INTERNATIONAL
UNIVERSITY OF SARAJEVO

CS308 Spring 2022

Software Requirements Specification Document

Project: Student Attendance System

Members:
Edin Žiga
Faruk Imamović
Nedim Kunovac
Mirza Redžepović

Instructor:
Dr. Kanita Karađuzović - Hadžiabdić

26.05.2022.

Table of Contents

ABSTRACT

For many years, professors have had the tedious task of manually keeping track whether students have attended their classes at university. The Student Attendance System intents to simply and modernize that tedious process through the use of students ID cards and a QR Code scanner. It will be implemented as a Web Application that any professor can access, much like the IUS Student Information System. The system provides one main functionality with multiple supporting functionalities, with the main consisting of the professor selecting a course and taking the attendance for that course. Other supporting functionalities include viewing course details, individual student attendance, and others. This project will be done using React and NodeJS with SQL, with the development timeline lasting from February 2022 to mid-May 2022.

Document Revision History

Rev1.0 March 22, 2022 – Initial version

Rev2.0 May 15, 2022 – Shortened Abstract, merged some functionalities, updated Use Case Diagram, corrected use cases and updated contribution table.

Rev3.0 May 26, 2022 – Added "Software Requirements Specification Document" to the cover page.

1 INTRODUCTION

Everything in our lives has been digitalized in some way, be it a simple or complex task. Something so simple such as taking attendance has completely gone under the radar at our university. This application is a solution to improve the way that task is accomplished. To replace the tedious task of manually writing on a piece of paper or filling up spreadsheets, this solution uses up less time and requires less work. By just simply scanning the barcode on the student card, the application does all the heavy lifting left. It saves the attendance record in a special table that can be looked up by the professor at any time. With this method, 2 major issues are tackled: time consumption and fraudulent activity. As mentioned before, the time this method would save creates opportunities to accomplish other tasks. This way, more focus can be placed on the students and help them understand lectures better. Additionally, fraudulent activity is not uncommon to bear witness to as students have a habit of signing their friends or doing something similar, therefore giving inaccurate attendance records. Technology is used to improve the way we do things, and that is exactly what this application does.

The entire process relies on just 2 things: the professor having the application installed on his/her device (PC, mobile phone, etc.) and having a personal account which will be created beforehand for everyone that needs it. After logging in with their account, the professor will have access to every feature the application provides, from recording the attendance and checking it to viewing their courses in detailed form. Moreover, it will provide snappy and easy to access information the professor would consider valuable, such as individual attendance records for students or managing the course information. The application will be able to run on a majority of devices and will require an internet connection. As far as the professor data is concerned, it will be encrypted to maximize security and will be managed by an Admin team. Up to 50 people can use the application at the same time, which is plenty enough to cover one time slot of lectures at the university.

2 SYSTEM FEATURES AND USE CASES

2.1 SYSTEM FEATURES

The application revolves around 5 features that the professor is capable of accessing. Below we can see all of them along with their User (Doel, 2018) and Functional Requirements (ReQtest, 2012).

FTR1: Account login: The professor is given the ability to log in to a specific assigned account in which all the relevant information regarding the courses that are assigned to him/her are contained. Each professor will be provided with an initial account.

UR1.1: Entering account details (Must have)

FR1.1.1: The application provides boxes in which the professor can enter his/her account information in order to access his/her account.

FR1.1.2: The application provides a button in order to confirm that the information has been entered and is to be verified.

FTR2: Taking attendance: The professor who has logged in through the application with his/her personal account is given the ability to record the attendance of a student in 2 different ways, scanning the barcode on the student card and manually entering the ID of the student. That action is emulated by the name of the student being recorded in a table specific to the attendance record of that specific day.

UR2.1: Scan barcode on student card with QR Code scanner (Must have)

FR2.1.1: The application shows a box in which the scanner is contained in order to make scanning easier.

FR2.1.2: The application displays the name and surname of the student that has been scanned as a mean of confirmation of that action.

