Software Requirements Specification

Version 1.0

June 19, 2020

Student Credit Box System

Suzan Hazourri

Batoul Mohammad Ali

Batol Rashed

Doha Al Mahmoud

<<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](#_Toc77487624) 1

[1.4. References 2](#_Toc77487625)

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

[2.0. Overall Description](#_Toc77487627) 3

[2.1 System Environment](#_Toc77487628) 3

[2.2 Functional Requirements Specification](#_Toc77487629) 4

[2.2.1 Employee use case](#_Toc77487630) 4

[Use case: Getting a loan](#_Toc77487631) 4

[Use case: loan resumption 5](#_Toc77487631)

[2.2.2 Cashier Use Case](#_Toc77487632) 5

[Use case: pay off a loan 6](#_Toc77487633)

[Use case: Inquiry about the amount paid out 6](#_Toc77487633)

[2.2.3 Manager Use Case 7](#_Toc77487634)

[Use case: Stop granting the mothly loan 7](#_Toc77487635)

[2.3 User Characteristics](#_Toc77487648) 8

[3.0. Requirements Specification](#_Toc77487650) 9

[3.1 External Interface Requirements](#_Toc77487651) 9

[3.2 Functional Requirements](#_Toc77487652) 9

[3.2.1 get a loan](#_Toc77487653) 9

[3.2.2 loan resumption](#_Toc77487654) 10

[3.2.3 pay off the loan](#_Toc77487655) 10

[3.2.4 inquiry about the amount paid of a loan](#_Toc77487656) 11

[3.2.5 stop granting the monthly loan](#_Toc77487657) 11

[3.3 Detailed Non-Functional Requirements](#_Toc77487665) 12

[3.3.1 scalability](#_Toc77487666) 12

[3.3.2 availability](#_Toc77487667) 12

3.3.3 Reliability………………………………………………………………………………………..12

3.3.4 Capacity……………………………………………………………………………………….…12

# 1.0. Introduction

## 1.1. Purpose

This file aims to describe the system of automating the student loan fund at Al-Baath University and the basic and subsidiary requirements it needs, and outlines the tasks that this system must provide to meet all needs.

## 1.2. Scope of Project

This software system will be an automated system for student credit bureau. This system would be designed to maximize productivity by providing tools to help automate file review and loan granting, which would have been done manually. By increasing employee efficiency, the system meets the needs of employees while still easy to understand and use.

More specifically, this system is designed to allow employees to manage and communicate with a group of student auditors to grant them loans. The program will be facilitated with simple interfaces.

## 1.3. Glossary

|  |  |
| --- | --- |
| **Term** | **Definition** |
| Student Credit Office | An office that grants university students access to loans to assist their studies |
| cashier | The employee responsible for paying the loans and receiving the money |
| manager | is a person who has the highest authority at work and has powers to accept or reject loans |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

## 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

Cashier

Employee

Manager

Student

Software homepage

System Manager

Automated student credit center automation system

Figure - System Environment

The Student Credit Center Automation System consists of four active actors and one collaborative system.

The employee, cashier access the main page of the software that allows limited services. The manager accesses the entire system directly.

Note: The student is not directly connected to the system he communicates to the system through the employees working in the office

## 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 Employee Use Case

Getting a loan

Employee

#### Loan resumption

Student

#### 

#### Use case: Getting a loan

#### Diagram:

Employee

Getting a loan

Student

**Brief Description**

The student arrives at the office and the employee records a loan request for him.

**Initial Step-By-Step Description**

#### 1. The employee enters the student's information into the system

#### 2. The employee checks the conditions for accepting the sponsor and attaches the files

#### 3. The employee verifies that the student obtained the approval of student affairs and attaches the files to the student's information

#### 4. The system organizes the student card

#### 5. The system gives a student a special serial number

#### Xref: Section 3.2.1, Getting a loan

#### Use case: Loan resumption

#### Diagram:

employee

Loan resumption

Student

**Brief Description**

The student arrives at the office and the employee registers a request to resume the student loan after it has been previously suspended from the manager

**Initial Step-By-Step Description**

#### 1. The employee enters the student number into the program

#### 2. The employee verifies the student's success.

#### 3. The employee activates the student card in the system

#### Xref: Section 3.2.2, Loan resumption

### 2.2.2 Cashier Use Case

Students

Cashier

Pay off a loan

#### Use case: Pay off the loan

**Diagram:**

Students

Cashier

Pay off the loan

**Brief Description**

The cashier deducts the amount that the student has paid

**Initial Step-By-Step Description**

1. The cashier chooses the button to search the student’s serial number to access the student’s database.
2. Determine the type of loan from the system that the student will pay (monthly / computer)
3. The system determines the amount to be paid with the addition of delay benefits
4. The cashier enters the amount the student pays into the system.
5. The system deducts the amount paid from the amount that the student must pay.
6. The system prints a receipt of the amount paid, its benefits and date.

