Software Requirements Specification

Version 1.5

<<Annotated Version>>

June 20, 2020

Book Store System

Hazar Al-Abdullah

Lilas Al-Shwaity

Aya Shullar

Wessam Al-Mahmoud

Submitted in full fulfillment

Of the requirements of

Software Engineering course

<<Any comments inside double brackets such as these are *not* part of this SRS but are comments upon this SRS example to help the reader understand the point being made.

Refer to the SRS Template for details on the purpose and rules for each section of this document. >>

# Table of Contents

[Table of Contents i](#_Toc43582823)

[List of Figures i](#_Toc43582824)

[1.0. Introduction 1](#_Toc43582825)

[1.1. Purpose 1](#_Toc43582826)

[1.2. Scope of Project 1](#_Toc43582827)

[1.3. Glossary 2](#_Toc43582828)

[1.4. References 2](#_Toc43582829)

[1.5. Overview of Document 2](#_Toc43582830)

[2.0. Overall Description 3](#_Toc43582831)

[2.1 System Environment 3](#_Toc43582832)

[2.2 Functional Requirements Specification 4](#_Toc43582833)

[2.2.1 Use case: View all books 4](#_Toc43582834)

[2.2.2 Use case: Search for a book 5](#_Toc43582835)

[2.2.3 Use case: Add a book 6](#_Toc43582836)

[2.2.4 Use case: Update a book 7](#_Toc43582837)

[**Xref:** Section 3.2.4, Update a book 7](#_Toc43582838)

[Use case: Update Reviewer 8](#_Toc43582839)

[Use case: Update Article 8](#_Toc43582840)

[Use case: Receive Article 9](#_Toc43582841)

[Use case: Assign Reviewer 9](#_Toc43582842)

[Use case: Receive Review 10](#_Toc43582843)

[Use case: Check Status 11](#_Toc43582844)

[Use case: Send Response 11](#_Toc43582845)

[Use case: Send Copyright 12](#_Toc43582846)

[Use case: Remove Article 13](#_Toc43582847)

[Use case: Publish Article 13](#_Toc43582848)

[2.3 User Characteristics 14](#_Toc43582849)

[2.4 Non-Functional Requirements 14](#_Toc43582850)

[3.0. Requirements Specification 16](#_Toc43582851)

[3.1 External Interface Requirements 16](#_Toc43582852)

[3.2 Functional Requirements 17](#_Toc43582853)

[3.2.1 View all books 17](#_Toc43582854)

[3.2.2 Search for a book 17](#_Toc43582855)

[3.2.3 Add a book 18](#_Toc43582856)

[3.2.4 Update a book 18](#_Toc43582857)

[3.2.5 Update Person 19](#_Toc43582858)

[3.3 Detailed Non-Functional Requirements 19](#_Toc43582859)

[3.3.1 Logical Structure of the Data 19](#_Toc43582860)

[3.3.2 Security 21](#_Toc43582861)

[Index 23](#_Toc43582862)

# 

# List of Figures

[Figure 1 - System Environment 3](#_Toc43582863)

[Figure 4 - Logical Structure of the Article Manager Data 20](#_Toc43582864)

# 1.0. Introduction

## 1.1. Purpose

The purpose of this document is to present a detailed description of the Book Store System. It will explain the purpose and features of the system, what the system will do, the constraints under which it must operate. This document is intended for both the stakeholders and the developers of the system and will be proposed to Mrs. Nahla Saad Eldeen.

## 1.2. Scope of Project

This software system will be a Book Store System for a local author. This system will be designed to maximize the author’s productivity by providing tools to assist in automating the books publishing process. By maximizing the author’s work efficiency and production the system will meet the author’s needs while remaining easy to understand and use.

More specifically, this system is designed to allow an author to manage and publish books to a public website. The system also contains a relational database containing a list of books.

