**Software Requirements Specification**

**Version 1.0**

**<<Annotated Version>>**

**June 25, 2021**

**Hotel Management System**

**\*Bayan khattab<3>**

**\*Jofran Alrfaie<2>**

**\*Khadija Ahme<1>**

<<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](#_Toc77487619)

[List of Figures ii](#_Toc77487620)

[1.0. Introduction 1](#_Toc77487621)

[1.1. Purpose 1](#_Toc77487622)

[1.2. Scope of Project 1](#_Toc77487623)

[1.3. Glossary 2](#_Toc77487624)

[1.4. References 2](#_Toc77487625)

[1.5. Overview of Document 2](#_Toc77487626)

[2.0. Overall Description 4](#_Toc77487627)

[2.1 System Environment 4](#_Toc77487628)

[2.2 Functional Requirements Specification 5](#_Toc77487629)

[2.2.1 Reader Use Case 5](#_Toc77487630)

[Use case: Search Article 5](#_Toc77487631)

[2.2.2 Author Use Case 6](#_Toc77487632)

[Use case: Submit Article 6](#_Toc77487633)

[2.2.3 Reviewer Use Case 7](#_Toc77487634)

[Use case: Submit Review 7](#_Toc77487635)

[2.2.4 Editor Use Cases 8](#_Toc77487636)

[Use case: Update Author 8](#_Toc77487637)

[Use case: Update Reviewer 9](#_Toc77487638)

[Use case: Update Article 9](#_Toc77487639)

[Use case: Receive Article 10](#_Toc77487640)

[Use case: Assign Reviewer 11](#_Toc77487641)

[Use case: Receive Review 11](#_Toc77487642)

[Use case: Check Status 12](#_Toc77487643)

[Use case: Send Response 12](#_Toc77487644)

[Use case: Send Copyright 13](#_Toc77487645)

[Use case: Remove Article 14](#_Toc77487646)

[Use case: Publish Article 14](#_Toc77487647)

[2.3 User Characteristics 15](#_Toc77487648)

[2.4 Non-Functional Requirements 15](#_Toc77487649)

[3.0. Requirements Specification 17](#_Toc77487650)

[3.1 External Interface Requirements 17](#_Toc77487651)

[3.2 Functional Requirements 17](#_Toc77487652)

[3.2.1 Search Article 17](#_Toc77487653)

[3.2.2 Communicate 18](#_Toc77487654)

[3.2.3 Add Author 18](#_Toc77487655)

[3.2.4 Add Reviewer 19](#_Toc77487656)

[3.2.5 Update Person 19](#_Toc77487657)

[3.2.6 Update Article Status 20](#_Toc77487658)

[3.2.7 Enter Communication 20](#_Toc77487659)

[3.2.8 Assign Reviewer 21](#_Toc77487660)

[3.2.9 Check Status 21](#_Toc77487661)

[3.2.10 Send Communication 22](#_Toc77487662)

[3.2.11 Publish Article 22](#_Toc77487663)

[3.2.12 Remove Article 23](#_Toc77487664)

[3.3 Detailed Non-Functional Requirements 23](#_Toc77487665)

[3.3.1 Logical Structure of the Data 23](#_Toc77487666)

[3.3.2 Security 25](#_Toc77487667)

[Index 26](#_Toc77487668)

# List of Figures

[Figure 1 - System Environment 4](#_Toc77487669)

[Figure 2 - Article Submission Process 6](#_Toc77487670)

[Figure 3 - Editor Use Cases 8](#_Toc77487671)

[Figure 4 - Logical Structure of the Article Manager Data 23](#_Toc77487672)

# Introduction

The Hotel Management System is a tool for booking the rooms of Hotel through online process by the Customer. It provides the proper management tools and ease of access to the Customer Information. ..

## 1.1. Purpose

The Software Requirements Speciﬁcation (SRS) will provide a detailed description of the requirements for the Hotel Management System (HMS). This SRS will allow for a complete understanding of what is to be expected from the newly introduced system which is to be constructed. The clear understanding of the system and its' functionality will allow for the correct software to be developed for the end user and will be used for the development of the future stages of the project. This SRS will provide the foundation for the project. From this SRS, the Hotel Management System can be designed, constructed, and ﬁnally tested. This SRS will be used by the system development team which is constructing the HMS and the hotel end users. The Project team will use the SRS to fully understand the expectations of this HMS to construct the appropriate software. The hotel end users will be able to use this SRS as a "test" to see if the constructing team will be constructing the system to their expectations. If it is not to their expectations the end users can specify how it is not to their liking and the team will change the SRS to ﬁt the end users' needs.

