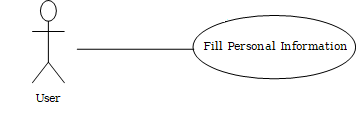
The user accesses the system through the application. Users communication with the system is through voice input.

## 2.2 Functional Requirements Specification

This section outlines the use cases for the user.

### 2.2.1 Use case: Fill Personal Information

**Diagram:**

****

**Brief Description**

The User accesses the system, inputs his personal information (e.g. name, age, job, …).

**Initial Step-By-Step Description**

Before this use case can be initiated, the User has already downloaded the Virtual Assistant application.

1. The User opens the application.
2. The system displays the information needed to be input.
3. The User inputs his/her personal information.
4. The system then transfers the user to the main application page.

**Xref:** Section 3.2.1, Fill Personal Information

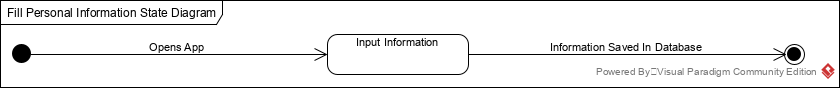
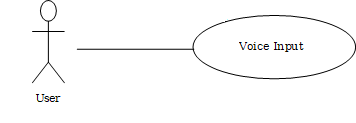


Figure - Fill Personal Information Process

The User opens the application after downloading and installing it. The system opens a bunch of pages requiring the user to fill some information about him/her so his experience gets to its best. The information filled then gets saved in the database to be used later in optimizing the user’s experience.

### 2.2.2 Use case: Voice Input

**Diagram:**

****

**Brief Description**

The User can use a keyword to activate the input functionality in the app.

**Initial Step-By-Step Description**

Before this use case can be initiated, the system has already activated the listener for the keyword.

1. The User says “Ok Assistant”.
2. The System starts listening to the user’s input.
3. The System then compiles the user’s input into commands to perform certain actions.

**Xref:** Section 3.2.2, Voice Input

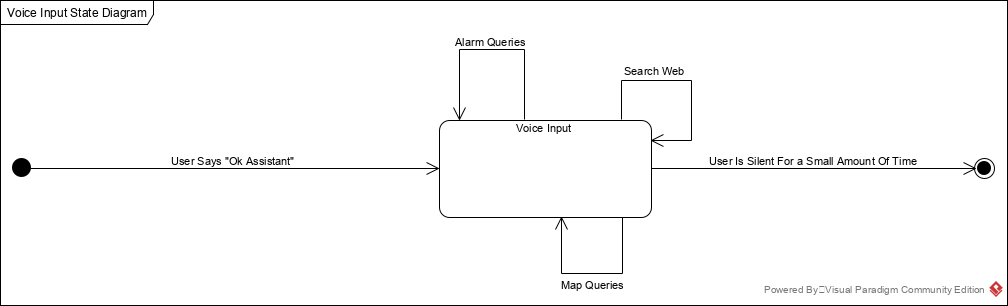


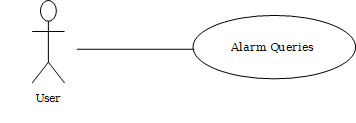
Figure 3 – Voice Input Process

The *Voice Input Process* state-transition diagram summarizes the use cases listed below.

The User starts off by saying “Ok Assistant”. The system then starts listening to the user voice commands. The User either makes Alarm Queries, Map Queries or Searches the web via voice commands. After a short amount of silence, the system stops listening to voice input and starts working on the commands input by the user.

### 2.2.3 **Use case: Alarm Queries**

**Diagram:**

****

**Brief Description**

The User tells the system to set an alarm for a certain time.

**Initial Step-By-Step Description**

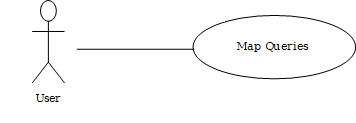
Before this use case can be initiated, the User has already said “Ok Assistant”.

1. The User makes an alarm query (e.g. wake me up in 45 minutes, set alarm on Sunday at 8:00 am).
2. The System uses the *voice input* to set an alarm for the user.

**Xref:** Section 3.2.2, Alarm Queries

### 2.2.4 Map Queries

**Diagram**



**Brief Description**

The User tells the system to set a waypoint or to look for a place on the map.

**Initial Step-By-Step Description**

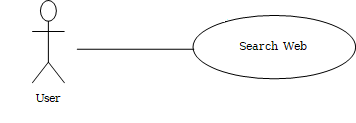
Before this use case can be initiated, the User has already said “Ok Assistant”.

1. The User makes a map query (e.g. find the shortest route to work, tell me where is the New York street is).
2. The System uses the *voice input* to set process map queries.

