Management Information System

Software Requirements Specification

Version 1.1

2019-10-31

Techno Rhino Team



ALBOROM WARD - 201732120143

IBRAHIM MOHAMED IBRAHIM ISMAIL – 201732120159

MOHAMED JIFRY HAZZALY MOHAMED – 201732120165

ABDALMUHAYMEN M.A.KHALIL – 201730210234

TARMOM MOHAMMED SALEH ABDULLAH – 201732120166

SAEED SALEH ALMASOODI – 201732120146

Thanks to teacher: Jian bing

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Description** | **Author** | **Comments** |
| 10/22 | Initial writing | WARD | First steps of the document |
| 10/31 | First revision | IBRAHIM | Completing the document |
| 11/7 | Second revision | Team | Updated some requirements |
| 11/21 | Last revision | Team | Last update for the document |

**Table of Contents**

Revision History ii

1. Introduction 1

1.1 Purpose 1

1.2 Scope 1

1.3 Definitions, Acronyms, and Abbreviations 1

Administrator Side: is the GUI application that meant to be used by the school administrator to access the school system and manipulate the students and instructors data. 1

1.4 References 2

1.5 Overview 2

2. General Description 2

2.1 Product Perspective 2

2.2 User Characteristics 3

2.3 General Constraints 3

2.4 Assumptions and Dependencies 3

3. Specific Requirements 3

3A Administrator Side Requirements: 3

3W Website Side Requirements. 4

3.1 External Interface Requirements 5

3.1.1 User Interfaces 5

3.1.2 Hardware Interfaces 5

3.1.3 Software Interfaces 5

3.1.4 Communications Interfaces 5

3.2 Functional Requirements 6

3.2.1 <Administrator registers a student> 6

3.2.2 <Administrator issue new curriculum> 6

3.2.3 <Student selects a course> 6

3.2.4 <Instructor selects a course to teach> 7

3.2.5 <student drops a course> 7

3.3 Use Cases 8

3.3.1 Use Case #1 8

3.3.2 Use Case #2 8

3.3.3 Use Case #3 9

3.3.4 Use Case #4 9

3.3.5 Use Case #5 9

3.3.6 Use Case #6 10

3.3.7 Use Case #7 10

3.3.8 Use Case #8 10

3.3.9 Use Case #9 11

3.4 Classes / Objects 11

3.4.1 <Course> 11

3.4.2 <Student #2> 11

3.4.3 <Instructor #3> 12

3.4.4 <Notification #4> 12

3.4.5 <DB #5> 12

3.4.6 <Controller #6> 12

3.5 Non-Functional Requirements 13

3.5.1 Performance 13

3.5.2 Reliability 13

3.5.3 Availability 13

3.5.4 Security 13

3.5.5 Maintainability 13

3.7 Design Constraints 13

3.8 Database Visualization 14

4. Change Management Process 16

A. Appendices 16

A.1 Appendix 1 16

A.2 Appendix 2 19

# 1. Introduction

Every university or school needs an information management system that can manage students information and courses, and the different and special thing about our project is that it contains an easy beautiful user interface, a good and stable database that order and manage all the data including students information and courses information, the website can edit information containing add info, edit info, delete info, show grades, show study reports, print out info, add courses, drop courses, open courses, close courses. It contains many functions and works on desktop platforms, we have created this website to make students and teachers information manageable and easy access.

## 1.1 Purpose

*This SRS is to introduce our virgin idea and the first project of our team which is a website to manage the information of students* by creating this website to we make students and teachers information manageable and easy access

## 1.2 Scope

*(1) allow the ease of access to the school resources.*

*(2) help administrators to manage students information.*

*(3) allow instructors to access their own information, courses information, students information, and see the available courses, as well as the number of students enrolled in each course.*

*(4) limit the need of the physical, face-to-face interactions between the students and instructors.*

## 1.3 Definitions, Acronyms, and Abbreviations

## Administrator Side: is the GUI application that meant to be used by the school administrator to access the school system and manipulate the students and instructors data.

