“Operation Elrond Domingo”

Software Requirements Specification

Version .1

1/29/2020

Ryan Tyniec

“Team Leader”

Tyler Dagge

“Team Designer”

Stephen Galvin

“Team Programmer”

Franklin Schneider

“Team Programmer”

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Description** | **Author** | **Comments** |
| 1/30/20 | 0.1 | Ryan Tyniec | Created a Web API project, configured API with user authentication, used entity framework generated code base for user authentication. |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Document Approval**

The following Software Requirements Specification has been accepted and approved by the following:

|  |  |  |  |
| --- | --- | --- | --- |
| **Signature** | **Printed Name** | **Title** | **Date** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Table of Contents**

**1. Introduction 1**

1.1 Purpose 1

1.2 Scope 1

1.3 Definitions, Acronyms, and Abbreviations 1

1.4 Overview 1

**2. General Description 1**

2.1 Product Perspective 1

2.2 Product Functions 2

2.3 Users and Characteristics 2

2.4 General Constraints 2

2.5 Assumptions and Dependencies 2

2.6 Operating Environment 2

**3. Specific Requirements 2**

3.1 External Interface Requirements 3

*3.1.1 User Interfaces 3*

*3.1.2 Hardware Interfaces 3*

*3.1.3 Software Interfaces 3*

*3.1.4 Communications Interfaces 3*

3.2 Functional Requirements 3

*3.2.1 <Functional Requirement or Feature #1> 3*

*3.2.2 <Functional Requirement or Feature #2> 3*

3.3 Use Cases 4

*3.3.1 Use Case #1 4*

*3.3.2 Use Case #2 4*

3.4 Non-Functional Requirements 4

*3.5.1 Performance 5*

*3.5.2 Reliability 5*

*3.5.3 Availability 5*

*3.5.4 Security 5*

*3.5.5 Maintainability 5*

*3.5.6 Portability 5*

3.5 Design Constraints 5

3.6 Logical Database Requirements 5

3.7 Other Requirements 5

**4. Analysis Models 5**

4.1 Sequence Diagrams 6

**5. Change Management Process 6**

**A. Appendices 6**

A.1 Appendix 1 7

A.2 Appendix 2 7

**List of Figures**

[Figure 1: Data Flow Diagram Example 1 6](#_2u6wntf)

[Figure 2 Data Flow Diagram Example 2 6](#_19c6y18)

**List of Tables**

[Table 1: Use case 1 4](#_49x2ik5)

# 1. Introduction

The introduction to the Software Requirement Specification (SRS) document should provide an overview of the complete SRS document. While writing this document please remember that this document should contain all of the information needed by a software engineer to adequately design and implement the software product described by the requirements listed in this document.

## 1.1 Purpose

The purpose of this document is to provide a thorough overview of the “Dungeons and Dragons Character Creator” application.. The document will primarily be used for both the developers of the application, as well as for any potential clients that are requesting the software. The document will include a general description of the project, as well as the overall scope, goals, and limitations of the project.

## 1.2 Scope

(1) **Our Project Name:** Dungeons & Dragons Character Builder

(2) **What is the goal of this application?:** The application will be designed to allow players of Dungeons and Dragons to be able to create, edit, and view character builds for games. The application will allow users to build and store all of their builds.

(3) Our goal for this software is to create and manage a service based Windows application for desktops. To build the application, we are using Microsoft’s .NET Framework as a platform of development. We plan to implement a comprehensive WPF front end that allows users to view their character build sheets, as well as create new ones. Due to the nature of Dungeons and Dragons, a primary goal for the team is to allow character creation for all players to be as simple and intuitive as possible. To achieve this, the project should have an emphasis on creating an intuitive user interface that is easy to use.

Since this application is highly reliant on data, we also plan to incorporate a local database for the application. Due to the nature of software development, we plan to use programming methods that ensure there is as little dependency as possible between our front end design and back end local database to allow expansion of the software to be more efficient after release.