**Xref:** Section 3.2.2, Map Queries

### 2.2.5 Search Web

**Diagram**



**Brief Description**

The User tells the system to search the web for a certain information.

**Initial Step-By-Step Description**

Before this use case can be initiated, the User has already said “Ok Assistant”.

1. The User *tells* the system to look for something on the web (e.g. how to tie a tie,).
2. The System uses the *voice input* to search the web for the wanted information.

**Xref:** Section 3.2.3, Search Web

## 2.3 User Characteristics

The User is expected to be fluent in English and knows how to handle mobile phones.

## 2.4 Non-Functional Requirements

The system compiler will be on a server with high speed Internet capability. The physical machine to be used will be a mobile phone with internet connection.. The speed of the User’s connection will depend on the hardware used rather than characteristics of this system.

# 3.0. Requirements Specification

## 3.1 Functional Requirements

### Fill Personal Information

|  |  |
| --- | --- |
| **Use Case Name** | Search Article |
| **Xref** | Section 2.2.1, Search Article  SDD, Section 7.1 |
| **Trigger** | The Reader assesses the Online Journal Website |
| **Precondition** | The Web is displayed with grids for searching |
| **Basic Path** | 1. The Reader chooses how to search the Web site. The choices are by Author, by Category, and by Keyword. 2. If the search is by Author, the system creates and presents an alphabetical list of all authors in the database. In the case of an article with multiple authors, each is contained in the list. 3. The Reader selects an author. 4. The system creates and presents a list of all articles by that author in the database. 5. The Reader selects an article. 6. The system displays the Abstract for the article. 7. The Reader selects to download the article or to return to the article list or to the previous list. |
| **Alternative Paths** | In step 2, if the Reader selects to search by category, the system creates and presents a list of all categories in the database.   1. The Reader selects a category. 2. The system creates and presents a list of all articles in that category in the database. Return to step 5.   In step 2, if the Reader selects to search by keyword, the system presents a dialog box to enter the keyword or phrase.   1. The Reader enters a keyword or phrase. 2. The system searches the Abstracts for all articles with that keyword or phrase and creates and presents a list of all such articles in the database. Return to step 5. |
| **Postcondition** | The selected article is downloaded to the client machine. |
| **Exception Paths** | The Reader may abandon the search at any time. |
| **Other** | The categories list is generated from the information provided when article are published and not predefined in the Online Journal database. |

### 3.2.2 Voice Input

|  |  |
| --- | --- |
| **Use Case Name** | Communicate |
| **XRef** | Section 2.2.2, Submit Article; Section 2.2.3, Submit Review  SDD, Section 7.2 |
| **Trigger** | The user selects a *mailto* link. |
| **Precondition** | The user is on the *Communicate* page linked from the Online Journal Main Page. |
| **Basic Path** | This use case uses the *mailto* HTML tag. This invokes the client email facility. |
| **Alternative Paths** | If the user prefers to use his or her own email directly, sufficient information will be contained on the Web page to do so. |
| **Postcondition** | The message is sent. |
| **Exception Paths** | The attempt may be abandoned at any time. |
| **Other** | None |

### 3.2.3 Alarm Queries

|  |  |
| --- | --- |
| **Use Case Name** | Add Author |
| **XRef** | Section 2.2.4, Update Author  SDD, Section 7.3 |
| **Trigger** | The Editor selects to add a new author to the database. |
| **Precondition** | The Editor has accessed the Article Manager main screen. |
| **Basic Path** | 1. The system presents a blank grid to enter the author information. 2. The Editor enters the information and submits the form. 3. The system checks that the name and email address fields are not blank and updates the database. |
| **Alternative Paths** | If in step 2, either field is blank, the Editor is instructed to add an entry. No validation for correctness is made. |
| **Postcondition** | The Author has been added to the database. |
| **Exception Paths** | The Editor may abandon the operation at any time. |
| **Other** | The author information includes the name mailing address and email address. |

### 3.2.4 Map Queries

|  |  |
| --- | --- |
| **Use Case Name** | Add Reviewer |
| **XRef** | Section 2.2.4, Update Reviewer  SDD, Section 7.4 |
| **Trigger** | The Editor selects to add a new reviewer to the database. |
| **Precondition** | The Editor has accessed the Article Manager main screen. |
| **Basic Path** | 1. The system accesses the Historical Society (HS) database and presents an alphabetical list of the society members. 2. The Editor selects a person. 3. The system transfers the member information from the HS database to the Article Manager (AM) database. If there is no email address in the HS database, the editor is prompted for an entry in that field. 4. The information is entered into the AM database. |
| **Alternative Paths** | In step 3, if there is no entry for the email address in the HS database or on this grid, the Editor will be reprompted for an entry. No validation for correctness is made. |
| **Postcondition** | The Reviewer has been added to the database. |
| **Exception Paths** | The Editor may abandon the operation at any time. |
| **Other** | The Reviewer information includes name, membership number, mailing address, categories of interest, and email address. |

