Software Requirements

Specification

for
Automated Course Scheduling System → ACSS

Version 1.2 approved

Prepared by Akhat Suleimenov, Dariga Shokayeva, Madina Yeleukina, Shyngys Karishev

NYUAD

08.12.2021

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Dariga Shokayeva	06.12.2021	Revise the requirements	v1.1
Akhat Suleimenov	08.12.2021	Finalize the requirements	v1.2

1. Introduction

1.1 Purpose

The purpose of the following document is to specify the software requirements of an automated course scheduling system. Release number: v1.2. The given SRS aims to describe the revised system requirements of an entire ACSS.

1.2 Document Conventions

The document uses the following conventions:

NYUAD	New York University Abu Dhabi
NYU	New York University
ACSS	Automated Course Scheduling System
IDE	Integrated Development Environment
VS	Visual Studio
SRS	Software Requirements Specifications

1.3 Intended Audience and Reading Suggestions

This project aims to build an automated course scheduling system for NYUAD students, and it is restricted within the university premises. This project is supervised by Professor Mai Oudah and reflects the team's efforts to develop the final project for the Software Engineering course. The rest of the document would present interface requirements, functional and non-functional requirements of ACSS. The table of contents reflects the order in which SRS is structured. This project is designed for NYUAD students and is useful as well as to the Albert system developers and NYUAD staff responsible for the student well-being. While the intended audience includes students, university staff and developers, it is assumed that the SRS document is primarily made for the project builders and professors responsible for the associated course.

1.4 Product Scope

The purpose of the automated course scheduling system is to ease course registration and to create a convenient and easy-to-use application for students, preparing to register for NYUAD courses soon. The system is based on the course information provided on the NYU Albert system with its details of the course category, timings and professors. We will have a scheduling system that would ask students to authorize, import the data from Albert and build a number of possible course schedules based on the students' course selections. Above all, we hope to provide a comfortable user experience along with a functional and useful application that would enhance students' course registration experience.

1.5 References

1. Software Requirements Specification Document with Example. Retrieved from https://krazytech.com/projects
2. SRS Template - IEEE Full version. Software Engineering course supplementary materials, Week 3: Communication

2. Overall Description

2.1 Product Perspective

The purpose of the ACSS is to make course registration easier and to create a convenient web-application for students, preparing to register for NYUAD courses soon. The system is based on the course information provided on the NYU Albert system with its details of the course category, timings and professors. Therefore, there will be communication between these two platforms. Thus, the system would be based on the larger system (Albert) while providing new functionalities oriented to students.

2.2 Product Functions

The software will be available only for NYU students, therefore the software will have an authorization process that will ask for the NYU account for verification. After successful verification, the software could be used.

With the software, the users will be able to search for classes and create a flexible schedule. The result will be based on the criteria the user inputs. There are several search criteria, and it will be possible for the system to manage the classes that have such criterias. The users may then choose the classes to use them in the schedule that will be then created by the software.

After the search of the classes or picking classes from the whole list, user can change the schedule, watch for possibilities of permutations of schedules, allowing the user to select the schedule that fits the student's needs better.

2.3 User Classes and Characteristics

Search class – contains all the parameters and functions needed to search courses.
Schedule class – contains the attributes and functions of the schedule.
Login class – contains the information and functions needed to authorize user.
User class – contains the information about users and their methods.
Course class – contains the information about courses and their methods.

2.4 Operating Environment

Operating system: Windows and macOS
Since the system is online, the hardware system will be all the hardware able to connect to the internet and support web browsers.

2.5 Design and Implementation Constraints

The Internet connection is a constraint for the application. Since the application fetches data from the database over the Internet, it is crucial that there is an Internet connection for the application to function. Yet, the backup file with the course information may be created regularly to make the system less dependent on the operation of Albert and/or Internet connection.

2.6 Assumptions and Dependencies

The dependency on NYU Albert might result in an issue in the program, since ACSS is using the data from Albert. The possible solution is to create backup for the course information imported from Albert regularly as stated above.

3. External Interface Requirements

3.1 User Interfaces

- Front-end software: HTML, CSS, JavaScript.
- Back-end software: JavaScript, Python.

3.2 Hardware Interfaces

Since the system is online, the hardware system will be all the hardware able to connect to the internet. For example, WAN-LAN, Ethernet Cross-Cable, etc. Communication protocols are described in section 3.4.

3.3 Software Interfaces

The following table presents the software we have decided to use for the automated course scheduling online system.

Software	*Description*
Operating System	We have chosen Windows and macOS as both of them are very commonly used by our team members
Development Platform	As only those who have a nyu.edu account must be able to access our system, we will add Google Authorization so that only users with NYU affiliated account may use the system
IDE	We may implement the system using Visual Studio IDE because of its simplicity and wide range of functionality
Modeling Web Tool	We have chosen Creately to build models of our system because it provides an availability to work together on the same projects
HTML, CSS, Dart, JavaScript	We have chosen to use HTML, CSS, JavaScript to develop front-end of the system because they are commonly used for that purposes
JavaScript, Python	We have chosen to use Python and JavaScript to develop back-end of the system because of their simplicity

3.4 Communications Interfaces

Our system should be compatible with all common web browsers including but not limited to Google Chrome, Safari, Mozilla Firefox, Internet Explorer. The system will use the HTTP protocol for communication over the internet and the TCP/IP protocol suite for communication over the intranet. There may appear privacy issues regarding accessing courses offered NYUAD which will be resolved by authorization process. We will use external Google API to ensure that the authorization process is secure.

