Software Requirements Specification

Version 1.0

May 5/2021

Library Automation System

Mohammad Mohammad

Mohammad AL Mahmod

# Table of Contents

[Table of Contents 2](#_Toc43663402)

[List of Figures **Error! Bookmark not defined.**](#_Toc43663403)

[1.0. Introduction 3](#_Toc43663404)

[1.1. Purpose 3](#_Toc43663405)

[1.2. Scope of Project 3](#_Toc43663406)

[1.3. Glossary 4](#_Toc43663407)

[1.4. References 4](#_Toc43663408)

[1.5. Overview of Document 4](#_Toc43663409)

[2.0. Overall Description 5](#_Toc43663410)

[2.1 System Environment 5](#_Toc43663411)

[2.2 Functional Requirements Specification 5](#_Toc43663412)

[2.2.1 Library Employee Use Cases 6](#_Toc43663413)

[Use case: Login 6](#_Toc43663414)

[Use case**: Barcode scanning** 7](#_Toc43663415)

[Use case: **Record order** 8](#_Toc43663416)

[Use case**: give the stuff** 9](#_Toc43663417)

[Use case: **Accounting** 10](#_Toc43663418)

[2.3 User Characteristics 11](#_Toc43663419)

[2.4 Non-Functional Requirements 11](#_Toc43663420)

[3.0. Requirements Specification 11](#_Toc43663421)

[3.1 External Interface Requirements 11](#_Toc43663422)

[3.2 Functional Requirements 12](#_Toc43663423)

[3.2.1 **Login** 12](#_Toc43663424)

[3.2.2 **Barcode scanning** 12](#_Toc43663425)

[3.2.3 **Record order** 13](#_Toc43663426)

[3.2.4 **give stuff** 13](#_Toc43663427)

[3.2.5 **Accounting** **case** 14](#_Toc43663428)

[3.3 Detailed Non-Functional Requirements 14](#_Toc43663429)

[3.3.1 Logical Structure of the Data 14](#_Toc43663430)

[3.3.2 Security 17](#_Toc43663431)

# 1.0. Introduction

## 1.1. Purpose

The purpose of this document is to provide a detailed description of the library automation system. It will explain the purpose and features of the system, system interfaces, what the system will do, the constraints under which it must operate, and how the system will interact with external stimuli. This document is intended for both stakeholders and system developers and will be presented to Eng. Nahla Saad El Din for approval.

## 1.2. Scope of Project

This software system will act as an automation system for university libraries. This system will be designed to maximize library productivity by providing tools to help automate the quick search for lectures that the student did not purchase, which could have been purchased manually. By increasing the efficiency of the library's work and production, the system meets the student's needs while still being easy to understand and use.

The program will facilitate communication between the employee and the student via the student's electronic card. The system also contains a relational database containing a list of the student's name, university number, university year and university semester.

## 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 | 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

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

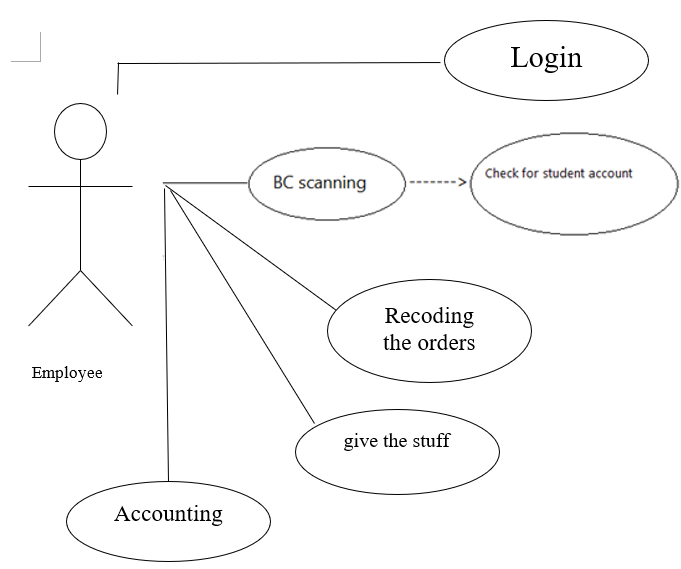


Figure 1 - System Environment

The library automation system consists of one activator and one cooperating system.