#### Xref: Section 3.2.1, Pay off the loan

Use case: **Inquiry about the amount paid out**

**Diagram:**

Cashier

Inquiry about the amount paid out

**Brief Description**

The treasurer obtains information about a loan for a specific student

**Initial Step-By-Step Description**

1. The cashier chooses to enter the student number and search for him in the system.

2. The system searches for the student number database

3. The cashier chooses details about this loan.

4. The system shows lists of the sums withdrawn and paid for the student number.

**Xref:** Section 3.2.2, **Inquiry about the amount paid out**

### 2.2.3 Manager Use Case

**Use case:** Stop granting the monthly loan

#### Diagram:

Manager

Stop granting the monthly loan

**Brief Description**

The principal pauses cards for students who fail the system until they succeed

**Initial Step-By-Step Description**

1. At the beginning of each academic year, the director requests the system to review student databases and search for students who have not submitted a success document

2. The system gives links to student cards

3. The principal activates the suspension of student cards

**Xref:** Section 3.3.1, Stop granting the monthly loan

## 2.3 User Characteristics

All users (employee / manager /

Cashier) from using buttons and drop-down menus

And similar tools.

The main screen of the program will have a simple interface

It contains a button for a loan and a button to repay the loan

The manager is expected to be familiar with Windows and that

It is able to handle databases.

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

## 2.4 Non-Functional Requirements

It is assumed that there is a local network between the staff devices. There is a printer connected to the cashier's computer to print a link of a certain size and not color (white / black). The speed of the software depends on the capabilities of the computer.

The manager will run on the private computer that contains the Access database. Access is already installed on this computer, which is a Windows operating system.

3.0. Requirements Specification

## 3.1 External Interface Requirements

## The only link for an external system is the link to the university staff database to verify the identity of the employees working on the system. A condition has been imposed that the employee be affiliated with the Student Loan Office. The database domains of interest to log in to the system are the employee's name and password

## 3.2 Functional Requirements

### 3.2.1 get a loan

|  |  |
| --- | --- |
| **Use Case Name** | get a loan |
| **XRef** | Section 2.2.1, get a loan  SDD, Section 7.3 |
| **Trigger** | The employee chooses to add a new student to the database. |
| **Precondition** | The employee has arrived at the main screen |
| **Basic Path** | 1. 1. The employee enters the student's information into the system 2. 2. The employee checks the conditions for accepting the sponsor and attaching the files 3. 3. The employee verifies that the student obtained the approval of student affairs and attached files to the student's information 4. 4. The system organizes the student card 5. 5. The system gives the student a special serial number |
| **Alternative Paths** |  |
| **Postcondition** | The Author has been added to the database. |
| **Exception Paths** | The employee may abandon the process at any time. |
| **Other** | include student information, name, university number, academic specialization and personal phone number…  The system cancels work if the system is discovered by the presence of the student’s name and university number Mojo in the database <<because he is entitled to register once on the loan>> |

### 3.2.2 Loan resumption

|  |  |
| --- | --- |
| **Use Case Name** | Loan resumption |
| **XRef** | Section 2.2.1, Loan resumption  SDD, Section 7.4 |
| **Trigger** | The employee chooses to resume the student's loan and activate his card |
| **Precondition** | The employee has arrived at the main screen |
| **Basic Path** | 1. 1. The employee enters the student number into the program 2. 2. The employee verifies the student's success. 3. 3. The employee activates the student card in the system |
| **Alternative Paths** | in step 1, if the entered student number is not in the rule, the employee is directed to re-enter the number correctly correctly. |
| **Postcondition** |  |
| **Exception Paths** | the employee may abandon the process at any time. |
| **Other** |  |

