Software requirement specification document for project school managment system

Ahmed hossny,karim ali,ahmed abdulla,omar yasser
Supervised by: Dr. ..ayman ezzat....

June 9, 2021

1 Introduction

1.1 Purpose of this document

This is the school system management we are building this project to maintain a paperless management to the school data and saving it in a database to have an easy access to all the school data.

1.2 Scope of this document

controlling and managing school data in an ordered and uniformed way and having a paperless school documents with easy access

1.3 Overview

we are going to allow our user do manipulate data by adding, removing and modify data to maintain clear specified data also calculating some information about these data and display it to the user

1.4 Business Context

our customer needs an access to database of the students, teachers, staffs and the school rooms while knowing the student paid fees and his remained installments and bus fees

2 General Description

2.1 Product Functions

login() add(student) delete(student) update(student) view(student) count(student), add(teacher) delete(teacher) update(teacher) view(teacher) count(teacher) add(staff) delete(staff) update(staff) view(staff) count(staff) add(rooms) delete(rooms) update(rooms) view(rooms) count(rooms) previous() exit(teacher,student,admin,staff,rooms)

2.2 Similar System Information

A system proposed by Yahya, M. Wijoyo, H. (2020). Developing School Information Program: Integrated Management System based on Character Value at SMP Negeri 9 Tapung. International Journal of Asian Education,This research aims to develop a student data management system for 9 Tapung Junior High School (SMP) Negeri 9 Tapung. The system has not yet recorded the use of school computers or networks using programming languages based on HTML, JavaScript, CSS and other text, and PHP. The design of the SMP Negeri 9 Tapung student information system uses a reading and writing method in its manufacturing process, in which each stage can be repeated until the desired result is obtained. The manufacturing stage of the system is research, analysis, design and implementation. To make this application, the author used PHP and MySQL. The results of this development research created a student information system application that can display school data, student data, what form of graphs and their development, including development data about honesty and discipline. The system also displays observed student interests and talents, extracurricular activities, student irregularities and grades. This application still needs to be developed, improved, added features and tips, as well as criticism from all relevant parties, to create a better structured information system to improve the quality and features of SMP Negeri 9 Tapung.also this project displayed alto of data development and statistics about data inserted which we will not do it

2.3 User Characteristics

the are two main user that will login to the system whom are the admin and staff each has speechified role which admin will be able to manipulate all data except preparing rooms and studet distribution which is done only by the staff

2.4 User Problem Statement

This section describes the essential problem(s) currently confronted by the user community.

2.5 User Objectives

users needed more interactions than we offered as they needed full access on all school departments and employees also alot of statistics and development about the data inserted

Table 1: Functional Requirement XYZ

Function Name	login.
Description	allow user to login into the sytsem.
Critically	critcal.
Technical issues	any NULL values.
Cost and schedule	3hr.
Risks	-.
Dependencies with other requirements	-.
input	id, password
output	boolean
Pre-Condition	welcome page.
Post-Condition	user page

Table 2: Functional Requirement XYZ

Function Name	add.
Description	add and insert values in the database.
Critically	high.
Technical issues	any NULL values or different datatype.
Cost and schedule	2hr.
Risks	-.
Dependencies with other requirements	-.
input	id ,name,birthdate,religion,phonenumber,nationalitiy,address.
output	
Pre-Condition	user page.
Post-Condition	perivous page

2.6 General Constraints

we could face alot of requirments without giving proper budget also some resources may be weak like our PC and laptops adding alot of limitations also may be one our members got infected by corona virus.

3 Functional Requirements

This section lists the functional requirements in ranked order. Functional requirements describes the possible effects of a software system, in other words, what the system must accomplish. Other kinds of requirements (such as interface requirements, performance requirements, or reliability requirements) describe how the system accomplishes its functional requirements see table 8. Each functional requirement should be specified in a format similar to the following:

4 Interface Requirements

This section describes how the software interfaces with other software products or users for input or output. Examples of such interfaces include library routines, token streams, shared memory, data streams, and so forth.