**Website Side:** is a school web application that will be available for school students and instructors to access and review their data and school related options.

**ZJNU:** Zhejiang Normal University.

## 1.4 References

*(1) ZJNU students platform: http://10.1.70.168/*

*(2) ZNJU academic affairs platform:*<http://jwc.zjnu.edu.cn/>

*(3) Software Design course, school MIS GUI project.*

## 1.5 Overview

*This SRS contains:*

* *All the requirements gathered by Techno Rhino team, for both the Administrator-side, and Website-side.*
* *A description of the Database structure used in this system,illustrated via diagrams.*
* *The system implementation,relations between classes explained via class diagrams.*
* *List of the most important functions in the system.*
* *Use cases diagrams, and detailed explanation of each use case.*
* *The process, in which is used to develop this system.*
* *Test cases, which is carried out by the Test-team of Techno Rhino.*

# 2. General Description

*School MIS ( management information system).*

*Is a system to manage students info in a given school, the system is divided into two parts, administrator side, admin can add students to the system, create curriculum, create courses, and send notes to the website, where the students can access their info, as well as select courses, drop courses, and see the latest updates from the school.*

## 2.1 Product Perspective

*Compared with the school platform.this system is developed by inexperienced team of developers, so some functions might not be as expected.*

*But, the main idea of this system was to mimic and implement a solution of the difficulties that was facing,and still facing the students in the official ZJNU student platform,which is the language.for a newcomers to the school, they are not expected to read chines, which is the one and only option you get to choose using the website. Hence the idea started to grow. since we are students of Software Engineering, this idea is quite familiar for each software student in ZJNU who has used the website before.*

## 2.2 User Characteristics

***Administrator:*** *the person who is responsible of the manipulation of students,courses,instructors,and notifications.*

***Instructors:*** *teachers,lecturers,doctors,professors who are going to use the system.*

***Students:*** *graduate,undergraduate students, since it is a university system.*

## 2.3 General Constraints

*Since it is a course project, there are a lot of constraints:*

* *There is no budget.*
* *The system must handle server outage or crash due to a heavy load of users requests.*
* *Each student must be a valid, enrolled, student of ZJNU.*
* *Administrators can be only a member of the study affairs.*
* *Instructors must be registered as a faculty member of ZJNU.*

## 2.4 Assumptions and Dependencies

All contents of this SRS are under the following assumptions:

* The school has only one campus.
* The school located in China.
* The system serves students,and instructors.
* No course assignments involved in the system.
* The number of users(students,instructors) is limited.

Any change of these assumptions and dependencies, may affect the system requirements.however, in case of any changes here the SRS must be reviewed and corrected again.

# 3. Specific Requirements

## 3A Administrator Side Requirements:

***REQA1***:

*The administrator can login into the system after he/she gets granted with the login privilege in the System’s database.*

***REQA2****:*

*After the administrator login, he/she redirected to the GUI’s main page where he/she can see the students information.*

***REQA3:***

*Administrator can access three main functions, Students management, Courses management, and notifications.*

***REQA4:***

*In Students Management, administrator can add a new student to the system, Edit existing student information, delete student record, and query search for a specific student by the student’s ID number*.

***REQA5:***

*In Courses Management, administrator can issue new semester curriculum, add list of compulsory courses, and list of elective courses that the student can select from.*

***REQA6:***

*In Notifications, administrator can issue some notifications for the students to be displayed in the website, such as reminders, advises, and the latest updates about the school system*.

***REQA7:***

*In the GUI interface the administrator information is displayed such as the name, role, and status.*

***REQA8:***

*Administrator can logout from the system by clicking on the logout button on the GUI.*

## 3W Website Side Requirements.

***REQW9:***

*The student can login into the website by his student number as username, and last 6 digits from his/shes passport as password.*

***REQW10:***

*After the student login, he/she can change the password number if he/she would like to*.

