Software Requirements Specification

Version 1.0

<<Annotated Version>> April 15, 2004

Web Publishing System

Joan Teamleader Paul Adams Bobbie Baker Charles Charlie

Submitted in partial fulfillment Of the requirements of

CS 310 Software Engineering

<<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.

This work is based upon the submissions of the Spring 2004 CS 310. The students who submitted these team projects were Thomas Clay, Dustin Denney, Erjon Dervishaj, Tiffanie Dew, Blake Guice, Jonathan Medders, Marla Medders, Tammie Odom, Amro Shorbatli, Joseph Smith, Jay Snellen, Chase Tinney, and Stefanie Watts. >>

|  |  |  |  |
| --- | --- | --- | --- |
|  | | **Table of Contents** |  |
| Table of Contents List of Figures  1.0. Introduction | |  | i ii 1 |
| 1.1. Purpose | |  | 1 |
| 1.2. Scope of Project | |  | 1 |
| 1.3. Glossary | |  | 2 |
| 1.4. References | |  | 2 |
| 1.5. Overview of Document | |  | 2 |
| 2.0. | **Error! Bookmark not defined.** | | |
| 2.1 | 4 | | |
| 2.2 | 5 | | |
| 2.2.1 | 5 | | |

Use case: Search Article 5

2.2.2 6

Use case: Submit Article 6

2.2.3 7

Use case: Submit Review 7

2.2.4 7

Use case: Update Author 8

Use case: Update Reviewer 9

Use case: Update Article 9

Use case: Receive Article 10

Use case: Assign Reviewer 11

Use case: Receive Review 11

Use case: Check Status 12

Use case: Send Response 12

Use case: Send Copyright 13

Use case: Remove Article 14

Use case: Publish Article 14

|  |  |  |  |
| --- | --- | --- | --- |
| 2.3 | 16 | | |
| 2.4 | 16 | | |
| 3.0. | **Error! Bookmark not defined.** | | |
| 3.1 | 18 | | |
| 3.2 | 18 | | |
| 3.2.1 | | 18 |  |
| 3.2.2 | | 19 |  |
| 3.2.3 | | 19 |  |
| 3.2.4 | | 20 |  |
| 3.2.5 | | **Error! Bookmark not defined.** |  |
| 3.2.6 | | 21 |  |
| 3.2.7 | | 21 |  |
| 3.2.8 | | 22 |  |
| 3.2.9 | | 22 |  |
| 3.2.10 | | 23 |  |
| 3.2.11 | | 23 |  |
| 3.2.12 | | 24 |  |
| 3.3 24 | |  |  |
| 3.3.1 | | 24 |  |
| 3.3.2  Index | | 26 | 26 |

# List of Figures

Figure 1 - System Environment 4

Figure 2 - Article Submission Process 6

Figure 3 - Editor Use Cases 8

Figure 4 - Logical Structure of the Article Manager Data 23

* 1. **Introduction**

#### Purpose

The purpose of this document is to present a detailed description of the Furniture Rental Store 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.

#### Scope of Project

This software system will be a Furniture Rental Store System on an online platform for mass consumers. This system will be designed to allow any User to make an account on the platform, sign in, browse the catalog for required furniture, rent it or loan it, and pay the fee after return.

This system is also designed to allow an Admin account to add and delete furniture and their details to an active catalog, see details of all rented and loaned furniture, and keep track of the investment and profit from the venture. The software will facilitate notifications to Admins due to shortage in inventory. The system also contains a relational database containing a list of Users, Admins, and Furniture.