## 1.2. Scope of Project

The introducing software, Hotel Management System which is going to be implemented for Hotel Dayal will automate the major operations of the hotel. The Reservation System is to keep track in room and hall reservation and check availability. The Room Management System is for manage all room types room services. The Inventory Control System will keep track in all inventories of the hotel and guest details will handled by guest management. Administration department will monitor the all. There is three End Users for HMS. The End Users Are Owner, Manager and Receptionist. Owner can access to all system functionalities without any restrictions. Manager can access to all system functionalities with limited restrictions. Receptionist can only access to the Reservation management section. To keep restrictions for each End User levels HMS can create different Login functions. The objectives of the automated Hotel Management System is to simplify the day to day processes of the hotel. The system will be able to handle many services to take care of all customers in a quick manner. As a solution to the large amount of ﬁle handling happening at the hotel, this software will be used to overcome those drawbacks. Safety, easiness of using and most importantly the efﬁciency of information retrieval are some beneﬁts the development team going to present with this system. The system should be user appropriate, easy to use, provide easy recovery of

appropriate, easy to use, provide easy recovery of errors and have an overall end user high subjective satisfaction.

## 1.3. Glossary

|  |  |
| --- | --- |
| **Term** | **Definition** |
| SRS | Software Requirements Speciﬁcation ..  . |
| HMS | Hotel Management System .. |
| USR | Reviewer or Author |
| Editor | Person who receives articles, sends articles for review, and makes final judgments for publications. |
| Field | A cell within a form. |
| **End users** | The people who will be actually using the system SQL Structural Query Language .. |
| 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. |

## 1.4. References

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

## 1.5. Overview of Document

. This SRS is organized into two parts the ﬁrst is the overall description and the second section is the speciﬁc requirement. The overall description will describe the requirement of Hotel Management System. The speciﬁc requirement section describes the detail of the system.

# 2.0. Overall Description

## 2.1 System Environment

admin

customer

admin

Online website

hotel Manager

Hotel System

employee

Figure 1 - System Environment

***2.1 Flow chart***

The booking System has three active actors and one cooperating system. The customer, admin, or hotel admin accesses the online booking through the Internet. Any admin communication with the system is through control panel. The customer accesses the entire system directly.

***2.2 Functional Requirements Specification***

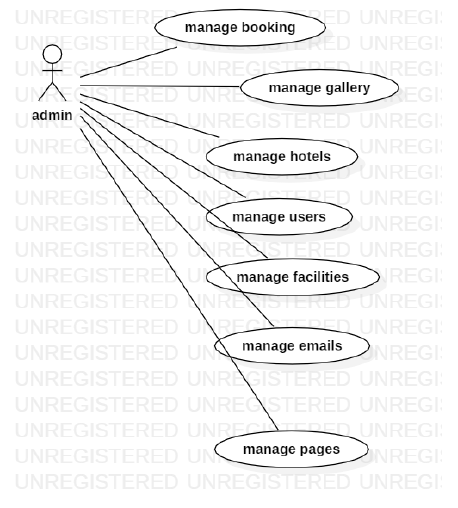
This section outlines the use cases for each of the active readers separately. The reader, the author and the reviewer have only one use case apiece while the editor is main actor in this system.

2.2.1: admin Use Case

In case of multiple admin, this term refers to the *principal admin*, with whom all communication is made.

Use case: admin use case

**Diagram:**

****

**Brief Description**

The admin confirms the booking.

**Initial Step-By-Step Description**

Before this use case can be initiated, the admin has already connected to the Online Booking System.

1. The admin chooses the *reservation to confirm it*.

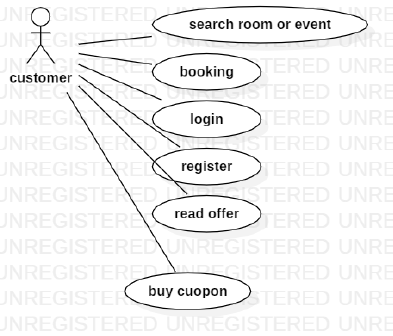
2. The admin click on confirm the email will send to customer that your room confirmed.

3. The System generates and sends an email acknowledgement.

Xref: section 3.2.1, admin hotel

2.2.2: Use case: **Customer**

Diagram:



**Brief Description**

The customer accesses the Online Booking Website, searches for an room and reserve it.

**Initial Step-By-Step Description**

Before this use case can be initiated, the customer has already accessed the Online booking system.

8. The customer chooses to search by price, location, or keyword.

9. The system displays the choices to the Customer.

10. The customer selects the room or event desired.

11. The system presents the details of the order to the customer.

**Xref:** section 3.2.2, add customer;

### 2.2.3 Employee Use Case

**Diagram:**

Employee

Add customer

**Brief Description**

The confirms the booking. .