***REQW11:***

*The homepage has several sections,notifications,select courses,and courses, as well as the student can see his information displayed in the homepage, such as name, college, and major.*

***REQW12:***

*In the notifications page, the student can see the latest notifications released by the school.*

***REQW13:***

*In the select courses page, the student can select number of elective courses issued by the school for the given semester.*

***REQW14:***

*In the courses page, the student can check the compulsory courses he/she has to attend for the given semester, elective courses he/she selected, and he/she can also drop courses.*

***REQW15:***

*Finally the student can logout the system by clicking the logout button.*

## 3.1 External Interface Requirements

### 3.1.1 User Interfaces

***REQA2****:*

*After the administrator login, he/she redirected to the GUI’s main page where he/she can see the students information.*

***REQA7:***

*In the GUI interface the administrator information is displayed such as the name, role, and status.*

***REQA8:***

*Administrator can logout from the system by clicking on the logout button on the GUI.*

### 3.1.2 Hardware Interfaces

*None*

### 3.1.3 Software Interfaces

***REQA3:***

*Administrator can access three main functions, Students management, Courses management, and notifications*.

***REQA4:***

*In Students Management, administrator can add a new student to the system, Edit existing student information, delete student record, and query search for a specific student by the student’s ID number.*

***REQA5:***

*In Courses Management, administrator can issue new semester curriculum, add list of compulsory courses, and list of elective courses that the student can select from.*

### 3.1.4 Communications Interfaces

*Communication between the administrator GUI and the Website platform, is through the database. Such as administrator issue a course from his side,the course stored in the database,the student or instructor can access the course from the website platform through the database linking both sides.*

## 3.2 Functional Requirements

### 3.2.1 <Administrator registers a student>

3.2.1.1 Introduction

*In this feature the administrator can register a new student to the school system*.

3.2.1.2 Inputs

*Student information*.

3.2.1.3 Processing

*All the given above data stored in the database.*

3.2.1.4 Outputs

*Administrator can fetch student information.*

*Student now can access the website.*

3.2.1.5 Error Handling

*Missing values shall be considered, what if the admin forgot to enter one of the required values? What he is going to see?*

### 3.2.2 <Administrator issue new curriculum>

3.2.2.1 Introduction

*Administrator can issue new curriculum for each semester(or year)*.

3.2.2.2 Inputs

*Curriculum information.*

*With curriculum information, a list of courses that can be chosen each semester.*

3.2.2.3 Processing

*Curriculum information, and courses are stored in the database.*

3.2.2.4 Outputs

*Administrator can manipulate courses.*

*Students can select,drop courses.*

*Instructors can be assigned courses to teach.*

3.2.2.5 Error Handling

*Duplicate courses values must be taking into account, what can we do to prevent this?*

### 3.2.3 <Student selects a course>

3.2.3.1 Introduction

*The student can select a course, either compulsory, or elective*.

3.2.3.2 Inputs

*Course number.*

3.2.3.3 Processing

*Course information is fetched from the database.*

3.2.3.4 Outputs

*Student can see the course information.*

3.2.3.5 Error Handling

*What if the course number is wrong? What should the student see?*

*What if there is a crash between courses what should the system do?*

### 3.2.4 <Instructor selects a course to teach>

3.2.4.1 Introduction

*Instructor selects a course to teach perhaps!*

3.2.4.2 Inputs

*Course id.*

3.2.4.3 Processing

*Course information fetched from the database.*

3.2.4.4 Outputs

*Instructor can see the course information, with number of students enrolled in the course.*

3.2.4.5 Error Handling

*What should happen if the instructor selects a course then he decides to drop it? What should happen if the course he selects gets removed from the system by administrator?*