#### Glossary

|  |  |
| --- | --- |
| **Term** | **Definition** |
| Active Article | The document that is tracked by the system; it is a narrative  that is planned to be posted to the public website. |
| Author | Person submitting an article to be reviewed. In case of  multiple authors, this term refers to the *principal author*, with whom all communication is made. |
| Database | Collection of all the information monitored by this system. |
| Editor | Person who receives articles, sends articles for review, and  makes final judgments for publications. |
| Field | A cell within a form. |
| Historical Society Database | The existing membership database (also HS database). |
| Member | A member of the Historical Society listed in the HS database. |
| Reader | Anyone visiting the site to read articles. |
| Review | A written recommendation about the appropriateness of an  article for publication; may include suggestions for improvement. |
| Reviewer | A person that examines an article and has the ability to recommend approval of the article for publication or to  request that changes be made in the article. |
| 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 | Reviewer or Author. |

* 1. ***References***

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

#### 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

#### System Environment



Retail Website

User



bInventory Manager

nDatabase

bAdmin

**Figure 1 - System Environment**

The Furniture Rental Store System has two classes of active actors and one cooperating system. The Users access the Retail Website through the Internet. The Admins can access the system directly and have mechanisms to see the rentee of any furniture and the ability to manage inventory. Existing database may be linked.

The Store System can be divided into two component parts, the Retail Website itself, and the inventory manager accessible only to Admins.

#### Functional Requirements Specification

This section outlines the use cases for each of the active readers separately. The reader, the author and the reviewer have only one use case apiece while the editor is main actor in this system.

* + 1. **Admin specific use cases**

Use case:

**2.2.1.1 Create Product**

**Diagram:**



Create Product

ADMIN PRODUCT

### Brief Description

The Admin initializes the template for a new product to add to the inventory, and fills in details.

### Initial Step-By-Step Description

Before this use case can be initiated, the Admin has already accessed their account in the Furniture Rental Store System.

1. The Admin taps on the area containing their username from the Navbar.
2. The System reveals the contents of a dropdown menu.
3. The Admin selects Dashboard from the options.
4. The System displays various possible options.
5. The Admin clicks the “Add Product” option.
6. The System displays a blank template for filling in details.
7. The Admin fills in the requisite details and clicks “Add Product” button.
8. The System adds product to the inventory.

**Xref:** Sectio

Use case:

**2.2.1.2 View Product**

**Diagram:**



View Product

Admin

### Brief Description

The Admin views the details of a particular product.

### Initial Step-By-Step Description

Before this use case can be initiated, the Admin has already accessed their account in the Furniture Rental Store System.

1. The Admin taps on the area containing their username from the Navbar.
2. The System reveals the contents of a dropdown menu.
3. The Admin selects Dashboard from the options.
4. The System lists all available furniture as a list.
5. The Admin clicks “View Details” button of desired product.
6. The System displays details of product.
7. The Admin fills in the requisite details and clicks “Add Product” button.
8. The System adds product to the inventory.

**Xref:** Section 3.2.1, View Product

Use case:

**2.2.1.3 Update Product**

**Diagram:**



UpdateProduct

ADMIN PRODUCT

### Brief Description

The Admin initializes the template for a new product to add to the inventory, and fills in details.

### Initial Step-By-Step Description

Before this use case can be initiated, the Admin has already accessed their account in the Furniture Rental Store System.

1. The Admin taps on the area containing their username from the Navbar.
2. The System reveals the contents of a dropdown menu.
3. The Admin selects Dashboard from the options.
4. The System lists all available furniture as a list.
5. The Admin clicks the button having edit symbol of desired product.
6. The System displays current details of product.
7. The Admin edits the requisite details and clicks “Add Product” button.
8. The System updates product details.

**Xref:** Section 3.2.1, Create Product

Use case:

**2.2.1.4 Delete Product**

**Diagram:**



Delete Product

Admin

### Brief Description

The Admin deletes product from the inventory.

### Initial Step-By-Step Description

Before this use case can be initiated, the Admin has already accessed their account in the Furniture Rental Store System.

1. The Admin taps on the area containing their username from the Navbar.
2. The System reveals the contents of a dropdown menu.
3. The Admin selects Dashboard from the options.
4. The System displays various possible options.
5. The Admin clicks the “Add Product” option.
6. The System displays a blank template for filling in details.