UR2.2: Manually enter student ID (Must have)

FR2.2.1: The application shows a pop-up window which prompts the professor to enter the student ID.

FR2.2.2: The application displays the name and surname of the student that has been scanned as a mean of confirmation of that action.

FTR3: Course list and information: The professor who has logged in through the application with his/her personal account is given the ability to view all the courses he/she has been assigned to. Additionally, more information about the course is provided inside every tab.

UR3.1: Select wanted course (Must have)

FR3.1.1: The application displays all the courses the professor is assigned to as tabs that are stacked.

FR3.1.2: The application highlights the course which is selected to make the selection apparent to the professor.

UR3.2: View additional information for a course (Must have)

FR3.2.1: The application displays additional information for the course as a drop-down menu when the tab is clicked. The details that will be included but are not limited to are course description, venue, time and total attendance percentage.

FTR4: Attendance list: The professor who has logged in through the application with his/her personal account is given the ability to view the complete attendance list for each course and also view just a specific day on which the class was held in order to make the information that is searched for easier to find.

UR4.1: Complete attendance list for a specific course (Should have)

FR4.1.1: The application enables the professor to see the complete attendance list presented as a table.

UR4.2: Per day record of attendance for a specific course (Should have)

FR4.2.1: The application will give the option to select a specific day on which class has been conducted for a specific course. The attendance will be displayed in a table in which the status will be marked with a '✓' as attended and '✗' as not attended inside a table.

FTR5: Attendance record for each student: The professor who has logged in through the application with his/her personal account is given the ability to track the total attendance record for each student to therefore see whether or not the student has an acceptable attendance record.

UR5.1: Percentage of classes attended for each student (Should have)

FR5.1.1: The application will display the percentage of classes attended for each student in order to track whether they have an acceptable attendance record.

FR5.1.2: The application will highlight the tab of the student who has an unsatisfactory attendance record.

UR5.2: Search function (Could have)

FR5.2.1: The application shows a box in which the professor could enter a specific student name that he/she is searching for.

2.2 USE CASES

The system has 2 primary user classes, which includes professor and administrator. Each user class will have different levels of access and views of the system. Below we can see the use case diagram for our system (Figure 1).