## 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 | Author. |

## 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

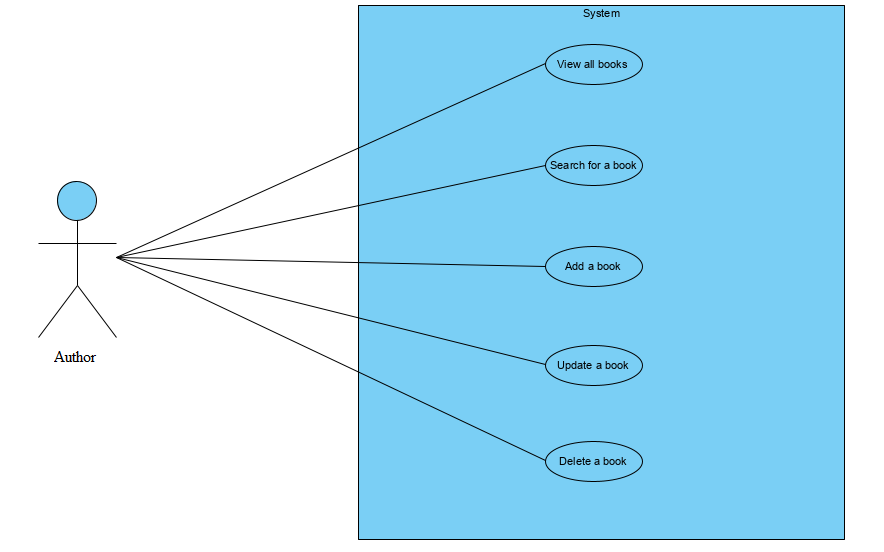


Figure 1 - System Environment

The Book Store System has one active actor.

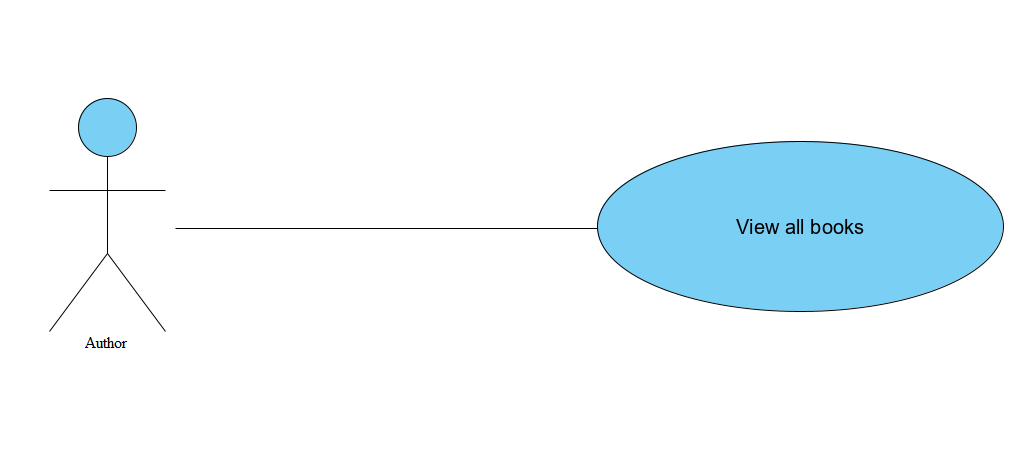
The Author accesses the website through the Internet.

## 2.2 Functional Requirements Specification

This section outlines the use cases for the author.

### 2.2.1 Use case: View all books

**Diagram:**



**Brief Description**

The Author accesses the website, the system automatically shows all the books in the database.

**Initial Step-By-Step Description**

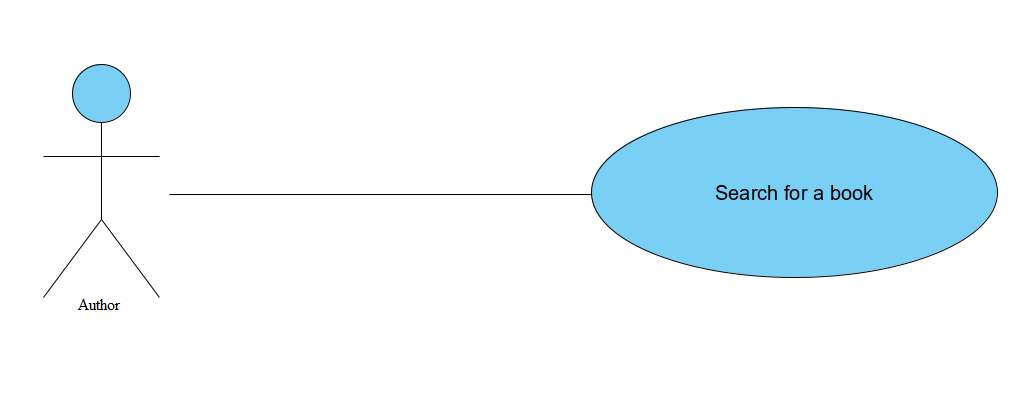
Before this use case can be initiated, the Author has already accessed the website.