**Xref:** Sectio

Use case:

**2.2.1.5 View Investment / Revenue**

**Diagram:**



View investment/

Revenue

Admin

### Brief Description

The Admin views the details of a particular product.

### Initial Step-By-Step Description

Before this use case can be initiated, the Admin has already accessed their account in the Furniture Rental Store System.

1. The Admin taps on the area containing their username from the Navbar.
2. The System reveals the contents of a dropdown menu.
3. The Admin selects Dashboard from the options.
4. The System lists all options, including count of investment and revenue
5. The Admin views these fields

**Xref:** Section 3.2.1, View Product

Use case:

**2.2.1.6 View Notifications**

**Diagram:**



View Notifications

Admin

### Brief Description

The Admin views the details of a particular product.

### Initial Step-By-Step Description

Before this use case can be initiated, the Admin has already accessed their account in the Furniture Rental Store System.

1. The Admin taps on the area containing their username from the Navbar.
2. The System reveals the contents of a dropdown menu.
3. The Admin selects Dashboard from the options.
4. The System lists all options.
5. The Admin clicks on “Notifications” option.
6. The System shows all notifications.

**Xref:** Section 3.2.1, View Product

Use case:

**2.2.1.7 Control Rent Requests**

**Diagram:**



Control Rent

Requests

ADMIN CUSTOMER

### Brief Description

The Admin views the details of a particular product.

### Initial Step-By-Step Description

Before this use case can be initiated, the Admin has already accessed their account in the Furniture Rental Store System.

1. The Admin taps on the area containing their username from the Navbar.
2. The System reveals the contents of a dropdown menu.
3. The Admin selects Dashboard from the options.
4. The System lists all options.
5. The Admin clicks on “Rent Request” option.
6. The System displays all Rent Requests.
7. The Admin approves or rejects the requests.
8. The System processes accordingly and notifies the User (awaiting delivery if approved).
9. If the Admin approves request, they can go to “Rent Delivery” under Dashboard options and release Product to Customer by clicking “Shipped” button.
10. The System updates status to Customer.

**Xref:** Section 3.2.1, View Product

* + 1. **Customer Activity Use Cases**

Use case:

**2.2.2.1 Search Product**

**Diagram:**



Search Product

## Customer

### Brief Description

The Customer searches for a specific product or type through the Search Bar.

### Initial Step-By-Step Description

Before this use case can be initiated, the Customer must have already logged in through their account.

1. If Customer cannot find product in home page, they can type the name of product or the type in the Search Bar present in the Navbar and click Enter.
2. The System processes the queries and displays all searches.

**Xref:** Section 3.2.2, Communicate

Use case:

**2.2.2.2 View Product Details**

**Diagram:**



View Product Details

## Customer

### Brief Description

The Customer views all details of a product description.

### Initial Step-By-Step Description

Before this use case can be initiated, the Customer must have already logged in through their account.

1. Either through searching or on the homepage, the Customer can find a particular Product overview.
2. The Customer taps on required Product overview.
3. The System retrieves all Product information and displays on screen.
4. The Customer views required details.

**Xref:** Section 3.2.2, Communicate

Use case:

**2.2.2.3 Rent Product**

**Diagram:**



Rent Product

CUSTOMER ADMIN

### Brief Description

The Customer submits request for renting desired Product.

### Initial Step-By-Step Description

Before this use case can be initiated, the Customer must have already logged in through their account.

1. Either through searching or on the homepage, the Customer can find a particular Product overview.
2. The Customer taps on required Product overview.
3. The System retrieves all Product information and displays on screen.
4. The Customer clicks on “Rent Now” button.
5. The System sends Rent request to Admin.
6. Admin must accept or reject Request. In either case, Customer is notified (and furniture is rented if approved.)
7. Further, Admin must approve shipping to release Product to Customer.
8. If rejected, Customer can click on “Try Again” for sending reapproval.

**Xref:** Section 3.2.2, Communicate

Use case:

**2.2.2.4 View Rent History**

**Diagram:**



View Rent History

## Customer

### Brief Description

The Customer views their history of renting products.