As of now, the application’s scope is limited to the desktop it is being used on. Therefore, besides the initial install, the application will not directly require internet access or external connections to databases.

## 1.3 Definitions, Acronyms, and Abbreviations

|  |  |
| --- | --- |
| User | Person who is using the application when it is released. |
| D&D | Abbreviation for the game Dungeons and Dragons. |
| WPF | Windows Presentation Foundation - the platform we will be using to create the user interface of the application. |
| Front End | The part of the application that users will interact with. |
| Back End | The part of the application that works with the front end; this handles the initialization of the application and handles data transmission. |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

## 1.4 Overview

The remainder of this document is comprised of three additional chapters. The remaining chapters will follow the same naming conventions and numbering precedence. The remaining chapters will further detail the functions of the project and limitations of the application.

# 2. General Description

<This section of the SRS should describe the general factors that affect the product and its requirements. This section does not state specific requirements; it only makes those requirements easier to understand.>

## 2.1 Product Perspective

This product is a new self-contained product that is utilizing the Dungeons and Dragons roleplaying system to make a character creation software that aims to provide a benefit to both new players and returning players. This product is designed to be used to supplement a player intending to create a new character, and provides that service so that people may play their own game of dungeons and dragons. The larger “system” of dungeons and dragons does not require this product, however in our teams observation of the player base of this game, a character builder that focuses on user choice would find a strong home inside the community of people involved with the game. Within

## 2.2 Product Functions

Major Functions:

* Accessible as a standard Windows application.
* Create a character for Dungeons and Dragons.
* Display character sheet upon finishing the creation of a character.
* An intuitive user interface for the user to be guided through character creation.
* The program must let the user make decisions in regards to building the character.
* The program must explain what the options the user can make and how they change their stats.

## 2.3 Users and Characteristics

**User Classes in order of importance**

1. **New D&D Players**: Inexperienced, need the step-by-step creation process, need character features explained to them.
2. **Experienced D&D Players**: Are already knowledgeable about D&D and would use the software just for convenience.
   1. **People Trying to Get Others Into D&D**: This group is a subset of the above players, they are generally experienced and want to get more people into their hobby/playgroup, and intend to use this product to make the skill gap a smaller one for people to jump.
3. **The Development Team:** This group will be the developers as we intend to test, use, and expand upon this project overtime, and will need to dedicate time where we as a team put ourselves into the shoes of the other user types.

## 2.4 General Constraints

<This subsection of the SRS should provide a general description of any other items that will limit the designer/developer’s options for designing/developing the system.>

* **Time**: Only have <1 semester to work on the project (Along with other responsibilities).
* **Skill/Ability**: Lack of general skill and knowledge of both the program language and the method to complete the program restricts ability and slows the development process.

## 2.5 Assumptions and Dependencies

<This subsection of the SRS should list each of the factors that affect the requirements stated in the SRS. These factors are not design constraints on the software but are, rather, any changes to them that can affect the requirements in the SRS. For example, an assumption might be that a specific operating system will be available on the hardware designated for the software product. If, in fact, the operating system is not available, the SRS would then have to change accordingly.>

How we intend to handle data might change during development (API vs. Some form of File System), and thus the way of implementing some features of the software would have to change accordingly.

## 2.6 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.>

Windows, desktop based software, currently slated with a file system that will sit with the product that contains the user data for character creation.

# 3. Specific Requirements

<This will be the largest and most important section of the SRS. The customer requirements are embodied within Section 2 (functions), but this section will give the D-requirements that are used to guide the project’s software design, implementation, and testing.

Each requirement in this section should be:

* Correct
* Traceable (both forward and backward to prior/future artifacts)
* Unambiguous
* Verifiable (i.e., testable)
* Prioritized (with respect to importance and/or stability)
* Complete
* Consistent (with other requirements)
* Uniquely identifiable (usually via numbering like 3.4.5.6)

Attention should be paid to carefully organize the requirements presented in this section so that they may easily accessed and understood. Furthermore, this SRS is not the software design document, therefore one should avoid the tendency to over-constrain (and therefore design) the software project within this SRS.>

## 3.2 Functional Requirements

This section describes specific features of the software project. If desired, some requirements may be specified in the use-case format and listed in the Use Cases Section.