1. The Author access the website.
2. The System then fetches all the books in the database and shows it to the author.

**Xref:** Section 3.2.1, View all books

### 2.2.2 Use case: Search for a book

**Diagram:**



**Brief Description**

The author searches for a certain book in the website.

**Initial Step-By-Step Description**

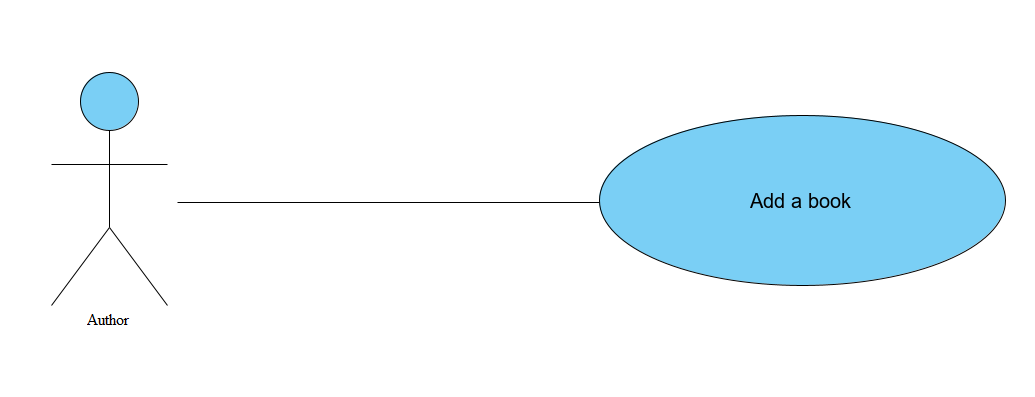
Before this use case can be initiated, the Author has already connected to the website.

1. The Author enters the book’s title or tag or keywords in the search bar.
2. The System search’s the database for a matching book.
3. The System displays a list of books related to the authors input.

**Xref:** Section 3.2.2, Search for a book

### 2.2.3 Use case: Add a book

**Diagram:**

****

**Brief Description**

The Author adds a new book to the website.

**Initial Step-By-Step Description**

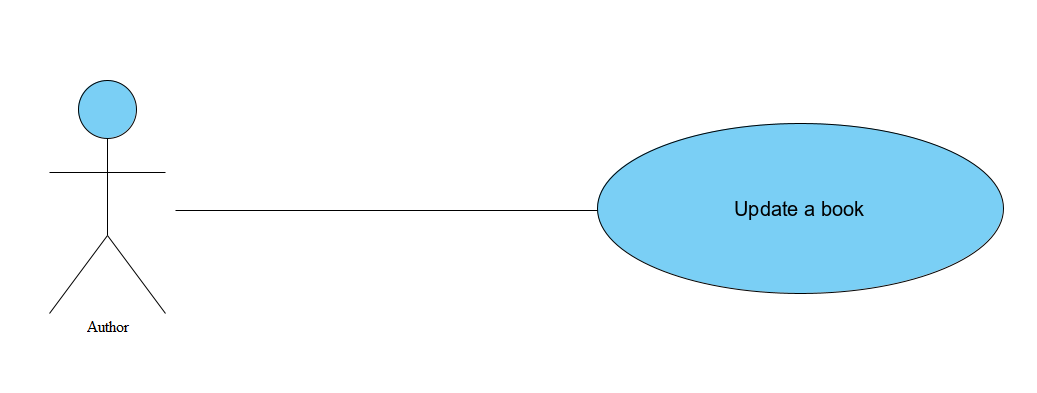
Before this use case can be initiated, the Author has already connected to the website.

1. The Author chooses to add a new book.
2. The System directs the Author to the addition page.
3. The Author selects the book to be uploaded.
4. The System adds the book to the database.

**Xref:** Section 3.2.3, Add a book

### 2.2.4 Use case: Update a book

**Diagram:**

****

**Brief Description**

The Author updates an existing book’s title, tag, content or keywords in the database.