Table 3: Functional Requirement XYZ

Function Name	delete.
Description	delete and remove values in the database.
Critically	high.
Technical issues	-.
Cost and schedule	1hr.
Risks	- deleting wrong data and losing it.
Dependencies with other requirements	-.
input	-.
output	database removed(raw)
Pre-Condition	user page.
Post-Condition	perivous page

Table 4: Functional Requirement XYZ

Function Name	update.
Description	changes and modify values in the database.
Critically	medium.
Technical issues	any NULL values or different datatype.
Cost and schedule	2hr.
Risks	-.
Dependencies with other requirements	-.
input	id ,name,birthdate,religion,phonenumber,nationalitiy,address.
output	
Pre-Condition	user page.
Post-Condition	perivous page

Table 5: Functional Requirement XYZ

Function Name	view.
Description	view all the data in the database.
Critically	medium.
Technical issues	any NULL values or different datatype.
Cost and schedule	1hr.
Risks	-.
Dependencies with other requirements	-.
input	
output	database
Pre-Condition	staff or room or teacher or student.
Post-Condition	previous page

Table 6: Functional Requirement XYZ

Function Name	count.
Description	conut and display values in the database.
Critically	low.
Technical issues	any NULL values or different datatype.
Cost and schedule	50min.
Risks	-.
Dependencies with other requirements	-.
input	id
output	max id
Pre-Condition	student,teacher,room,staff.
Post-Condition	previous page

Table 7: Functional Requirement XYZ

Function Name	exit.
Description	closes the system.
Critically	low.
Technical issues	-.
Cost and schedule	2hr.
Risks	might close after few seconds.
Dependencies with other requirements	-.
input	-
output	
Pre-Condition	-.
Post-Condition	-

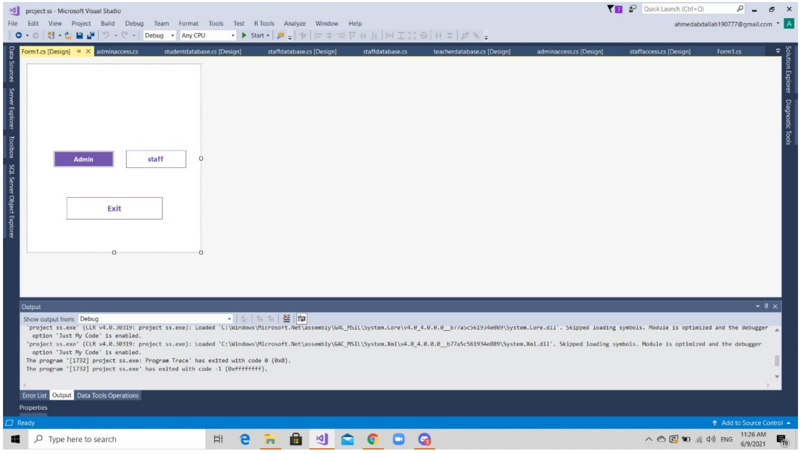
Table 8: Functional Requirement XYZ

Function Name	prevoius.
Description	return to the previous page.
Critically	low.
Technical issues	-.
Cost and schedule	1hr.
Risks	-.
Dependencies with other requirements	-.
input	-
output	-
Pre-Condition	userpage.
Post-Condition	previous page

4.1 User Interfaces

Use some software for primitive plan of your project. Describes how this product interfaces with the user.

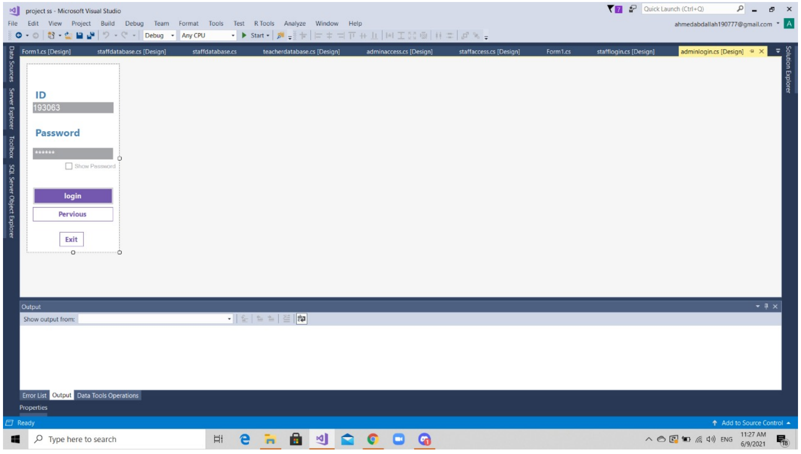
4.1.1 GUI



```
'project ss.exe' (CLR v4.0.30319: project ss.exe): Loaded 'C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System.Core\v4.0_4.0.0.0__b77a5c561934e089\System.Core.dll'. Skipped loading symbols. Module is optimized and the debugger
  option 'Just My Code' is enabled.
'project ss.exe' (CLR v4.0.30319: project ss.exe): Loaded 'C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System.Xml\v4.0_4.0.0.0__b77a5c561934e089\System.Xml.dll'. Skipped loading symbols. Module is optimized and the debugger
  option 'Just My Code' is enabled.
The program '[1732] project ss.exe: Program Trace' has exited with code 0 (0x0).
The program '[1732] project ss.exe' has exited with code -1 (0xffffffff).
```