### 3.2.3 Pay off the loan

|  |  |
| --- | --- |
| **Use Case Name** | Pay off the loan |
| **XRef** | Sec 2.2.2 Pay off the loan  SDD, Section 7.5 |
| **Trigger** | The cashier chooses to pay off a loan to a student |
| **Precondition** | The cashier has arrived at the main screen |
| **Basic Path** | 1. 1. The cashier chooses the button to search for the student's serial number to access the student's database. 2. 2. Determine the type of loan from the system that the student will pay (monthly / computer) 3. 3. The system determines the amount to be paid, with the addition of delay benefits 4. 4. The cashier enters the amount the student pays into the system. 5. 5. The system deducts the amount paid from the amount that the student must pay. 6. 6. The system prints a receipt of the amount paid, its benefits and date. |
| **Alternative Paths** | in step 6, if the amount is paid in full, prints the clearance receipt |
| **Postcondition** | The database has been updated. |
| **Exception Paths** | If the student is more than 6 months late and has not paid a premium, the system shows a notification of the delay and his phone number to communicate with him. |
| **Other** |  |

### 3.2.4 Inquiry about the amount paid out of a loan

|  |  |
| --- | --- |
| **Use Case Name** | Inquiry about the amount paid out of a loan |
| **XRef** | Section 2.2.2, Inquiry about the amount paid out of a loan  SDD, Section 7.6 |
| **Trigger** | The cashier inquires about a student loan that is in the database |
| **Precondition** | The cashier has arrived at the main screen |
| **Basic Path** | 1. The cashier chooses to enter the student number and search for it in the system.  2. The system searches for the student number database  3. The cashier chooses details about this loan.  4. The system displays lists of amounts withdrawn and paid for the student number. |
| **Alternative Paths** | in step 3, if the cashier does not choose details  4. This loan will display only the total amount remaining |
| **Postcondition** |  |
| **Exception Paths** |  |
| **Other** |  |

### 3.2.5 Stop granting the monthly loan

|  |  |
| --- | --- |
| **Use Case Name** | Stop granting the monthly loan |
| **XRef** | Section 2.2.3, Stop granting the monthly loan  SDD, Section 7.7 |
| **Trigger** | chooses to pause or remove from a database. |
| **Precondition** | The manager has arrived at the manager’s main screen. |
| **Basic Path** | 1. 1. At the beginning of each academic year, the director requests the system to review student databases and search for students who have not submitted a success document 2. 2. The system gives links to student cards 3. 3. The principal activates the suspension of student cards |
| **Alternative Paths** | None |
| **Postcondition** | The requestor may be removed from the database. |
| **Exception Paths** | The manager may abandon the process at any time. |
| **Other** |  |

## Detailed Non-Functional Requirements

In systems engineering and requirements engineering, a non-functional requirement

(NFR) is a requirement that specifies criteria that can be used to judge the operation

of a system, rather than specific behaviors. They are contrasted with functional

requirements that define specific behavior or functions

It is assumed that there is a local network between the staff devices. There is a printer connected to the cashier's computer to print a link of a certain size and not color (white / black). The speed of the software depends on the capabilities of the computer.

The manager will run on the private computer that contains the Access database. Access

is already installed on this computer, which is a Windows operating system

* + 1. scalability:

Scalability is a non-functional property of a system that describes the ability to

appropriately handle increasing (and decreasing) workloads. According to Coulouris et

al. [[Dol05](http://berb.github.io/diploma-thesis/original/0_bibliography.html#Dollimore2005)], "a system is described as scalable, if it will remain effective when there is a

significant increase in the number of resources and the number of users". Sometimes,

scalability is a requirement that necessitates the usage of a distributed system in the first

place. Also, scalability is not to be confused with raw speed or performance. Scalability

competes with and complements other non-functional requirements such as availability,

reliability and performance.

* + 1. availability :

is a requirement that aims at the indentured availability of a system during a

certain period. It is often denoted as percentiles of uptime, restricting the

maximum time to be unavailable

* + 1. Reliability :

is a closely related requirement that describes the time span of operational

behavior, often measured as meantime between failures. Scalability,

anticipating increasing load of a system, challenges both requirements. A

potential overload of the systems due to limited scalability harms availability

and reliability

* + 1. Capacity :

Capacity requirements deal with the amount of information or services that can be

handled by the component or system. These are important since they establish the way

that the system can be used. If the capacity needs are not clearly defined, developers

might underestimate what is needed and the users will find the system unusable.