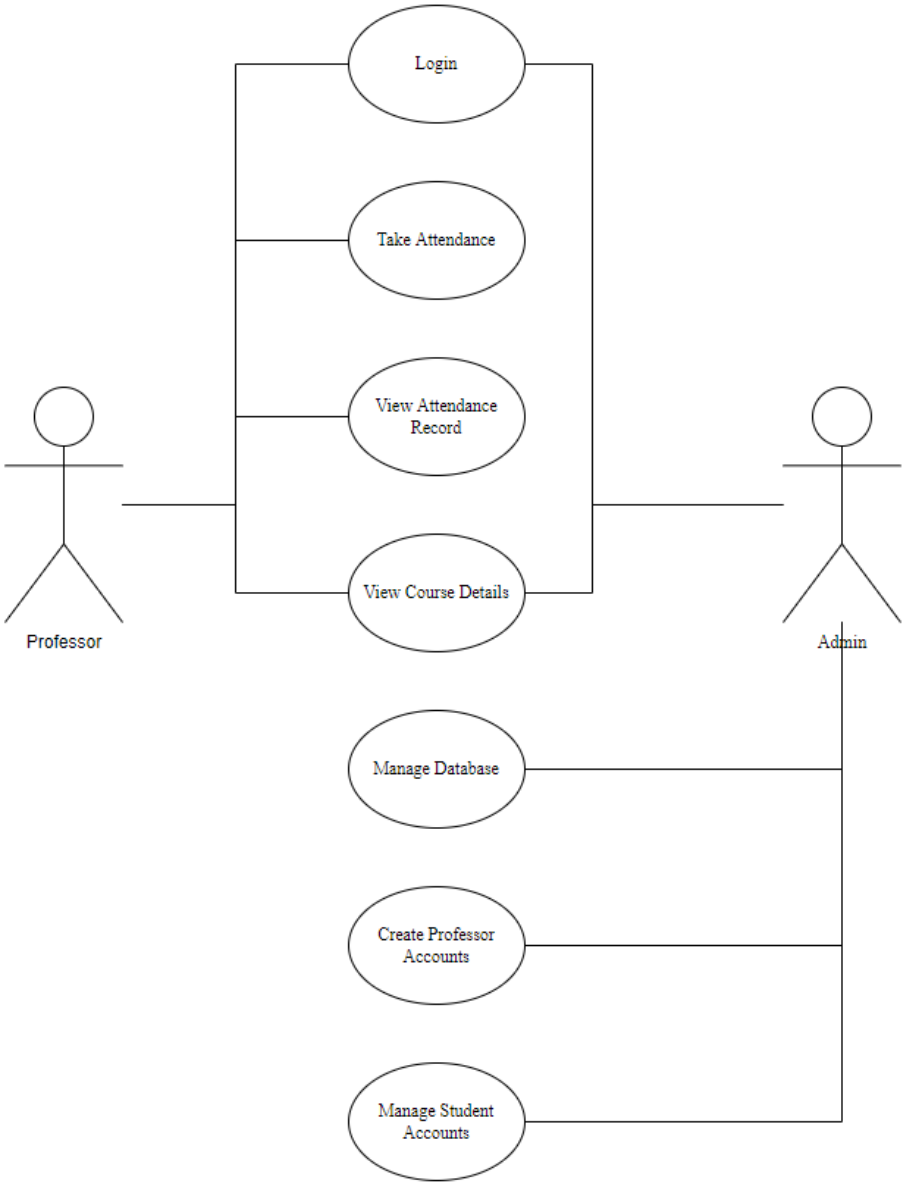


Figure 1. Use Case Diagram

Use Case	Description	Correspondence with UR	UR Description
UC1	Log-in	UR1.1	Entering account details
UC2	Taking attendance	UR2.1 & UR2.2	Scanning or manually entering
UC3	Viewing course details	UR3.1	Select wanted course
UC4	Viewing attendance record	UR4.1	Complete attendance list
UC4.1	Viewing individual student attendance records	UR5.1	Percentage attended for each student
UC4.2	Searching attendance record	UR5.2	Search function
UC5	Managing database	/	/
UC6	Creating professor accounts	/	/

Table 2. Use Cases

2.2.1 UC1: Log-in (UR1.1 Entering account details)

Description: The purpose of Log-in use case is describing how a professor would Log-in to the system.

Priority: Must have

Pre-condition: Internet connection, browser.

Post-Condition: An account is created for the Professor.

Basic flow:

1. The professor types the homepage URL.
2. The professor inputs their username in the username field
3. The professor inputs their password into password field
4. The professor clicks “Log in” button

What can go wrong: In case the professor forgets their password, an error message will be displayed, warning the professor that they inputted the wrong password. If the professor cannot remember their password, an option “forgot your password” will be available, which will prompt password recovery via email.

2.2.2 UC2: Taking Attendance (UR2.1 & UR2.2 Scanning or manually entering)

Description: The purpose of the Taking Attendance use case is describing how a professor would perform attendance taking.

Priority: Must have

Pre-condition: Internet connection, browser, device has camera, logged in.

Post-Condition: Student Attendance record is updated.

Basic flow:

1. The professor performs the log-in process
2. The professor selects for which course he wants to take attendance for by pressing “take attendance” button next to the desired course
3. The professor gathers student ID cards
4. The professor scans ID cards by pointing their camera at the barcode on the student ID cards
5. The professor is prompted by a notification when a card is scanned
6. The professor clicks “done” button once all cards are scanned

What can go wrong: In case that a student ID card cannot be scanned (damaged, dirty etc.), the professor has a “manual entry” button where the professor can type in the student ID and manually add the student to the attendance record.

