1.Introduction

Student Classroom Manager is a standalone desktop application that can manage student-classroom assignments. There are various features such as adding and removing a student, changing classrooms according to capacity and searching for available classrooms, courses and instructors. The program designed as stand-alone program that is independent of all other programs and services.

USER REQUIREMENTS:

***Functional Requirement 1*:** The user shall be able to add a student.
Rationale: If a student wants to enroll a course then some user must add this student to that course. But only some users that has given access can add students.

Functional Requirement 2: Course and Classroom classes are connected to each other by a HAS-A relationship.
Rationale: The relationship between Course and Classroom is composition. For example every course has a classroom and every classroom has a course. In other words, there is a HAS-A (Composition) relationship.

Functional Requirement 3: The user should be able to add a student to a class, within the lecture's assigned classes, with appropriate capacity, move a class to another with larger capacity.
Rationale: When a user with given access to manipulating classes, courses and students tries to add a new student to a class; first the capacity of the course's first section will be checked. If it is available, the new student will be placed in that class. In the case of the first section being full, the other section or sections will be checked orderly. Assuming that all available sections' classes are full,

one of the existing section's class will be moved to another class with greater capacity.

Functional Requirement 4: The user shall be able to withdraw a course.
Rationale: If a student wants to withdraw a course then some user that has given access can withdraw courses. If a student withdraws a course, then there are available places for new students who wants to enroll that course. Which can be used in add drop or adding a new student.

Functional Requirement 5: According to the users search input, appropriate programs will be seen on the screen.
Rationale: If the user searches for a course, a program related to the course will be displayed. The user's registered classroom, time and instructor will determine the appropriate time to be displayed in this way. If the user searches for a classroom, user will be able to see the course in that class throughout the week. If the user continues to search for the instructor, this time user will be able to see the weekly program of the instructor.

Non-functional Requirement 1: The user should be able to utilize the system properly by consulting the manual.
Rationale: A detailed guide will be provided to user, eliminating the need for trial and error learning.

SYSTEM REQUIREMENTS:

Functional Requirement 6: The system shall be able to read from available file.

Rationale: The system shall be able to create data for courses, classrooms and instructors. It does this by reading data from given two files.

***Functional Requirement 7*:** The system shall be able to save the students', classes' and courses' latest changed versions to a file on the file system.
Rationale: This fundamental feature is necessary to give the generated class matches durable data storage.

Functional Requirement 8: The system should be able to load the most recently updated versions of students, classrooms and courses saved in a file from the file system.
Rationale: To load and continue working on a ClassMatches that has already been generated, this fundamental feature is necessary.

Non-functional Requirement 2: The program must use a public git repository.

Non-functional Requirement 3: The system shall be able to run on a Windows system.
Rationale: While the primary target platform is Windows, the chosen programming language may also allow the application to run on other platforms. The application is not reliant on features specific to any particular operating system.