### Initial Step-By-Step Description

Before this use case can be initiated, the Customer must have already logged in through their account.

1. The Customer taps on the area containing their username from the Navbar.
2. The System reveals the contents of a dropdown menu.
3. The Customer selects My Rent Products from the options.
4. The System retrieves all information about the products rented by the user and displays them in chronological order.

**Xref:** Section 3.2.2, Communicate

Use case:

**2.2.2.5 Cancel Renting**

**Diagram:**



Cancel Renting

## Customer

### Brief Description

The Customer views their history of renting products.

### Initial Step-By-Step Description

Before this use case can be initiated, the Customer must have already logged in through their account.

1. The Customer taps on the area containing their username from the Navbar.
2. The System reveals the contents of a dropdown menu.
3. The Customer selects My Rent Products from the options.
4. The System retrieves all information about the products rented by the Customer or pending Rent.
5. The Customer can click on “Cancel” of such a product pending approval.
6. The System removes Product from pending approval.

**Xref:** Section 3.2.2, Communicate

Use case:

**2.2.2.6** **Cancel Request**

**Diagram:**



Cancel Request

## Customer

### Brief Description

The Customer views their history of renting products.

### Initial Step-By-Step Description

Before this use case can be initiated, the Customer must have already logged in through their account.

1. The Customer taps on the area containing their username from the Navbar.
2. The System reveals the contents of a dropdown menu.
3. The Customer selects My Rent Products from the options.
4. The System retrieves all information about the products rented by the Customer or pending Rent.
5. The Customer can click on “Cancel” of such a product pending approval.
6. The System removes Product from pending approval.

**Xref:** Section 3.2.2, Communicate

Use case:

**2.2.2.7 Return / Report Damage**

**Diagram:**



Return / Report

Damaged

## Customer

### Brief Description

The Customer returns rented Product or reports it as Damaged (which automatically returns it).

### Initial Step-By-Step Description

Before this use case can be initiated, the Customer must have already logged in through their account.

1. The Customer taps on the area containing their username from the Navbar.
2. The System reveals the contents of a dropdown menu.
3. The Customer selects My Rent Products from the options.
4. The System retrieves all information about the products rented by the Customer or pending Rent.
5. The Customer can click on “Return” or “Report Damaged” buttons.
6. If the Customer selects “Return”, System will update status and return Product to inventory.
7. If the Customer selects “Report Damaged”, System will update status and remove product from inventory.

**Xref:** Section 3.2.2, Communicate

Use case:

**2.2.2.8 View Bill**

**Diagram:**



View Bill

## Customer

### Brief Description

The Customer views total bill for all products.

### Initial Step-By-Step Description

Before this use case can be initiated, the Customer must have already logged in through their account.

1. The Customer taps on the area containing their username from the Navbar.
2. The System reveals the contents of a dropdown menu.
3. The Customer selects My Rent Products from the options.
4. The System retrieves all information about the products rented by the Customer or pending Rent.
5. The Customer can click on “Return” or “Report Damaged” buttons.
6. If the Customer selects “Return”, System will update status and return Product to inventory.
7. If the Customer selects “Report Damaged”, System will update status and remove product from inventory.

**Xref:** Section 3.2.2, Communicate

* + 1. **Use Cases related to socials**

The Editor has the following sets of use cases:



Update

Handle

## Edito

Ck

Send

Publish

**Figure 3 - Editor Use Cases**

Use case:

**2.2.3.1 Create Account**

**Diagram:**



Create Account

## Person (Customer here)

### Brief Description

The Person (here, only Customer) creates an account by supplying required details.

### Initial Step-By-Step Description

Before this use case can be initiated, Person must be on home page of website.

1. The Person selects Dropdown menu on the Navbar.
2. The System displays some more option.
3. The Person can click either “Login” or “Register”.
4. If the Person selects “Register”, System displays a window with details to fill.
5. The Person fills details and clicks “Register” button.
6. If System approves, Person ends up in homepage with profile.
7. If Person selects “Login”, it opens a window, which has an option for redirecting to “Register”.