2.2.3 UC4 Viewing Course Details (UR3.1 Select wanted course)

Description: The purpose of the viewing course details use case is for the professor to be able to see relevant information about each course using a dropdown on the same page.

Priority: Could Have

Pre-condition: Internet connection, browser, logged in.

Post-Condition: Professor sees general information about course, including time, location etc.

Basic flow:

1. The professor clicks on specific course he wants to view the details of so the dropdown can appear

What can go wrong: Some courses may not have been added to the course list yet, this cannot be solved or detected by the system, in this case the professor needs to contact the admin.

2.2.4 UC3: Viewing Attendance Record (UR4.1 Complete attendance list)

Description: The purpose of the Viewing Attendance Record is for the professor to be able to see how many times each student attended a lecture,

Priority: Must have

Pre-condition: Internet connection, browser, logged in.

Post-Condition: The professor is presented with a list of all students from a specific course and can see how many times each student attended

Basic flow:

1. The professor performs the log-in process
2. The professor selects for which course he wants to view the attendance for by pressing "view attendance record" button next to the desired course

What can go wrong: Some students are not enrolled or added to the system yet, this cannot be solved or detected by the system, in this case the professor needs to contact the admin.

2.2.4.1 UC3.2 Viewing Individual Student Attendance Records (UR5.1 Percentage attended for each student)

Description: The purpose of Viewing individual student attendance records use case is for the professor to be able to see all the times a student attended lectures.

Priority: Could have

Pre-condition: Internet connection, browser, logged in, is in the view attendance record view.

Post-Condition: Professor can see specific student attendance record.

Basic flow:

1. The professor clicks on specific student he wants to view the attendance record of

What can go wrong: Some students are not enrolled or added to the system yet, this cannot be solved or detected by the system, in this case the professor needs to contact the admin.

2.2.4.2 UC3.1 Search (UR5.2 Search function)

Description: The purpose of Search use case is for the professor to be able to search specific student attendance records.

Priority: Could have

Pre-condition: Internet connection, browser, logged in, is in the view attendance record view.

Post-Condition: Professor can see specific student attendance record.

Basic flow:

1. The professor clicks on the search bar
2. The professor types the desired student's name

What can go wrong: Some students are not enrolled or added to the system yet, this cannot be solved or detected by the system, in this case the professor needs to contact the admin.

2.2.5 UC5 Managing database

Description: The purpose of managing database use case is allowing the admin to add new courses for professors and adding students to courses

Priority: Must have

Pre-condition: Internet connection, browser.

Post-Condition: Admin has added courses and/or students

Basic flow:

1. The admin logs into the database management system
2. The admin can now choose whether to add new course or edit existing course

What can go wrong: There should be no perceivable issues as the admin is technically experienced and the system is intuitive.

2.2.6 UC6 Create Professor Accounts

Description: The purpose of Creating professor accounts use case is allowing the admin to create new professor accounts for more security.

Priority: Must have

Pre-condition: Internet connection, browser.

Post-Condition: Admin has created new professor account

Basic flow:

1. The admin logs into the database management system
2. The admin goes to the professor account creation tool
3. The admin enters relevant information name, surname, ID, email, password etc.
4. The admin clicks the "done" button

What can go wrong: There should be no perceivable issues as the admin is technically experienced and the system is intuitive.

2.3 NON-FUNCTIONAL REQUIREMENTS

Having non-functional requirements is key in keeping our project on a steady stream. Non-functional requirements provide an easy overview of what needs to be kept in mind while working, *Table 1* is a table which includes the names of our non-functional requirements, as well as a definition and more details (Rome, 2020). These non-functional requirements state that the system is expected to perform well under the intended use cases mentioned before (IBM, 2021).