4. System Features

4.1 Create Schedules

4.1.1 Description and Priority

The system will have a feature of generating schedules based on the courses selected by user from the course offerings and/or searched classes. All the possible variations of schedules should be generated at once, but the user should see the next or the previous variation of the schedule on the request. The priority of this task is Medium - High as it is the main feature of the system.

4.1.2 Stimulus/Response Sequences

The behavior defined for this feature will be stimulated by the user's click on button after choosing all the courses the user wants to be included in the schedule.

4.1.3 Functional Requirements

REQ-1: The system should show several possible schedules generated with the courses selected by user

REQ-2: The system should display error message if no courses were chosen by the user

REQ-3: The system should create schedules with no overlapping classes.

REQ-4: The system should show a message dialog if the user adds many (more than 9) courses.

REQ-5: The system should create load from json files that hold all the information about courses.

REQ-6: The system should create an empty json file if no possible schedule is possible.

REQ-7: The system should display error message if required json files are missing

4.2 Import Course Information

4.2.1 Description and Priority

The system would need to import course information for all the offered NYUAD courses from Albert. The imported course information should at least include the weekdays and time of the particular class. This system feature has high priority and high risk, as the system output would be based on whether the import from Albert was successful or not.

4.2.2 Stimulus/Response Sequences

The successful student authorization calls for the subsequent import of course information.

4.2.3 Functional Requirements

REQ-1: The import should start after successful student authorization

REQ-2: The imported information should include class timings of each imported course

REQ-3: The system should be able to check if the import is complete

REQ-4: The system should be able to work offline or with not operating Albert

REQ-5: The system should be able to load data from the backup file if the parsed data is malfunctional

REQ-6: The system should be able to update backup file when the parsed data is valid

4.3 Authorization

4.3.1 Description and Priority

This feature allows users to log in the system. No registration is needed because only NYU emails are valid. Hence, users will be able to access the course scheduling system. It's a high priority feature, since without it users will not be able to use the system at all.

4.3.2 Stimulus/Response Sequences

User logins through his NYU email then the system will load the main page with all the course information and search feature.

4.3.3 Functional Requirements

REQ-1: The system should be able to validate if the email address is of type "nyu.edu"

REQ-2: The system should be able to log in the user into the system

REQ-3: The system should be able to output the error message if the credentials are wrong

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The system must be interactive, and the delays involved should be minimal. So in every action-response of the system, there are no immediate delays. The system should not have delays in the process of showing the course catalogue, and in the stage of picking courses. However, during the stage of creating all the possible schedules, the system may take time to construct all possible schedules with no more than 5 seconds. Switching between schedules should be implemented without any delays. All other functions such as choosing courses, searching courses by names and/or codes should be implemented without any delays too.

5. 2 Safety Requirements

In case the system accidentally crashes, and all data in the databases get erased, the system will have to restore its past copy of the database from the archival storage. In doing so, the system should recreate a more recent state by reapplying or retrying compromised transaction operations from the backed-up log to the point of failure.

5.2 Security Requirements

In order to keep safe the information of the NYU courses, only NYU users will be able to access the program. All users will have to go through the authentication process at the beginning of the session. The email used for authentication should be an NYU email. Otherwise, the system will not let them authenticate.

5.3 Software Quality Attributes

- The system should be adaptable. The courses saved in the database should be up to date with the courses available at Albert. Hence, if a course is added/deleted at Albert, the system should be updated accordingly.
- A delay of one day may be appropriate.
- The system should be available to use. All NYU students should be able to use it without any difficulties or restrictions.
- The system should be correct. If a course is not on Albert, the course should not be in the system. All information should match the exact information from Albert.
- The schedule created should be valid according to the Albert rules(except for the knowledge of the pre-requisite courses).
- The system should be reliable. The admins should maintain the correct course information all the time.
- The system should be usable. The course scheduling system should satisfy a maximum number of users' needs.

5.4 Business Rules

- All users need a valid NYU email address

- An email is considered valid if the user can prove they have access to the email address associated with the user account.
- All users must add at least one course to the catalogue.
- If the create schedule button is pressed and the course catalogue is not empty, the scheduling system should create a list of schedules and output them in a scrolling window.
- If a course name is pressed, the drop-down window with sessions to pick from(checkboxes) should appear.
- If a filter is activated, the course catalogue should be updated accordingly to show only the courses that fit the filter chosen.

Appendix A: Glossary

NYUAD	New York University Abu Dhabi
NYU	New York University
ACSS	Automated Course Scheduling System
IDE	Integrated Development Environment
VS	Visual Studio
SRS	Software Requirements Specifications