### 3.2.5 <student drops a course>

3.2.5.1 Introduction

*Students all of a sudden decides to drop a course!*

3.2.5.2 Inputs

*From his/her course list, select a course to be dropped.*

3.2.5.3 Processing

*The course gets removed from his list of courses.*

3.2.5.4 Outputs

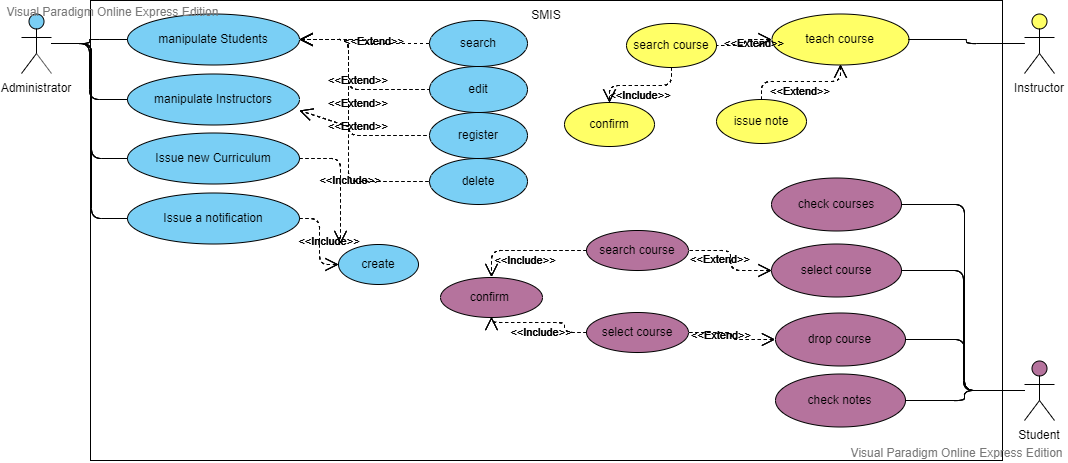
*The course is no longer in the student’s list*

3.2.5.5 Error Handling

*what if the website crashes in the middle of the dropping process? Is the course going to be removed from his/her list? Who so ? if the student click drop and close the page immediately? Is the course going to be dropped anyway?*

*How is the system is going to assure for the student that the course is dropped?*