Figure 1: Inheritance Relations

. . . . . .



ID: 193063

Password: ******

□ Show Password

Figure 2: Inheritance Relations

4.1.2 CLI

Describes the command-line interface if present. For each command, a description of all arguments and example values and invocations should be provided.

4.1.3 API

Describes the application programming interface, if present. For each public interface function, the name, arguments, return values, examples of invocation, and interactions with other functions should be provided.

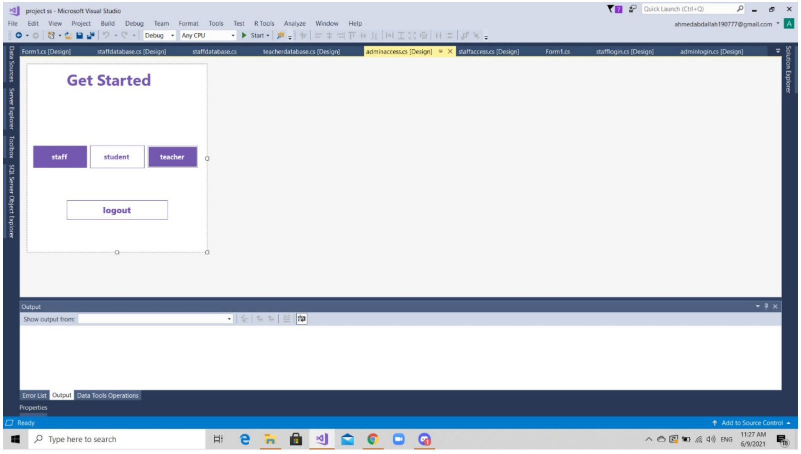
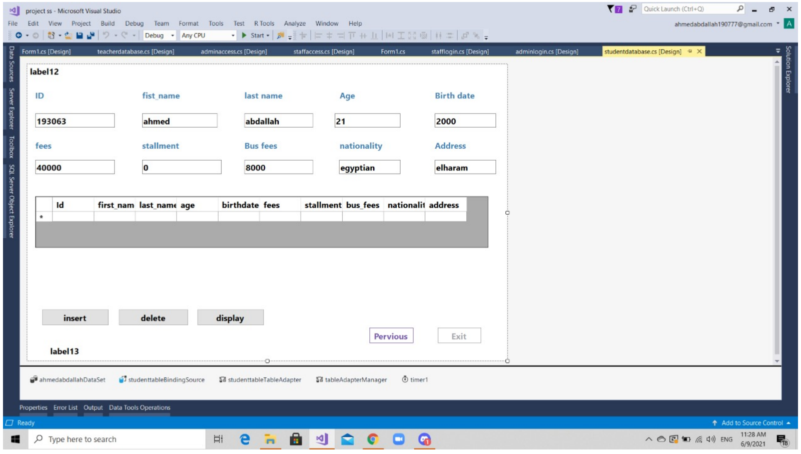


Figure 3: Inheritance Relations



ID: 193063
fist_name: ahmed
last name: abdallah
Age: 21
Birth date: 2000
fees: 40000
stallment: 0
Bus fees: 8000
nationality: egyptian
Address: elharam

Id	first_nam	last_name	age	birthdate	fees	stallment	bus_fees	nationalit	address

Figure 4: Inheritance Relations

4.1.4 Diagnostics or ROM

Describes how to obtain debugging information or other diagnostic data.