### 3.2.5 Search Web

|  |  |
| --- | --- |
| **Use Case Name** | Update Person |
| **XRef** | Sec 2.2.4 Update Author; Sec 2.2.4 Update Reviewer  SDD, Section 7.5 |
| **Trigger** | The Editor selects to update an author or reviewer and the person is already in the database. |
| **Precondition** | The Editor has accessed the Article Manager main screen. |
| **Basic Path** | 1. The Editor selects Author or Reviewer. 2. The system creates and presents an alphabetical list of people in the category. 3. The Editor selects a person to update. 4. The system presents the database information in grid form for modification. 5. The Editor updates the information and submits the form. 6. The system checks that required fields are not blank. |
| **Alternative Paths** | In step 5, if any required field is blank, the Editor is instructed to add an entry. No validation for correctness is made. |
| **Postcondition** | The database has been updated. |
| **Exception Paths** | If the person is not already in the database, the use case is abandoned. In addition, the Editor may abandon the operation at any time. |
| **Other** | This use case is not used when one of the other use cases is more appropriate, such as to add an article or a reviewer for an article. |

## 3.3 Detailed Non-Functional Requirements

### 3.3.1 Logical Structure of the Data

The logical structure of the data to be stored in the internal Article Manager database is given below.

Review

Reviewer

Article

Author

writes

sent to

writes

has

Figure - Logical Structure of the Article Manager Data

The data descriptions of each of these data entities is as follows:

**Author Data Entity**

|  |  |  |  |
| --- | --- | --- | --- |
| **Data Item** | **Type** | **Description** | **Comment** |
| Name | Text | Name of principle author |  |
| Email Address | Text | Internet address |  |
| Article | Pointer | Article entity | May be several |

**Reviewer Data Entity**

|  |  |  |  |
| --- | --- | --- | --- |
| **Data Item** | **Type** | **Description** | **Comment** |
| Name | Text | Name of principle author |  |
| ID | Integer | ID number of Historical Society member | Used as key in Historical Society Database |
| Email Address | Text | Internet address |  |
| Article | Pointer | Article entity of | May be several |
| Num Review | Integer | Review entity | Number of not returned reviews |
| History | Text | Comments on past performance |  |
| Specialty | Category | Area of expertise | May be several |

**Review Data Entity**

|  |  |  |  |
| --- | --- | --- | --- |
| **Data Item** | **Type** | **Description** | **Comment** |
| Article | Pointer | Article entity |  |
| Reviewer | Pointer | Reviewer entity | Single reviewer |
| Date Sent | Date | Date sent to reviewer |  |
| Returned | Date | Date returned; null if not returned |  |
| Contents | Text | Text of review |  |

**Article Data Entity**

|  |  |  |  |
| --- | --- | --- | --- |
| **Data Item** | **Type** | **Description** | **Comment** |
| Name | Text | Name of Article |  |
| Author | Pointer | Author entity | Name of principle author |
| Other Authors | Text | Other authors is any; else null | Not a pointer to an Author entity |
| Reviewer | Pointer | Reviewer entity | Will be several |
| Review | Pointer | Review entity | Set up when reviewer is set up |
| Contents | Text | Body of article | Contains Abstract as first paragraph. |
| Category | Text | Area of content | May be several |
| Accepted | Boolean | Article has been accepted for publication | Needs Copyright form returned |
| Copyright | Boolean | Copyright form has been returned | Not relevant unless Accepted is True. |
| Published | Boolean | Sent to Online Journal | Not relevant unless Accepted is True. Article is no longer active and does not appear in status checks. |

The Logical Structure of the data to be stored in the Online Journal database on the server is as follows:

**Published Article Entity**

|  |  |  |  |
| --- | --- | --- | --- |
| **Data Item** | **Type** | **Description** | **Comment** |
| Name | Text | Name of Article |  |
| Author | Text | Name of one Author | May be several |
| Abstract | Text | Abstract of article | Used for keyword search |
| Content | Text | Body of article |  |
| Category | Text | Area of content | May be several |

### 3.3.2 Security

The server on which the Online Journal resides will have its own security to prevent unauthorized *write*/*delete* access. There is no restriction on *read* access. The use of email by an Author or Reviewer is on the client systems and thus is external to the system.

The PC on which the Article Manager resides will have its own security. Only the Editor will have physical access to the machine and the program on it. There is no special protection built into this system other than to provide the editor with *write* access to the Online Journal to publish an article.