# GIT Department of Computer Engineering
# CSE 222/505 - Spring 2020
# Homework # Report

**Student Name**
**Student Number**

## 1. SYSTEM REQUIREMENTS

Write detailed system requirements. System requirements are detailed descriptions of the software system's functions, services, and operational constraints. The system requirements document (sometimes called a functional specification) should define exactly what is to be implemented. It may be part of the contract between the system buyer and the software developers. (Write detailed functional and non-functional requirements, etc.)

Here are some useful links:

https://cgi.csc.liv.ac.uk/~coopes/comp201/handouts/SE_L4.pdf

https://www.geeksforgeeks.org/software-engineering-classification-of-software-requirements/

http://www.inf.ed.ac.uk/teaching/courses/ip/CS2Ah0405-SoftwareRequirements.pdf

## 2. USE CASE DIAGRAMS

Create use case diagrams to describe the behavior of your system. Use case diagrams are usually referred to as behavior diagrams used to describe a set of actions (use cases) that some system or systems (subject) should or can perform in collaboration with one or more external users of the system (actors). Each use case should provide some observable and valuable result to the actors or other stakeholders of the system.

Useful links to create good use case diagrams:

https://www.tutorialspoint.com/uml/uml_use_case_diagram.htm

https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-use-case-diagram/

## 3. CLASS DIAGRAMS

Create class diagrams to describe the structure of your system. In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.

Useful links for class diagrams:

https://www.tutorialspoint.com/uml/uml_class_diagram.htm

https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-class-diagram/

## 4. OTHER DIAGRAMS

You can add any other diagrams in order to describe your system better.

https://www.tutorialspoint.com/uml/uml_standard_diagrams.htm

## 5. PROBLEM SOLUTION APPROACH

Write your problem solution approach. To solve a problem, you should define the problem, divide it into sub-problems, create a plan of steps, try if your approach works, etc. You can find useful articles on problem solving by googling "problem solving in software engineering".

## 6. TEST CASES

Write test cases to check whether your system works properly. A test case is a set of conditions or variables under which a tester determines whether the software satisfies **requirements** and **functions** properly.

Here are good definitions and examples to create test cases:

https://www.guru99.com/test-case.html

## 7. RUNNING AND RESULTS

Run your system, check your test cases, add your results as screenshots and tables.