### 3.2.1 FR 1

### TITLE: A Desktop Based Application

DESC: The user should be able to download this application onto their Windows computer, through an online distribution(such as a Google Drive folder), it should be free to download.

RAT: A method for the user to use the application.

PRIO: 9 (Largest priority, the basis of all other requirements falls into having an actual application.)

RISK: 1 (Lowest risk, as this is central to every other requirement, if this doesn’t function we will know right away and be able to step in and fix it.)

DEP: None

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).>

#### 3.2.1.2 Stimulus/Response Sequences

The user will need to download this application, allocate space for the files. Then the user will run the application launching a window that greets them and prompts them for input.

<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.>

#### 3.2.1.3 Functional Requirements

The software capabilities would be a program that compiles and holds the rest of the software together, this requirement is the base level upon which all others will be built from, a functional application that can be downloaded upon the users computer. The anticipated errors range from the standard failure to design the application in a manner that allows it to function, alongside other issues that may arise from downloading and setting up the software TBD TBD TBD(we need to expand upon this further into planning of the development.)

<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.>*

##### 3.2.1.3.1 REQ-1:

##### The application requires a method to install the application and then have a functional interface that allows for movement into the other functional requirements. This is the base level upon which others will draw from.

##### <Example, for the ATM Machine withdraw cash feature, the visual interface has to be working for the user to be able to enter information>

##### 3.2.1.3.2 REQ-2:

### 3.2.2 FR 2

TITLE: A User Interface

DESC: This should be a user interface, that allows the user to interact with the software in a clear and concise manner. The interface should be functional and not be poorly designed or strain the user in its use.

RAT: A method for the user to interact with the application.

PRIO: 8 (Large priority, this application needs to have user interaction, and this will be done through a user interface.)

RISK: 3 (low risk, most of risk stems from poor design of the interface, not on it running, this follows FR 1 in terms of it being very low level in terms of design and the issue with it would be very fast in detection.)

DEP: FR 1

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).>

#### 3.2.2.2 Stimulus/Response Sequences

The user will run the application and be greeted with a prompt to start character creation or to recover a previous character sheet to be displayed. Upon selecting the start creation they will be taken through the process listed in FR X, this creation process is detailed there. If choosing the second option, it will be detailed in FR 3.

#### 3.2.2.3 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.>*

##### 3.2.2.3.1 REQ-1:

##### The application requires a method to install the application and then have a functional interface that allows for movement into the other functional requirements. This is the base level upon which others will draw from.

##### <Example, for the ATM Machine withdraw cash feature, the visual interface has to be working for the user to be able to enter information>

##### 3.2.1.3.2 REQ-2:

### 3.2.3 FR 3

### TITLE: Displaying character sheet

### DESC: A user should be able to display their finalized D&D character sheet.

RAT: To allow the user to view the information they have submitted for their character.

PRIO: 8

RISK: 4

DEP: FR 1, FR 2, FR X

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).>

#### 3.2.2.2 Stimulus/Response Sequences

After the required information has been entered into the program, the user would be able to click a button to display their character sheet. The user will also be able to display a character sheet that has been created previously by loading the character sheet through the program.

#### 3.2.2.3 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.>*

##### 3.2.2.3.1 REQ-1:

##### 

##### <Example, for the ATM Machine withdraw cash feature, the visual interface has to be working for the user to be able to enter information>

##### 3.2.1.3.2 REQ-2:

## 3.3 Use Cases

### 3.3.1 Use Case #1

|  |  |
| --- | --- |
| **Use Case Name** | Search Article |
| **Reference** | Section 2.2.1, 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 articles are published and not predefined in the Online Journal database. |

*Table 1: Use case 1*

### 3.3.2 Use Case #2

…

## 3.4 Non-Functional Requirements

<Non-functional requirements may exist for the following attributes. Often these requirements must be achieved at a system-wide level rather than at a unit level. State the requirements in the following sections in measurable terms (e.g., 95% of transaction shall be processed in less than a second, system downtime may not exceed 1 minute per day, > 30 day MTBF value, etc.).>

### 3.5.1 Performance

### 3.5.2 Reliability

The program should be able to handle whatever the user does within it and build a character.