## 3.3 Use Cases



*Figure 1: use cases diagram.*

### 3.3.1 Use Case #1

***NAME:*** *Manipulate Students*

***Actors:*** *Administrator, System.*

***Entry Condition:*** *Logged in as an administrator in the admission application.*

***Exit Condition:*** *Results executed successfully.*

***Flow of Events:***

***Administrator:*** *logs in.*

***System:*** *verify the administrator.*

***Administrator:*** *Performs some of the available options, Insert, Delete, Edit, or Query students.*

***System:*** *successfully carry out the results accordingly.*

***Administrator:*** *see the results.*

***Special Requirements:***

*NONE*

### 3.3.2 Use Case #2

***NAME:*** *Issue new Courses*

***Actors:*** *Administrator, System.*

***Entry Condition:*** *Logged in as an administrator in the admission application.*

***Exit Condition:*** *Results executed successfully.*

***Flow of Events:***

***Administrator:*** *logs in.*

***System:*** *verify the administrator.*

***Administrator:*** *Clicks Insert.*

***System:*** *Pops up a insertion form.*

***Administrator:*** *Populate the insertion form with the course information, clicks insert.*

***System:*** *Inserts course into database, and populate the table with the new record.*

***Administrator:*** *gets confirmation message, and sees the course record in the table.*

***Special Requirements:***

None

### 3.3.3 Use Case #3

***NAME:*** *Manipulate Instructors*

***Actors:****Administrator, System.*

***Entry Condition:****Logged in as an administrator in the admission application.*

***Exit Condition:****Results successfully executed*

***Flow of Events:***

***Administrator:****Performs some of the available options, Insert, Delete, Edit, or Query instructors.*

***System:*** *carry out the results successfully.*

***Administrator:*** *sees the updated table with new changes.*

***Special Requirements:***

None

### 3.3.4 Use Case #4

***NAME:****Issue Notification*

***Actors:****Administrator, System*

***Entry Condition:****Logged in as an administrator in the admission application.*

***Exit Condition:****New notification has been issued.*

***Flow of Events:***

***Administrator:*** *Clicks create note*

***System:*** *Pops up a creation form.*

***Administrator:*** *Fills the form, then clicks insert.*

***System:*** *A new note is created and inserted to the database, as well as is shown on the table.*

***Special Requirements:***

None

### 3.3.5 Use Case #5

***NAME:*** *Check courses*

***Actors:*** *Student, Website*

***Entry Condition:*** *Logged in as student on the website.*

***Exit Condition:*** *Student gets a list of the courses he selected to attend.*

***Flow of Events:***

***Student:*** *on the home page, the student clicks on self-selected courses tab.*

***Website:*** *Retrieve a list of the courses the student is attending.*

***Special Requirements:***

None

### 3.3.6 Use Case #6

***NAME:*** *Drop Course.*

***Actors:****Student, Website.*

***Entry Condition:****Logged in as Student on the website.*

***Exit Condition:****student successfully dropped the course.*

***Flow of Events:***

***Student:*** *on the home page, the student clicks on self-selected courses tab.*

***Website:*** *Retrieve a list of the courses the student is attending, next to each course there is a remove button.*

***Student:*** *Clicks on the remove button parallel to the course he/she may want to remove.*

***Website:*** *Removes the course from the students list of courses.*

***Special Requirements:***

None

### 3.3.7 Use Case #7

***NAME:****Select Course*

***Actors:****Student, Website.*

***Entry Condition:*** *logged in as a student on the website.*

***Exit Condition:*** *course being added to student’s list of courses.*

***Flow of Events:***

***Student:*** *on the home page the student clicks on the choose courses tab.*

***Website:****redirect the student to a query page where he/she can search for a course by its id.*

***Student:****enters the course id, clicks search.*

***Website:****returns the course available. With a add button next to each course.*

***Student:****Student clicks on add.*

***Website:****redirect student the list of courses with the new course being added to his list.*

***Special Requirements:***

None

### 3.3.8 Use Case #8

***NAME:****Checks Notifications*

***Actors:****Student, Website.*

***Entry Condition:****Logged in as a student on the website.*

***Exit Condition:****scam through the list of notifications.*

***Flow of Events:***

***Student:*** *On the home page, student clicks on notifications storage tab.*

***Website:*** *redirects the student to the notifications page, shown a list of issued notes on the system.*

***Special Requirements:***

None

### 3.3.9 Use Case #9

***NAME:****Instructor checks courses and students*

***Actors:****Instructor, Website.*

***Entry Condition:****logged in as an instructor on the website.*

***Exit Condition:****sees a list of courses he/she teaches and for each course the list of students attending the course.*

***Flow of Events:***

***Instructor:*** *Clicks on teaching courses.*

***Website:*** *Redirect the instructor to a page with a list of courses he/she teaches, as well as an option for each course to check the students enrolled in the course.*

***Special Requirements:***

None

## 3.4 Classes / Objects

**Administrator side objects.**

### 3.4.1 <Course>

3.4.1.1 Attributes

id,name,teacher,semester,day,period,credits,and room.

3.4.1.2 Functions

Update record

Delete record

Insert record

Get Course information

### 3.4.2 <Student #2>

3.4.2.1 Attributes

iD,name,dOB,gender,passport,nationality,college,major,phone,address,password,email.

3.4.2.2 Functions

Update record

Delete record

Insert record

Get Student information

### 3.4.3 <Instructor #3>

3.4.3.1 Attributes

id,name,date\_of\_birth,gender,college,phone,address,password,degree,email.