**Initial Step-By-Step Description**

Before this use case can be initiated, the Author has already accessed the website.

1. The Author clicks the update a book button.
2. The System directs the Author to the updating page.
3. The Author selects the book to be updated.
4. The System opens the online editor so the Author can update a book’s title, tag, content or keywords.
5. The System saves the changes made into the database.

### **Xref:** Section 3.2.4, Update a book

#### Use case: Update Reviewer

**Diagram:**

Editor

Update Reviewer

Hist Soc DB

**Brief Description**

The Editor enters a new Reviewer or updates information about a current Reviewer.

**Initial Step-By-Step Description**

Before this use case can be initiated, the Editor has already accessed the main page of the Article Manager.

1. The Editor selects to *Add/Update Reviewer*.
2. The system presents a choice of adding or updating.
3. The Editor chooses to add or to update.
4. The system links to the Historical Society Database.
5. If the Editor is updating a Reviewer, the system and presents a grid with the information about the Reviewer; else the system presents list of members for the editor to select a Reviewer and presents a grid for the person selected.
6. The Editor fills in the information and submits the form.
7. The system verifies the information and returns the Editor to the Article Manager main page.

**Xref:** Section 3.2.4, Add Reviewer; Section 3.2.5, Update Person

#### Use case: Update Article

**Diagram:**

Editor

Update Article

**Brief Description**

The Editor enters information about an existing article.

**Initial Step-By-Step Description**

Before this use case can be initiated, the Editor has already accessed the main page of the Article Manager.

1. The Editor selects to *Update Article*.
2. The system presents s list of active articles.
3. The system presents the information about the chosen article.
4. The Editor updates and submits the form.
5. The system verifies the information and returns the Editor to the Article Manager main page.

**Xref:** Section 3.2.6, Update Article Status

**Handle Article use cases**

#### Use case: Receive Article

Editor

Receive Article

**Diagram:**

**Brief Description**

The Editor enters a new or revised article into the system.

**Initial Step-By-Step Description**

Before this use case can be initiated, the Editor has already accessed the main page of the Article Manager and has a file containing the article available.

1. The Editor selects to *Receive Article*.
2. The system presents a choice of entering a new article or updating an existing article.
3. The Editor chooses to add or to update.
4. If the Editor is updating an article, the system presents a list of articles to choose from and presents a grid for filling with the information; else the system presents a blank grid.
5. The Editor fills in the information and submits the form.
6. The system verifies the information and returns the Editor to the Article Manager main page.

**Xref:** Section 3.2.7, Enter Communication

#### Use case: Assign Reviewer

This use case extends the *Update Article* use case.

**Diagram:**

Editor

Assign Reviewer

Hist Soc DB

**Brief Description**

The Editor assigns one or more reviewers to an article.

**Initial Step-By-Step Description**

Before this use case can be initiated, the Editor has already accessed the article using the *Update Article* use case.

1. The Editor selects to *Assign Reviewer*.
2. The system presents a list of Reviewers with their status (see data description is section 3.3 below).
3. The Editor selects a Reviewer.
4. The system verifies that the person is still an active member using the Historical Society Database.
5. The Editor repeats steps 3 and 4 until sufficient reviewers are assigned.
6. The system emails the Reviewers, attaching the article and requesting that they do the review.
7. The system returns the Editor to the *Update Article* use case.

**Xref:** Section 3.2.8, Assign Reviewer

#### Use case: Receive Review

Editor

Receive Review

This use case extends the *Update Article* use case.

**Diagram:**

**Brief Description**

The Editor enters a review into the system.

**Initial Step-By-Step Description**

Before this use case can be initiated, the Editor has already accessed the article using the *Update Article* use case.

1. The Editor selects to *Receive Review*.
2. The system presents a grid for filling with the information.
3. The Editor fills in the information and submits the form.
4. The system verifies the information and returns the Editor to the Article Manager main page.

**Xref:** Section 3.2.7, Enter Communication

**Check Status use case:**

#### Use case: Check Status

**Diagram:**

Editor

Check Status

**Brief Description**

The Editor checks the status of all active articles.

**Initial Step-By-Step Description**

Before this use case can be initiated, the Editor has already accessed the main page of the Article Manager.