**Xref:** Section 3.2.3, Add Author; Section 3.2.5 Update Person

Use case:

**2.2.3.2 Account Login**

**Diagram:**



Log Into Account

## Person

### Brief Description

The Person logs into account by supplying username and password.

### Initial Step-By-Step Description

Before this use case can be initiated, Person must be on home page of website.

1. The Person selects Dropdown menu on the Navbar.
2. The System displays some more option.
3. The Person clicks on “Login”.
4. The System opens a window, which has options for entering username and password.
5. Person enters username and password in required fields.
6. If System can authenticate, it opens homepage with profile.

**Xref:** Section 3.2.3, Add Author; Section 3.2.5 Update Person

Use case:

**2.2.3.3 Edit Profile**

**Diagram:**



Create Account

## Person

### Brief Description

The Person logs into account by supplying username and password.

### Initial Step-By-Step Description

Before this use case can be initiated, Person must be on home page of website logged in.

1. The Person taps on the area containing their username from the Navbar.
2. The System reveals the contents of a dropdown menu.
3. The Person selects Profile from the options.
4. The System lists all details of the profile, along with 3 buttons.
5. If Person selects “Edit Profile” button, System opens a window with input fields of Name, Username and Email for Person to edit.
6. If Person clicks Update, System updates details.
7. If Person selects “Edit Billing Address” button, System opens a window with input fields of Phone and Delivery Address to edit.
8. If Person clicks Update, System updates details.
9. If Person selects “Change Password” button, System opens a window with input fields of Old Password, New Password and Confirm password to enter
10. If Person clicks Change Password, system authenticates and confirms new passwords.

**Xref:** Section 3.2.3, Add Author; Section 3.2.5 Update Person

Use case:

**2.2.3.4 Create / Delete Comments**

**Diagram:**



Create / Delete

Comments

PERSON PRODUCT

### Brief Description

The Person creates or deletes comment on pages of requires Products.

### Initial Step-By-Step Description

Before this use case can be initiated, Person must be on home page of website.

1. Either through searching or on the homepage, the Person can find a particular Product overview.
2. The Person taps on required Product overview.
3. The System retrieves all Product information and displays on screen. In the window, the bottom contains options for viewing and leaving comments.
4. In the Leave a Comment section, Person can type a comment in the field and click “Comment” button.
5. The System updates comments.
6. Below that section, the Person can see posted comments. The Person can click on “Delete” option of their posted comment.
7. The System removes comment.

**Xref:** Section 3.2.3, Add Author; Section 3.2.5 Update Person

Use case:

**2.2.3.5 Logout**

**Diagram:**



Logout

## Person

### Brief Description

The Person creates or deletes comment on pages of requires Products.

### Initial Step-By-Step Description

Before this use case can be initiated, Person must be on home page of website.

1. The Person taps on the area containing their username from the Navbar.
2. The System reveals the contents of a dropdown menu.
3. The Person selects Logout from the options.
4. System opens into window without profile.

**Xref:** Section 3.2.3, Add Author; Section 3.2.5 Update Person

#### 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.

#### 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

#### 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.

#### Functional Requirements

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

* + 1. Create Product

|  |  |
| --- | --- |
| **Use Case Name** | Create Product |
| **XRef** | Section 2.2.2.1 Create Product |
| **Trigger** | The Admin clicks on Add Product under Dashboard. |
| **Precondition** | Dashboard options are open. |
| **Basic Path** | 1. The Admin clicks the “Add Product” option. 2. The System displays a blank template for filling in details. 3. The Admin fills in the requisite details and clicks “Add Product” button. |

|  |  |
| --- | --- |
| **Alternative Paths** |  |
| **Postcondition** | Product is added to inventory. |
| **Exception Paths** | The Reader may abandon the creation at any time. |
| **Other** |  |