4.2 Hardware Interfaces

we are gonn need network between school pc and switches and good ram storage and internet access

4.3 Communications Interfaces

Describes network interfaces.

4.4 Software Interfaces

need visual studio program and its languages

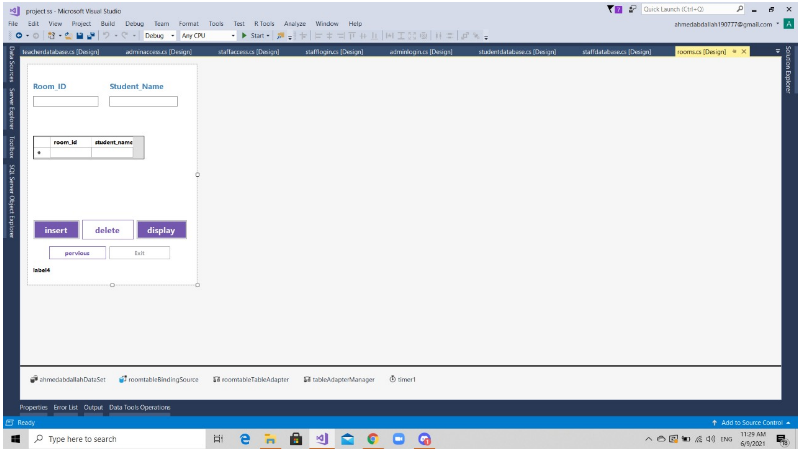
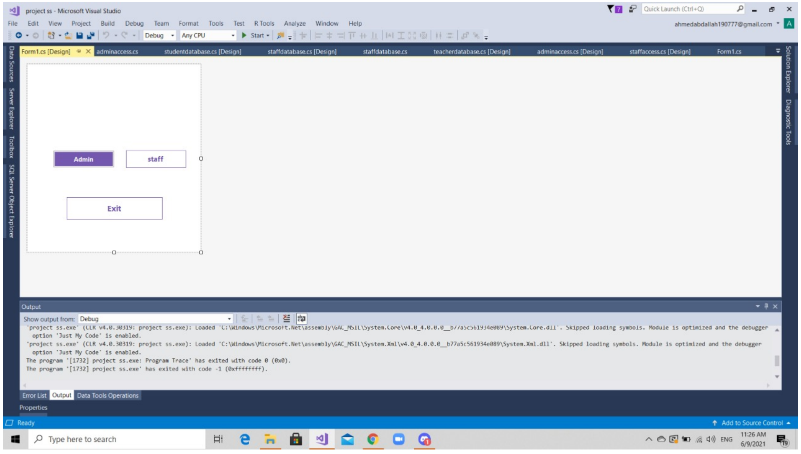


Figure 5: Inheritance Relations



```
'project ss.exe' (CLR v4.0.30319: project ss.exe): Loaded 'C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System.Core\v4.0_4.0.0.0__b77a5c561934e089\System.Core.dll'. Skipped loading symbols. Module is optimized and the debugger
  option 'Just My Code' is enabled.
'project ss.exe' (CLR v4.0.30319: project ss.exe): Loaded 'C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System.Xml\v4.0_4.0.0.0__b77a5c561934e089\System.Xml.dll'. Skipped loading symbols. Module is optimized and the debugger
  option 'Just My Code' is enabled.
The program '[1732] project ss.exe: Program Trace' has exited with code 0 (0x0).
The program '[1732] project ss.exe' has exited with code -1 (0xffffffff).
```

Figure 6: Inheritance Relations

5 Performance Requirements

need high cpu and ram and windows 7 or higher.

6 Design Constraints

buttons limitation or text box constraunt in single form.

6.1 Standards Compliance

need photo of the school and high quiality interface

6.2 Hardware Limitations

6.3 others as appropriate

7 Other non-functional attributes

Specifies any other particular non-functional attributes required by the system. Examples are provided below.

7.1 Security

for sequirity purposes we did a login page and database for it.

7.2 Binary Compatibility

7.3 Reliability

7.4 Maintainability

easy transformation between forms and a backup plan every 4 month.

7.5 Portability

connected network for school pc

7.6 Extensibility

could be used by another computers connected to the same network.

7.7 Re-usability

7.8 Application Affinity/Compatibility

windows 7 or higher.