The employee accesses the entire system directly via his account.

## 2.2 Functional Requirements Specification

This section explains the use cases for the library employee. The employee is the main actor in this system

### 2.2.1 Library Employee Use Cases

#### Use case: Login

**Diagram:**

Employee

loyee

Login

**Brief Description**

Initially, the employee accesses the automation system by logging in.

**Initial Step-By-Step Description**

Before starting this use case, the employee logs in to the system by entering his account and password.

1. The employee enters his account and password.
2. The employee logs into the system.
3. The system converts the employee to the main page.

**Xref:** Section 3.2.1, Login.

Enter the account and password

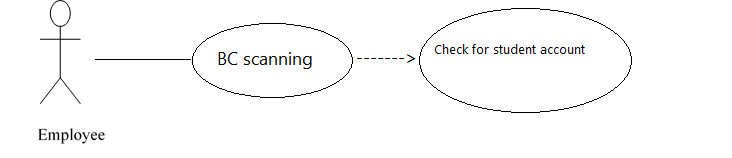
Open the system

Log in to the system

Figure 2 - Login Process

The library employee enters his account and password and then logs in to the system.

### Use case**: Barcode scanning**

 **Diagram:**

**Brief Description**

The employee scans the student's Barcode e-card.

**Initial Step-By-Step Description**

Before starting this use case, the student had already obtained an electronic card.

1. The employee scans the card's Barcode.
2. The system checks if there is an account for the student, in the event that there is an account that serves the student, and in the absence of an account, the system creates an account for the student.

**Xref:** Section 3.2.2, Barcode scanning

Create an account

Barcode scanning

Submit the card

Meet the needs of the student

Check for an account

The employee scans the barcode of the card and checks that the student has an account

(if it does not exist, a special account is created for him)

and then the student's need is met.

### Use case: **Record order**

**Diagram:**

Employee

Recoding the orders

**Brief Description**

Record the orders that student order it.

**Initial Step-By-Step Description**

Before this use case can be initiated, we must ensure that the student have account in the system.

1. The student will order stuff.
2. When he done the employee will add his orders to his account.
3. The student will go to accounting department.

**Xref:** Section 3.2.3, **Record order**



Create account

Student order stuff

Student in order department

Student go to accounting department

After student done with his orders the employee will check if he have account or not if he the employee will add his stuff to his account else the employee will create account for student and add his stuff .

### Use case**: give the stuff**

Employee

loyee

give the stuff

**Brief Description**

The student will have what he order it

**Initial Step-By-Step Description**

Before this use case can be initiated , the employee must record student orders .

1. The student get what he order it.
2. The student will go to accounting department

**Xref:** Section 3.2.4, **give stuff**







**Figure 5 – give the stuff**

When the student gives his requests to the employee, the student will get what he ask for, then he will go to accounting department.

### Use case: **Accounting**

Employee

loyee

Accounting

**Brief Description**

Student will pay for his orders (stuff) for the employee in Accounting department

**Initial Step-By-Step Description**

Before this use case can be initiated, the student must have orders.

1.The student get what he orders it.

2.If there is no line the student will go to Accountant directly.

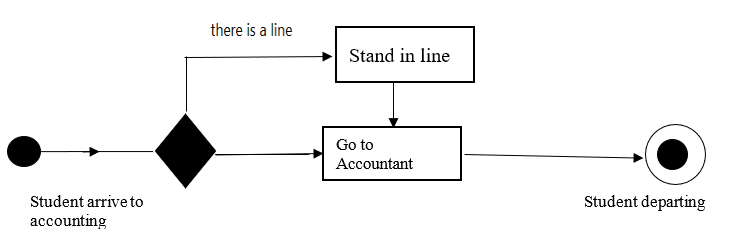
3.If there is a line the student will student in line.

4.when the student turn in line came, he will pay for his order.