* + 1. View and Modify Products

|  |  |
| --- | --- |
| **Use Case Name** | View Product, Update Product, Delete Product |
| **XRef** | Section 2.2.1.2, View Product; Section 2.2.1.3, Update Product;  Section 2.2.1.4, Delete Product |
| **Trigger** | The user clicks on one of three buttons in Product Overview. |
| **Precondition** | The user is on desired Product Overview. |
| **Basic Path** | 1. The system retrieves product information if user clicks View Details. 2. For other two buttons, System updates information of products (edit or deletion.) |
| **Alternative Paths** |  |
| **Postcondition** | Product details may be changed. |
| **Exception Paths** | The attempt may be abandoned at any time. |
| **Other** | None |

## View Activities

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

* + 1. Add Reviewer

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

## 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. |

|  |  |
| --- | --- |
|  | 1. The Editor selects a person to update. 2. The system presents the database information in grid form for modification. 3. The Editor updates the information and submits the form. 4. 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. |

* + 1. Update Article Status

|  |  |
| --- | --- |
| **Use Case Name** | Update Article Status |
| **XRef** | Section 2.2.4, Update Article SDD, Section 7.6 |
| **Trigger** | The Editor selects to update the status of an article in the database. |
| **Precondition** | The Editor has accessed the Article Manager main screen and the article is already in the database. |
| **Basic Path** | 1. The system creates and presents an alphabetical list of all active articles. 2. The Editor selects the article to update. 3. The system presents the information about the article in grid format. 4. The Editor updates the information and resubmits the form. |
| **Alternative Paths** | In step 4, the use case *Enter Communication* may be invoked. |
| **Postcondition** | The database has been updated. |
| **Exception Paths** | If the article 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 can be used to add categories for an article, to correct typographical errors, or to remove a reviewer who has missed a deadline for returning a review. It may also be used to allow access to the named use case to enter an updated  article or a review for an article. |

## Enter Communication

|  |  |
| --- | --- |
| **Use Case Name** | Enter Communication |
| **XRef** | Section 2.2.4, Receive Article; Section 2.2.4, Receive Review  SDD, Section 7.7 |
| **Trigger** | The Editor selects to add a document to the system. |

|  |  |
| --- | --- |
| **Precondition** | The Editor has accessed the Article Manager main screen and  has the file of the item to be entered available. |
| **Basic Path** | 1. The Editor selects the article using the *3.2.6, Update Article Status* use case. 2. The Editor attaches the file to the grid presented and updates the respective information about the article. 3. When the Editor updates the article status to indicate that a   review is returned, the respective entry in the Reviewer table is updated. |
| **Alternative Paths** | None |
| **Postcondition** | The article entry is updated in the database. |
| **Exception Paths** | The Editor may abandon the operation at any time. |
| **Other** | This use case extends *3.2.6, Update Article Status* |

* + 1. Assign Reviewer

|  |  |
| --- | --- |
| **Use Case Name** | Assign Reviewer |
| **XRef** | Section 2.2.4, Assign Reviewer  SDD, Section 7.8 |
| **Trigger** | The Editor selects to assign a reviewer to an article. |
| **Precondition** | The Editor has accessed the Article Manager main screen and the article is already in the database. . |
| **Basic Path** | 1. The Editor selects the article using the *3.2.6, Update Article Status* use case. 2. The system presents an alphabetical list of reviewers with their information. 3. The Editor selects a reviewer for the article. 4. The system updates the article database entry and emails the reviewer with the standard message and attaches the text of the article without author information. 5. The Editor has the option of repeating this use case from step 2. |
| **Alternative Paths** | None. |
| **Postcondition** | At least one reviewer has been added to the article information and the appropriate communication has been sent. |
| **Exception Paths** | The Editor may abandon the operation at any time. |
| **Other** | This use case extends *3.2.6, Update Article Status.* The Editor,  prior to implementation of this use case, will provide the message text. |