3.4.3.2 Functions

Update record

Delete record

Insert record

Get Instructor information

### 3.4.4 <Notification #4>

3.4.4.1 Attributes

id,title,content,issue\_date,author.

3.4.4.2 Functions

Update record

Delete record

Insert record

Get Notification information

**Website Side Objects.**

### 3.4.5 <DB #5>

3.4.5.1 Attributes

Driver,url,root,password.

3.4.5.2 Functions

Check user

Check password

Change password

Check teacher

Add course

Remove course

### 3.4.6 <Controller #6>

3.4.5.1 Attributes

None

3.4.5.2 Functions

DoGet

DoPost

And other JSP pages.

## 3.5 Non-Functional Requirements

### 3.5.1 Performance

*80% of the requests shall be processed in less than 5 seconds.*

*Server downtime must not exceed 1 hour a day, in a non-working time like night time, and less than 30 minutes in the working time.*

### 3.5.2 Reliability

*System may implement some backup plans to prevent the loss of information in case of any crash happens to the database, as well as more than one host in cases of server down.*

### 3.5.3 Availability

*System must be available 24/7 during the whole study year.*

*And must be accessible by any method such as, PCs, Smart Phones,and embedded systems.*

### 3.5.4 Security

*All data in the system must be protected, and the access to the database must be restricted to a number of people,such as the administrators,and maintainers.*

*The privacy policy should be taking into account, system users shall be guaranteed that their information will never be available to the public,or any officials with out their knowledge and approval.*

### 3.5.5 Maintainability

*Maintainers should ask for access privilege to the system in case of any maintenance needed.*

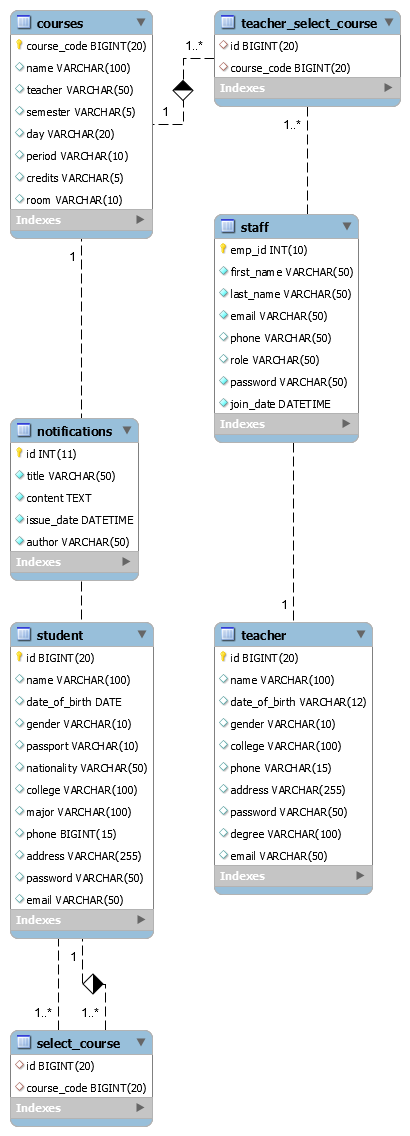
*System must be updated occasionally to keep the reliability and integration of the service.*

## 3.7 Design Constraints

*The system must complies to the school and people’s republic of China policies.*