Table 2. Non-functional requirements

Requirement	Definition	More Details
NFR1. Hardware Interface 1	Minimum hardware requirements for the server.	Operating System: Windows Processor: 6 core CPU (2.4Ghz or higher) with iGPU DISK Space: 256GB RAM: 8GB Query Cache: 2GB
NFR2. Hardware Interface 2	Minimum hardware requirements for user devices.	Any device able to run a browser and internet connection, should have camera (Windows, Linux, macOS, Android, iOS etc.).
NFR3. Software Interface	Software requirements for the system.	Database: mySQL Frontend: HTML, CSS, JavaScript, React, Bootstrap Backend: NodeJS
NFR4. Communication Interface	User needs to be connected to internet to access system via browser.	Internet connection.
NFR5. Performance	The number of users the system needs to be able to handle.	The system needs to be able to handle more than 50 users at the same time.
NFR6. Security	How the data and access will be secured.	All of the data will be encrypted, access will be secured by the fact that only admin can make new professor accounts.
NFR7. Reliability	System needs to be reliable in order for easy use.	Reliability will be achieved by optimizing our queries for fast and easy use, as well as maintaining the server regularly.
NFR7. Usability	How users interact with system.	The design will be intuitive and easy to use.
NFR8. Capacity	How the system will scale into the future.	256GB is more than enough to store data for a few semesters, later on can be backed up on other drives.
NFR9. Environmental	What kind of environments the system will perform in.	Every day in multiple classroom settings, end of semester when all professors are looking up attendance.

2.4 SYSTEM EVOLUTION

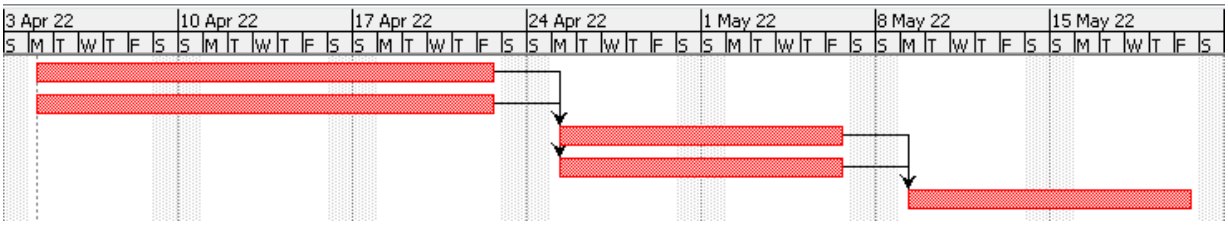
The Student Attendance System will be developed under the assumption that a functional real-time barcode scanner can be provided through the use of the aforementioned frontend technologies. Should the implementation of this feature prove to be impractical (hardware vise, performance wise or too time consuming), other methods of implementation will be researched. User hardware evolutions are not expected for the Student Attendance System, while server hardware evolutions might be, as this is entirely dependent on how the web application is received by the end users. The other requirements are not expected to change due to the nature of the application, but it must me noted that the user needs for additional features (besides the features provided in this document) are not set in stone, as the application is functionality and convenience based.

3 RELEASE PLAN

The release plan contains the timeline we intend to follow, as well as the responsibilities each team member will be required to do (Makadia, 2021).

Category	Requirement	Feature Description	Duration (in Days)	Increment	Dependencies	Priority	Release
FTR1	FR1.1.1	FirstTimePass	3	1	/	Medium, should have	1
	FR1.2.1	Login Form	1	1	/	High, must have	1
	FR1.2.2	Login Button	1	1	FR1.2.1	High, must have	1
FTR2	FR2.1.1	Box for scanner	8	3	FR3.1.2	High, must have	1
	FR2.1.2	Student added notification	1	1	FR2.1.1	High, must have	1
	FR2.2.1	Manual Add dialog box	2	2	FR2.1.1	High, must have	1
	FR2.2.2	Student added notification for manual	1	1	FR2.2.1	High, must have	1
FTR3	FR3.1.1	Display courses	6	2	FR1.2.2	High, must have	1
	FR3.1.2	Course Highlight	1	1	FR3.1.1	High, must have	1
	FR3.2.1	Course Dropdown	1	1	FR3.1.1	High, must have	1
	FR3.2.2	Add additional info for courses	3	2	FR3.1.1	High, must have	1
FTR4	FR4.1.1	Attendance list in table form	4	2	FR3.1.1	Medium, should have	1
	FR4.2.1	Select day	1	1	FR4.1.1	Medium, should have	1
	FR4.2.2	Display attendance for selected day	2	2	FR4.2.1	Medium, should have	1
FR5	FR5.1.1	Display specific student percentage	1	1	FR3.1.1	Medium, should have	1
	FR5.1.2	Highlight unsatisfactory students	2	1	FR3.1.2	Medium, should have	1
	FR5.2.1	Search students	6	2	FR5.1.1	Low, could have	1