1. The Editor selects to *Check Status*.
2. The system returns a scrollable list of all active articles with their status (see data description in section 3.3 below).
3. The system returns the Editor to the Article Manager main page.

**Xref:** Section 3.2.9, Check Status

**Send Recommendation use cases:**

#### Use case: Send Response

This use case extends the *Update Article* use case.

**Diagram:**

Editor

Send Response

**Brief Description**

The Editor sends a response to an Author.

**Initial Step-By-Step Description**

Before this use case can be initiated, the Editor has already accessed the article using the *Update Article* use case.

1. The Editor selects to *Send Response*.
2. The system calls the email system and puts the Author’s email address in the Recipient line and the name of the article on the subject line.
3. The Editor fills out the email text and sends the message.
4. The system returns the Editor to the Article Manager main page.

**Xref:** Section 3.210, Send Communication

#### Use case: Send Copyright

This use case extends the *Update Article* use case.

**Diagram:**

**Brief Description**

Editor

Send Copyright

The Editor sends a copyright form to an Author.

**Initial Step-By-Step Description**

Before this use case can be initiated, the Editor has already accessed the article using the *Update Article* use case.

1. The Editor selects to *Send Copyright*.
2. The system calls the email system and puts the Author’s email address in the Recipient line, the name of the article on the subject line, and attaches the copyright form.
3. The Editor fills out the email text and sends the message.
4. The system returns the Editor to the Article Manager main page.

**Xref:** Section 3.2.10, Send Communication

#### Use case: Remove Article

This use case extends the *Update Article* use case.

Editor

Remove Article

**Diagram:**

**Brief Description**

The Editor removes an article from the active category.

**Initial Step-By-Step Description**

Before this use case can be initiated, the Editor has already accessed the article using the *Update Article* use case.

1. The Editor selects to remove an article from the active database.
2. The system provides a list of articles with the status of each.
3. The Editor selects an article for removal.
4. The system removes the article from the active article database and returns the Editor to the Article Manager main page.

**Xref:** Section 3.2.12, Remove Article

**Publish Article use case:**

#### Use case: Publish Article

This use case extends the *Update Article* use case.

Editor

Publish Article

**Diagram:**

**Brief Description**

The Editor transfers an accepted article to the Online Journal.

**Initial Step-By-Step Description**

Before this use case can be initiated, the Editor has already accessed the article using the *Update Article* use case.

1. The Editor selects to *Publish Article*.
2. The system transfers the article to the Online Journal and updates the search information there.
3. The system removes the article from the active article database and returns the Editor to the Article Manager home page.

**Xref:** Section 3.2.11, Publish Article

<< Since three of the actors only have one use case each, the summary diagram only involves the Editor. Adapt the rules to the needs of the document rather than adapt the document to fit the rules. >>

## 2.3 User Characteristics

The Reader is expected to be Internet literate and be able to use a search engine. The main screen of the Online Journal Website will have the search function and a link to “Author/Reviewer Information.”

The Author and Reviewer are expected to be Internet literate and to be able to use email with attachments.

The Editor is expected to be Windows literate and to be able to use button, pull-down menus, and similar tools.

The detailed look of these pages is discussed in section 3.2 below.

## 2.4 Non-Functional Requirements

The Online Journal will be on a server with high speed Internet capability. The physical machine to be used will be determined by the Historical Society. The software developed here assumes the use of a tool such as Tomcat for connection between the Web pages and the database. The speed of the Reader’s connection will depend on the hardware used rather than characteristics of this system.

The Article Manager will run on the editor’s PC and will contain an Access database. Access is already installed on this computer and is a Windows operating system.

# 3.0. Requirements Specification

## 3.1 External Interface Requirements

The only link to an external system is the link to the Historical Society (HS) Database to verify the membership of a Reviewer. The Editor believes that a society member is much more likely to be an effective reviewer and has imposed a membership requirement for a Reviewer. The HS Database fields of interest to the Web Publishing Systems are member’s name, membership (ID) number, and email address (an optional field for the HS Database).

The *Assign Reviewer* use case sends the Reviewer ID to the HS Database and a Boolean is returned denoting membership status. The *Update Reviewer* use case requests a list of member names, membership numbers and (optional) email addresses when adding a new Reviewer. It returns a Boolean for membership status when updating a Reviewer.