## Check Status

|  |  |
| --- | --- |
| **Use Case Name** | Check Status |
| **XRef** | Section 2.2.4, Check Status  SDD, Section 7.9 |
| **Trigger** | The Editor has selected to check status of all active articles. |
| **Precondition** | The Editor has accessed the Article Manager main screen. |

|  |  |
| --- | --- |
| **Basic Path** | 1. The system creates and presents a list of all active articles organized by their status. 2. The Editor may request to see the full information about an   article. |
| **Alternative Paths** | None. |
| **Postcondition** | The requested information has been displayed. |
| **Exception Paths** | The Editor may abandon the operation at any time. |
| **Other** | The editor may provide an enhanced list of status later. At present, the following categories must be provided:   1. Received but no further action taken 2. Reviewers have been assigned but not all reviews are returned (include dates that reviewers were assigned and order by this criterion). 3. Reviews returned but no further action taken. 4. Recommendations for revision sent to Author but no response as of yet. 5. Author has revised article but no action has been taken. 6. Article has been accepted and copyright form has been sent. 7. Copyright form has been returned but article is not yet published.   A published article is automatically removed from the active article list. |

* + 1. Send Communication

|  |  |
| --- | --- |
| **Use Case Name** | Send Communication |
| **XRef** | Section 2.2.4, Send Response; Section 2.2.4, Send Copyright  SDD, Section 7.10 |
| **Trigger** | The editor selects to send a communication to an author. |
| **Precondition** | The Editor has accessed the Article Manager main screen. |
| **Basic Path** | 1. The system presents an alphabetical list of authors. 2. The Editor selects an author. 3. The system invokes the Editor’s email system entering the author’s email address into the *To:* entry. 4. The Editor uses the email facility. |
| **Alternative Paths** | None. |
| **Postcondition** | The communication has been sent. |
| **Exception Paths** | The Editor may abandon the operation at any time. |
| **Other** | The standard copyright form will be available in the Editor’s directory for attaching to the email message, if desired. |

## Publish Article

|  |  |
| --- | --- |
| **Use Case Name** | Publish Article |
| **XRef** | Section 2.2.4, Publish Article SDD, Section 7.11 |

|  |  |
| --- | --- |
| **Trigger** | The Editor selects to transfer an approved article to the Online  Journal. |
| **Precondition** | The Editor has accessed the Article Manager main screen. |
| **Basic Path** | 1. The system creates and presents an alphabetical list of the active articles that are flagged as having their copyright form returned. 2. The Editor selects an article to publish. 3. The system accesses the Online Database and transfers the article and its accompanying information to the Online Journal database. 4. The article is removed from the active article database. |
| **Alternative Paths** | None. |
| **Postcondition** | The article is properly transferred. |
| **Exception Paths** | The Editor may abandon the operation at any time. |
| **Other** | Find out from the Editor to see if the article information should be archived somewhere. |

* + 1. Remove Article

|  |  |
| --- | --- |
| **Use Case Name** | Remove Article |
| **XRef** | Section 2.2.4, Remove Article SDD, Section 7.12 |
| **Trigger** | The Editor selects to remove an article from the active article database. |
| **Precondition** | The Editor has accessed the Article Manager main screen. |
| **Basic Path** | 1. The system provides an alphabetized list of all active articles. 2. The editor selects an article. 3. The system displays the information about the article and requires that the Editor confirm the deletion. 4. The Editor confirms the deletion. |
| **Alternative Paths** | None. |
| **Postcondition** | The article is removed from the database. |
| **Exception Paths** | The Editor may abandon the operation at any time. |
| **Other** | Find out from the Editor to see if the article and its information  information should be archived somewhere. |

#### Detailed Non-Functional Requirements

* + 1. Logical Structure of the Data

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

sent

ha

**Figure 4 - Logical Structure of the Article Manager Data**

writ

Review

Reviewer

writ

Article

Author

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 |

* + 1. 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.

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,

### Index

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,

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

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

Database, 2, 9, 11, 14, 15, 16, 17, 18, 19, 20, update, 9, 11, 20, 21

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

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