		Name	Duration	Start	Finish	Predecessors
1		FTR1 - Nedim & Mirza	15 days?	4/4/22 8:00 AM	4/22/22 5:00 PM	
2		FTR2 - Edin & Faruk	15 days?	4/4/22 8:00 AM	4/22/22 5:00 PM	
3		FTR3 - Nedim and Mirza	10 days?	4/25/22 8:00 AM	5/6/22 5:00 PM	1
4		FTR4 - Edin and Faruk	10 days?	4/25/22 8:00 AM	5/6/22 5:00 PM	2
5		FTR5 - Everyone	10 days?	5/9/22 8:00 AM	5/20/22 5:00 PM	3;4



CONTRIBUTION TABLE

Project Name: Student Attendance System

Team members: Edin Žiga, Faruk Imamović, Nedim Kunovac, Mirza Redžepović

MILESTONE: **Software Requirements Document/ Software Design Document/Coding/Presentation preparation**

Date: 26.05.2022.

Team member name	**Task Assigned**	**Status of the tasks**
Edin Žiga	SRS Document REV1.0 review, lead release plan development, REV2.0 updates and finalization, REV3.0 updates	Completed
Faruk Imamović	SRS Document REV1.0 review, lead use case development, REV2.0 general review	Completed
Nedim Kunovac	SRS Document REV1.0 task distribution, review and finalization, lead system features development, REV2.0 contribution table review	Completed
Mirza Redžepović	SRS Document REV1.0 review, system evaluation development, REV2.0 corrected diagram design	Completed

References

Adobe Experience Cloud. (n.d.). *Waterfall Methodology*. Retrieved from Adobe Experience Cloud: https://www.workfront.com/project-management/methodologies/waterfall [Last accessed: 22.03.2022.]

Doel, R. V. (2018, January). *User Requirements*. Retrieved from PERFORMANCE VALIDATION: https://perfval.com/user-requirements/ [Last accessed: 22.03.2022.]

IBM. (2021). *Hardware Requirements for Database and Application Servers for Emptoris Spend Analysis*. Retrieved from IBM: https://www.ibm.com/docs/en/strategicsm/10.0.4?topic=esar-hardware-requirements-database-application-servers-emptoris-spend-analysis [Last accessed: 22.03.2022.]

Makadia, M. (2021, December). *7 Steps To Create a Successful Release Plan*. Retrieved from BUSINESS 2 COMMUNITY: https://www.business2community.com/b2b-marketing/7-steps-to-create-a-successful-release-plan-02446117 [Last accessed: 22.03.2022.]

ReQtest. (2012, April). *Why is the difference between functional and Non-functional requirements important?* Retrieved from ReQtest: https://reqtest.com/requirements-blog/functional-vs-non-functional-requirements/#:~:text=What%20are%20those%2C%20and%20how,what%20the%20system%20should%20do. [Last accessed: 22.03.2022.]

Rome, P. (2020, August). *What are Non Functional Requirements — With Examples*. Retrieved from PERFORCE: https://www.perforce.com/blog/alm/what-are-non-functional-requirements-examples [Last accessed: 22.03.2022.]