### 3.5.3 Availability

The Program will be available on Windows. The program should be able to be downloaded and run locally, so there would be no availability hurdles.

### 3.5.4 Security

<Example: 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.>

As an offline application, there should be no security concerns. If the user tried to open a file that wasn’t compatible with the application, the application would stop them.

### 3.5.5 Maintainability

The program should be made in such a way that updating information and adding new options for character creation would be simple and non-conflicting.

### 3.5.6 Portability

All the files needed for the application should be contained to one folder, allowing for program portability.

## 3.5 Design Constraints

<Specify design constraints imposed by other standards, company policies, hardware limitations, etc. that will impact this software project. Example, the software is required to have a login screen based on company policies.>

## 3.6 Logical Database Requirements

<Will a database be used? If so, what logical requirements exist for data formats, storage capabilities, data retention, data integrity, etc?>

A SQLite database file will be used to store organized data, but the data stored in the tables doesn't really interact with one another; and there is not so much data that size would become an issue.

## 3.7 Other Requirements

<Catchall section for any additional requirements that did not belong to the previous sections. If there are none, exclude this section>

# 4. Analysis Models

<List all analysis models used in developing specific requirements previously given in this SRS. Each model should include an introduction and a narrative description. Furthermore, each model should be traceable the SRS’s requirements.>

## Sequence Diagrams

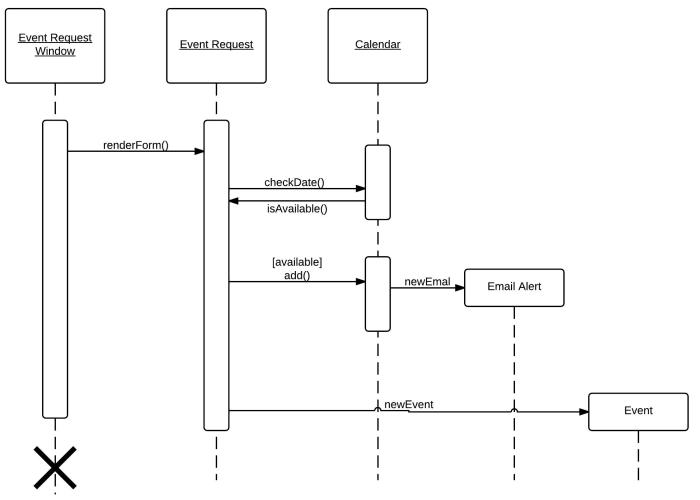


Figure 1: Data Flow Diagram Example 1

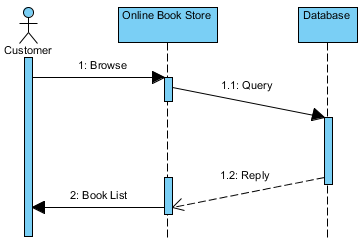


Figure 2 Data Flow Diagram Example 2

<At least one sequence diagram should be included for each requirement or use case.>

# 5. Change Management Process

<Identify and describe the process that will be used to update the SRS, as needed, when project scope or requirements change. Who can submit changes and by what means, and how will these changes be approved.>

# References

# Appendices

<Appendices may be used to provide additional (and hopefully helpful) information. If present, the SRS should explicitly state whether the information contained within an appendix is to be considered as a part of the SRS’s overall set of requirements. Example Appendices could include (initial) conceptual documents for the software project, marketing materials, minutes of meetings with the customer(s), etc.>

## A.1 Appendix 1

## A.2 Appendix 2