7.9 Resource Utilization

7.10 Serviceability

data managment

7.11 others as appropriate

8 Preliminary Object-Oriented Domain Analysis

. .

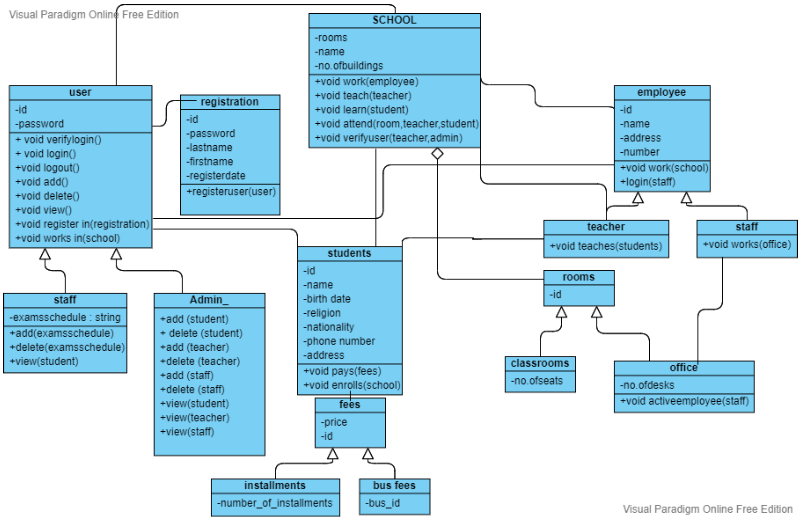


Figure 7: Inheritance Relations

8.1 Inheritance Relationships

This section should contain a set of graphs that illustrate the primary inheritance hierarchy (is-kind-of) for the system. For example:

8.2 Class descriptions

This section presents a more detailed description of each class identified during the OO Domain Analysis. For more details on the process giving rise to these descriptions, see Lecture 5.3: OO Domain Analysis and/or texts on object-oriented software development. Each class description should conform to the following structure:

8.2.1 Class name

Abstract or Concrete: Indicates whether this class is abstract or concrete.

8.2.2 List of Superclasses:

Names all immediate superclasses.

8.2.3 List of Subclasses:

Names all immediate subclasses.

8.2.4 Purpose:

States the basic purpose of the class.

8.2.5 Collaborations:

Names each class with which this class must interact in order to accomplish its purpose, and how.

8.2.6 Attributes:

Lists each attribute (state variable) associated with each instance of this class, and indicates examples of possible values (or a range).

8.2.7 Operations

: Lists each operation that can be invoked upon instances of this class. For each operation, the arguments (and their type), the return value (and its type), and any side effects of the operation should be specified.

8.2.8 Constraints:

Lists any restrictions upon the general state or behavior of instances of this class.

9 Operational Scenarios

This section should describe a set of scenarios that illustrate, from the user's perspective, what will be experienced when utilizing the system under various situations. In the article Inquiry-Based Requirements Analysis (IEEE Software, March 1994), scenarios are defined as follows: In the broad sense, a scenario is simply a proposed specific use of the system. More specifically, a scenario is a description of one or more end-to-end transactions involving the required system and its environment. Scenarios can be documented in different ways, depending up on the level of detail needed. The simplest form is a use case, which consists merely of a short description with a number attached. More detailed forms are called scripts. These are usually represented as tables or diagrams and involved identifying an action and the agent (doer) of the action. For this reason, a script can also be called an action table. Although scenarios are useful in acquiring and validating requirements, they are not themselves requirements, because the describe the system's behavior only in specific situations; a specification, on the other hand, describes what the system should do in general.

10 Preliminary Schedule Adjusted

This section provides an initial version of the project plan, including the major tasks to be accomplished, their interdependence's, and their tentative start/stop dates. The plan also includes information on hardware, software, and resource requirements. The project plan should be accompanied by one or more PERT or GANTT charts.

11 Preliminary Budget Adjusted

This section provides an initial budget for the project, itemized by cost factor.

12 Appendices

Specifies other useful information for understanding the requirements. All SRS documents should include at least the following two appendices:

12.1 Definitions, Acronyms, Abbreviations

Provides definitions of unfamiliar definitions, terms, and acronyms.

12.2 Collected material

13 References

References