## 3.2 Functional Requirements

The Logical Structure of the Data is contained in Section 3.3.1.

### 3.2.1 View all books

|  |  |
| --- | --- |
| **Use Case Name** | View all books |
| **XRef** | Section 2.2.1, View all books |
| **Trigger** | The Author assesses the website |
| **Precondition** | The Author has added books to the database. |
| **Basic Path** | 1. The Author access the website. 2. The System then fetches all the books in the database and shows it to the author. |
| **Alternative Paths** | None |
| **Postcondition** | All of the books in the database are shown to the author. |
| **Exception Paths** | None |
| **Other** | None |

### 3.2.2 Search for a book

|  |  |
| --- | --- |
| **Use Case Name** | Search for a book |
| **XRef** | Section 2.2.2, Search for a book |
| **Trigger** | The author inputs a tag or a keyword or any text in the search bar and hits search. |
| **Precondition** | The author has already accessed the website. |
| **Basic Path** | 1. The Author enters the book’s title or tag or keywords in the search bar. 2. The System search’s the database for a matching book. 3. The System displays a list of books related to the authors input. |
| **Alternative Paths** | None |
| **Postcondition** | A list of books related to the author’s input is shown |
| **Exception Paths** | None |
| **Other** | None |

### 3.2.3 Add a book

|  |  |
| --- | --- |
| **Use Case Name** | Add a book |
| **XRef** | Section 2.2.3, Add a book |
| **Trigger** | The Author clicks the add new book button. |
| **Precondition** | The Author has accessed the website. |
| **Basic Path** | 1. The Author chooses to add a new book. 2. The System directs the Author to the addition page. 3. The Author selects the book to be uploaded. 4. The System adds the book to the database. |
| **Alternative Paths** | None |
| **Postcondition** | The System adds the book to the database. |
| **Exception Paths** | None |
| **Other** | None |

### 3.2.4 Update a book

|  |  |
| --- | --- |
| **Use Case Name** | Update a book |
| **XRef** | Section 2.2.4, Update a book |
| **Trigger** | The Author clicks the update a book button. |
| **Precondition** | The Author has accessed the website. |
| **Basic Path** | 1. The Author clicks the update a book button. 2. The System directs the Author to the updating page. 3. The Author selects the book to be updated. 4. The System opens the online editor so the Author can update a book’s title, tag, content or keywords. 5. The System saves the changes made into the database. |
| **Alternative Paths** | None |
| **Postcondition** | The System saves the changes made into the database. |
| **Exception Paths** | None |
| **Other** | None |

### 3.2.5 Update Person

|  |  |
| --- | --- |
| **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 4 - 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.

# Index

Abstract, 6, 17, 27

add, 9, 11, 19, 20, 21

Add, 8, 9, 19

Article, 1, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28

Article Manager, 5, 8, 9, 10, 11, 12, 13, 14, 15, 16, 19, 20, 21, 22, 23, 24, 25, 28

Author, 1, 4, 5, 6, 7, 8, 9, 13, 14, 16, 17, 19, 20, 22, 23, 25, 26, 27

Category, 5, 14, 17, 18, 20, 21, 23, 26, 27

Database, 2, 9, 11, 14, 15, 16, 17, 18, 19, 20, 21, 22, 24, 25, 26, 27

Editor, 1, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 19, 20, 21, 22, 23, 24, 25, 28

Field, 17, 19, 20

Form, 1, 6, 9, 10, 11, 12, 14, 19, 20, 21, 23, 24, 27

Grid, 9, 11, 12, 19, 20, 21

Historical Society, 1, 5, 9, 11, 16, 17, 19, 20, 26

Online Journal, 4, 5, 6, 7, 15, 16, 17, 18, 24, 27, 28

Reader, 4, 5, 6, 16, 17, 18

Review, 1, 7, 11, 12, 18, 21, 23, 26, 27

Reviewer, 1, 4, 5, 6, 7, 9, 11, 16, 17, 19, 20, 21, 22, 23, 26, 27

Security, 27, 28

Status, 11, 12, 13, 14, 17, 21, 22, 23, 27

update, 9, 11, 20, 21

Update, 8, 9, 10, 11, 12, 13, 14, 15, 17, 19, 20, 21, 22

User, 7, 16, 18

Web Publishing System, 1, 4, 5, 17