5. The Accountant will be issuing an invoice to the student.

6. The student will depart the library.

**Xref:** Section 3.2.5, **Accounting case**

**** **Figure 6 - Accounting**

When the student arrives to accounting department, there is 2 case if there is a que or not, if there are he will stand in a queue, else he will go directory to the accountant, when he is done, he will depart the library

## 2.3 User Characteristics

It is expected that users of this system (students) will be well versed in the use of electronic systems, to maintain the confidentiality of their accounts.

The system administrator is expected to be fully familiar with the Windows operating system and be able to use peripherals such as the printer, barcode device, etc., and the ability to solve problems you may encounter in the business.

**2.4 Non-Functional Requirements**

Ease of use, availability during working hours 8 hours a day.

Backup every 8 hours, privacy and security of user data.

Accessibility, performance improvement, and system reuse.

Lectures printed color.

The system will run on the system administrator's computer and will contain a relational database, Access is already installed on this computer, which is the Windows operating system.

* 1. **Requirements Specification**

## 3.1 External Interface Requirements

## The system needs a barcode device to scan and enter via the electronic card where confirmation and verification of personal information to log in, in order to achieve high reliability and security of the system, as well as a printer needs to print requests immediately.

## 3.2 Functional Requirements

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

### **3.2.1 Login**

|  |  |
| --- | --- |
| **Use Case Name** | Login |
| **XRef** | Section 3.2.1, Login. |
| **Trigger** | The employee logs into the system. |
| **Precondition** | The employee has an account and a password. |
| **Basic Path** | 1. The employee enters his account and password. 2. The employee logs into the system. 3. The system converts the employee to the main page. |
| **Alternative Paths** | No alternative paths. |
| **Postcondition** | The employee's account is registered in the database. |
| **Exception Paths** | No exception paths. |
| **Other** | None. |

### 3.2.2 **Barcode scanning**

|  |  |
| --- | --- |
| **Use Case Name** | Barcode scanning |
| **XRef** | Section 3.2.2, Barcode scanning. |
| **Trigger** | Scan the card's barcode. |
| **Precondition** | Submit an e-card. |
| **Basic Path** | .1The employee scans the card's Barcode.  .2The system checks if there is an account for the student, in the event that there is an account that serves the student, and in the absence of an account, the system creates an account for the student. |
| **Alternative Paths** | No alternative paths. |
| **Postcondition** | The system records the orders containing the student's lectures. |
| **Exception Paths** | No exception paths. |
| **Other** | None |

### 3.2.3 **Record order**

|  |  |
| --- | --- |
| **Use Case Name** | Record order. |
| **XRef** | Section 3.2.3, Record order**.** |
| **Trigger** | Insert the orders in to student account. |
| **Precondition** | The student must order at least one item. |
| **Basic Path** | 1.The student will order stuff.  2.When he done the employee will add his orders to his account.  3.The student will go to receving department . |
| **Alternative Paths** | No alternative paths. |
| **Postcondition** | None. |
| **Exception Paths** | No exception paths. |
| **Other** | None. |

### 

### 3.2.4 **give stuff**

|  |  |
| --- | --- |
| **Use Case Name** | give stuff |
| **XRef** | Section 3.2.4, give stuff |
| **Trigger** | The student take what he order it |
| **Precondition** | The employee must record his order |
| **Basic Path** | The student get what he order it.  The student will go to accounting department |
| **Alternative Paths** | No alternative paths. |
| **Postcondition** | None. |
| **Exception Paths** | No exception paths. |
| **Other** | None. |

### 3.2.5 **Accounting** **case**

|  |  |
| --- | --- |
| **Use Case Name** | Accounting. |
| **XRef** | Section 3.2.5, Accounting case. |
| **Trigger** | The student arrive to accounting department. |
| **Precondition** | The student must done from recoding order case . |
| **Basic Path** | 1.The student get what he order it.  2.If there is no line the student will go to Accountant directly.  3.If there is a line the student will student in line.  4.when the student turn in line came he will pay for his order.  5. The Accountant will issuing an invoice to the student.  6. The student will depart the facility. |
| **Alternative Paths** | No alternative paths. |
| **Postcondition** | None. |
| **Exception Paths** | No exception paths. |
| **Other** | None. |

## 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 7 - 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.