*And assure that the system doesn’t brick any of the rules and fully legal.*

## 3.8 Database Visualization



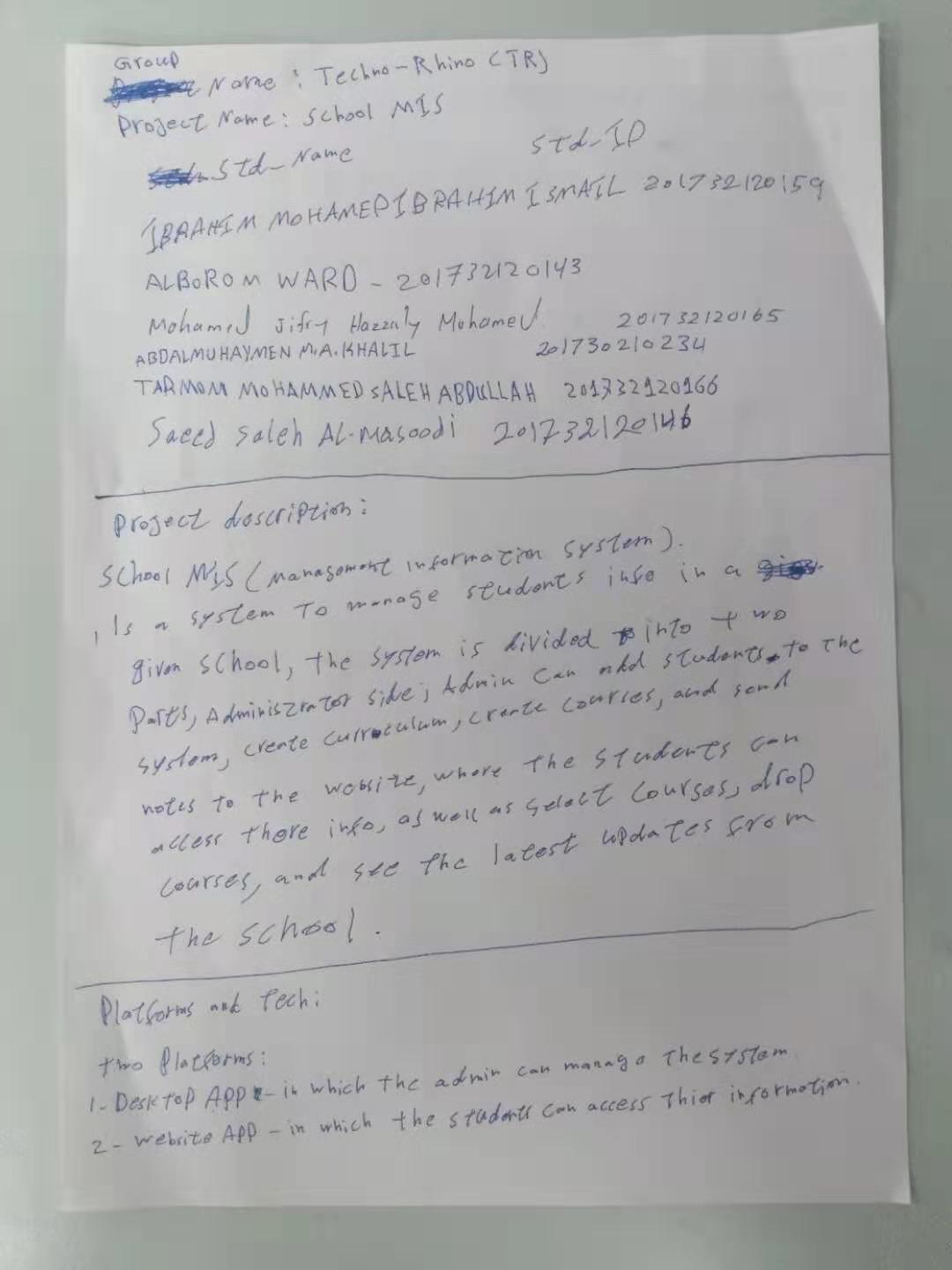
## Change Management Process

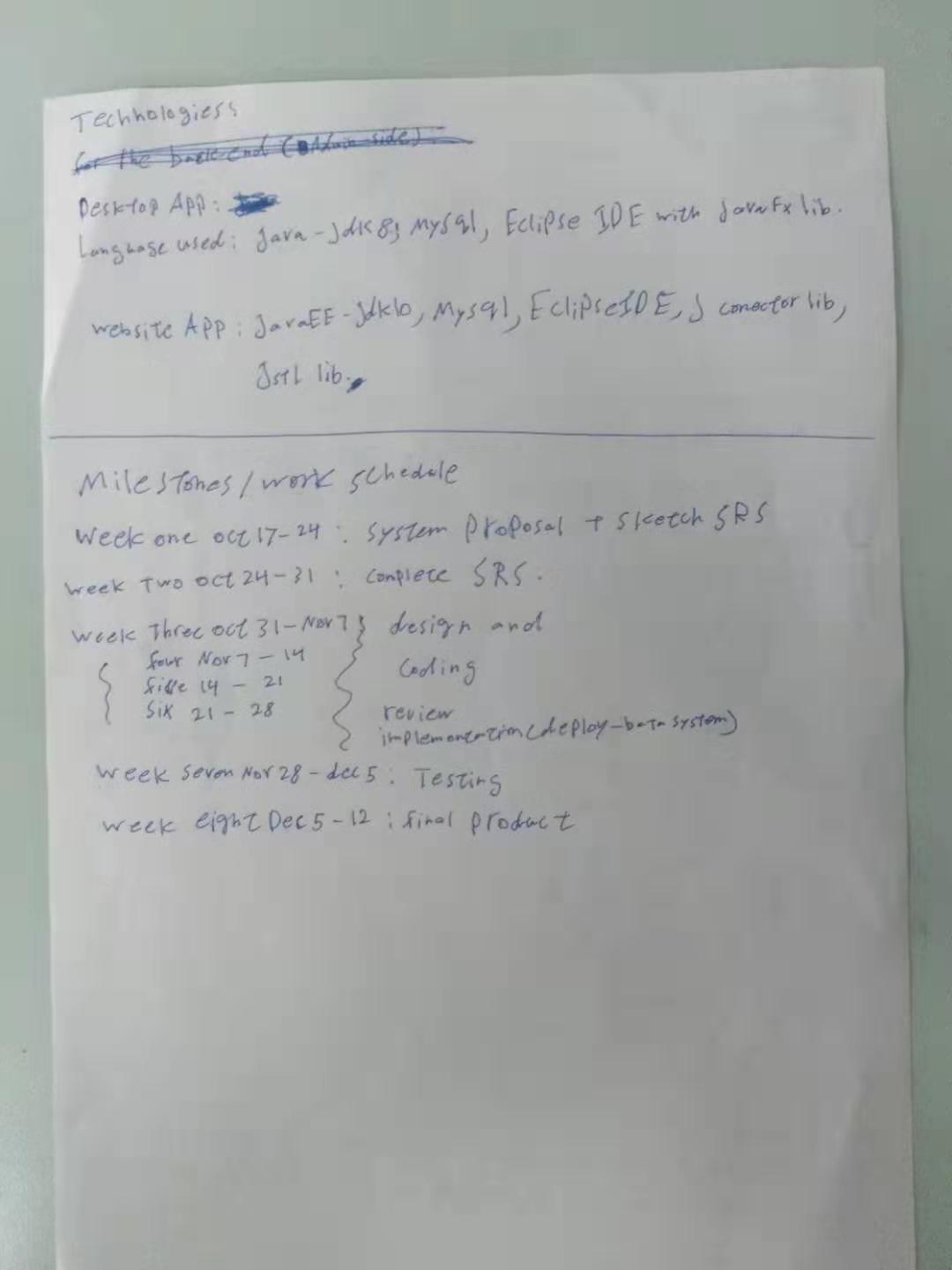
*The process used in this system development is the Water fall model, which will allow us to easily handle the cases of changes in the requirements, because it implements the traverse between steps back and forth as one of its main advantages.Hence, if we are in the coding step and found some features needed to be added/removed, we can go back to the document and do so.*

# A. Appendices

## A.1 Appendix 1

Project proposal paper:





## A.2 Appendix 2

Project process documents:

See the attached file (process document.doc).