**Initial Step-By-Step Description**

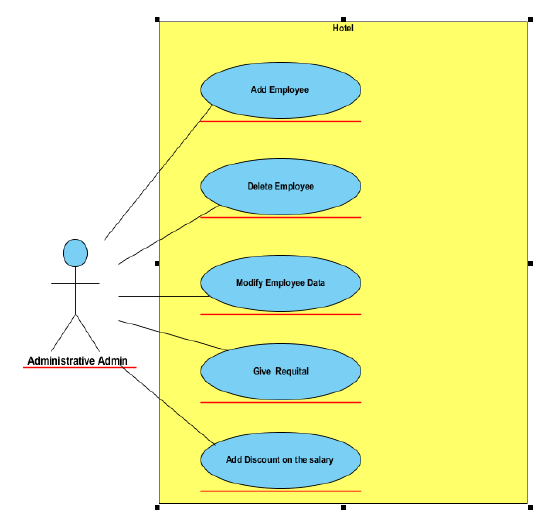
Before this use case can be initiated, the Employee has already connected to the Online hotel Website.

1. The Employee chooses the *Email Editor* button.
2. The System uses the *sendto* HTML tag to bring up the user’s email system.
3. The Employee fills in the Subject line and attaches the file as directed and emails it.
4. The System generates and sends an email acknowledgement.

**Xref:** Section 3.2.3, Employee

2.2.4: administrative admin use case

**Diagram:**



**Brief Description**

# Services provided to hotel managers: Administrative admin.

**Initial Step-By-Step Description**

# His mission is to manage the personnel affairs of the hotel, In addition to the hotel’s financial affairs department, he has the following responsibilities:

# 1- Add an employee to the working staff.

# 2- Dismissing an employee from his work.

# 3- Amending the employee's information when necessary.

# 4- Imposing penalties on an employee.

# 5- Granting a reward to an employee.

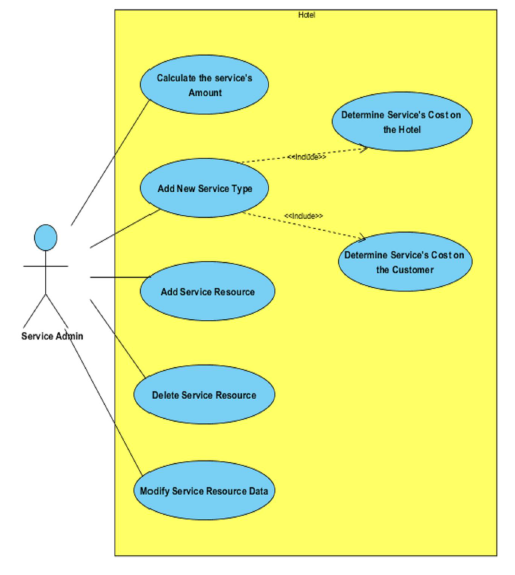
# 6- Calculating salaries of employees and disbursing them after taking into consideration (rebates) And employee rewards).

# 7- one account paid by the hotel to secure its requirements and pay taxes.

Xref: section 3.2.4 , adimistrative

2.2.5 : service admin use case

**Diagram:**



**Brief Description**

# Services provided to hotel managers : service admin

**Initial Step-By-Step Description :**

He is responsible for the services provided by the hotel and coordinating communication With the authorities that provide these services, and in detail, his mission will be:

# 1- Adding a new service at the hotel.

# 2- Calculating the available quantity of a service.

# 3- Add the service. Calculate the cost of the hotel towing

# 4- Determining the cost of the service that the customer incurs if he requests it.

# 5- Adding a new service agency to start dealing with it (company - factory - restaurant).

# 6 - Deleting a service agency the hotel was dealing with. G- Modifying the data of a service agency that the hotel deals with (location - phone numbers ..).

***Xref: section 3.2.5, service admin***

**2.3 User Characteristics :**

The customer is expected to be Internet literate and be able to use a search engine. The main screen of the Booking online system will have the search function and a link to “room details Information.”

The customer and admin hotel are expected to be Internet literate and to be able to use chat.

The admin 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

1. The system must ensure that all the transferable data as for examples customers credit or debit card number, CVV Code, e-payment should be done in secured connection.

2. The system must be able to handle multiple transactions a time.

3. The system must provide customers 24\*7 hours online booking service.

4. The system should support almost all the browsers (Internet Explorer, Safari, Chrome, and Firefox).

5. The system should be able to convert the price from R.S to USD.

6. System should send the newsletter about ongoing promotions or deal to registered customers.

7. Customers need to cancel the booking before 24 hrs. Otherwise their credit card will be charged for one day.

8. In promotion time the system will charge credit card promptly.

# 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 Search Article

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

### 3.2.2 Communicate

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

### 3.2.3 Add Author

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

### 3.2.4 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. |

### 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. |

### 3.2.6 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. |

### 3.2.7 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* |

### 3.2.8 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. |

### 3.2.9 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. |

### 3.2.10 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. |

### 3.2.11 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. |

### 3.2.12 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. |

## 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