German International University
Faculty of Informatics and Computer Science
Dr. Iman Awaad
Eng. Ahmed Sherif
Amir Haythem
Mohamed Essam
Yassein Eldamasy

Software Engineering, Winter 2025
Practice Assignment 3

Exercise 3-1

Mention at least seven non functional requirements of system and explain the importance of each non functional requirements.

Solution:

Performance
Performance defines how fast a software system or a particular piece of it responds to certain users' actions under a certain workload. In most cases, this metric explains how long a user must wait before the target operation happens (the page renders, a transaction is processed, etc.) given the overall number of users at the moment. But its not always like that. Performance requirements may describe background processes invisible to users, e.g. backup. But lets focus on user-centric performance.

Example of performance requirements:

The landing page supporting 5,000 users per hour must provide 6 second or less response time in a Chrome desktop browser, including the rendering of text and images and over an LTE

Scalability
Scalability assesses the highest workloads under which the system will still meet the performance requirements. There are two ways to enable your system scale as the workloads get higher: horizontal and vertical scaling.

Horizontal scaling is provided by adding more machines to the pool of servers. Vertical scaling is achieved by adding more CPU and RAM to the existing machines.

Example of scalability requirements:

The system must be scalable enough to support 1,000,000 visits at the same time while maintaining optimal performance. connection.

Reliability
Reliability specifies how likely the system or its element would run without a failure for a given period of time under predefined conditions. Traditionally, this probability is expressed in percentages. For instance, if the system has 85 percent reliability for a month, this means that during this month, under normal usage conditions, thereâs an 85 percent chance that the system wonât experience critical failure.

As you may have guessed, itâs fairly tricky to define critical failure, time, and normal usage conditions. Another, somewhat simpler approach to that metric is to count the number of critical bugs found in production for some period of time or calculate a mean time to failure.

Example of reliability requirements:

The system must perform without failure in 95 percent of use cases during a month.

Maintainability
Maintainability defines the time required for a solution or its component to be fixed, changed to increase performance or other qualities, or adapted to a changing environment. Like reliability, it can be expressed as a probability of repair during some time. For example, if you have 75 percent maintainability for 24 hours, this means that thereâs a 75 percent chance the component can be fixed in 24 hours. Maintainability is often measured with a metric like MTTRS â the mean time to restore the system.

Example of maintainability requirements:

The mean time to restore the system (MTTRS) following a system failure must not be greater than 10 minutes. MTTRS includes all corrective maintenance time and delay time.

Availability
Availability describes how likely the system is accessible to a user at a given point in time. While it can be expressed as an expected percentage of successful requests, you may also define it as a percentage of time the system is accessible for operation during some time period. For instance, the system may be available 98 percent of the time during a month. Availability is perhaps the most business-critical requirement, but to define it, you also must have estimations for reliability and maintainability.

Example of availability requirements:

The web dashboard must be available to US users 99.98 percent of the time every month during business hours EST.

Security
Security is a non-functional requirement assuring all data inside the system or its part will be protected against malware attacks or unauthorized access.

Usability
Usability indicates How effectively and easy users can learn and use a system

Example of usability requirements:

The error rate of users submitting their payment details at the checkout page mustn't exceed 10 percent.

Exercise 3-2

1. What are non functional requirements of Facebook ?

Solution:

1. Performance
2. Scalability
3. Reliability
4. Maintainability
5. Availability
6. Security
7. Usability

2. An Informatics website is going to be build for Informatics GIU students. What are non functional requirements of Informatics website ?

Solution:

1. Performance
2. Reliability

3. Maintainability
4. Availability
5. Security
6. Usability

Exercise 3-3

What are the differences between Unified Modeling Language (UML) diagram and use case diagram ?

Solution:

UML, short for Unified Modeling Language, is a standardized modeling language consisting of an integrated set of diagrams, developed to help system and software developers for specifying, visualizing, constructing, and documenting the artifacts of software systems, as well as for business modeling and other non-software systems. The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems. The UML is a very important part of developing object oriented software and the software

There are two types of UML diagrams : Structure diagrams and behavior diagram.

Structure diagrams show the static structure of the system and its parts on different abstraction and implementation levels and how they are related to each other.

Examples: Class Diagram, Component Diagram, Deployment Diagram, Object Diagram, Package Diagram, Composite Structure Diagram, Profile Diagram.

Behavior diagrams show the dynamic behavior of the objects in a system, which can be described as a series of changes to the system over time.

Examples: Use Case Diagram, Sequence Diagram, Activity Diagram, State Machine Diagram, Communication Diagram, Interaction Overview Diagram, Timing Diagram.

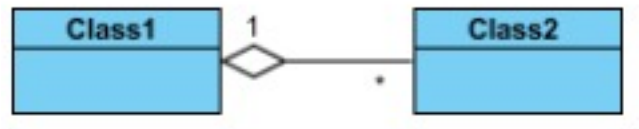
Exercise 3-4

Explain the differences between Aggregation and Composition dependencies in a class diagram.

Solution:

Aggregation:

A special type of association.
It represents a part of relationship.
Class2 is part of Class1.
Many instances (denoted by the *) of Class2 can be associated with Class1.
Objects of Class1 and Class2 have separate lifetimes.
A solid line with an unfilled diamond at the association end connected to the class of composite



Example: Class class and Class students ,if class got destructed students will exist.

Composition:

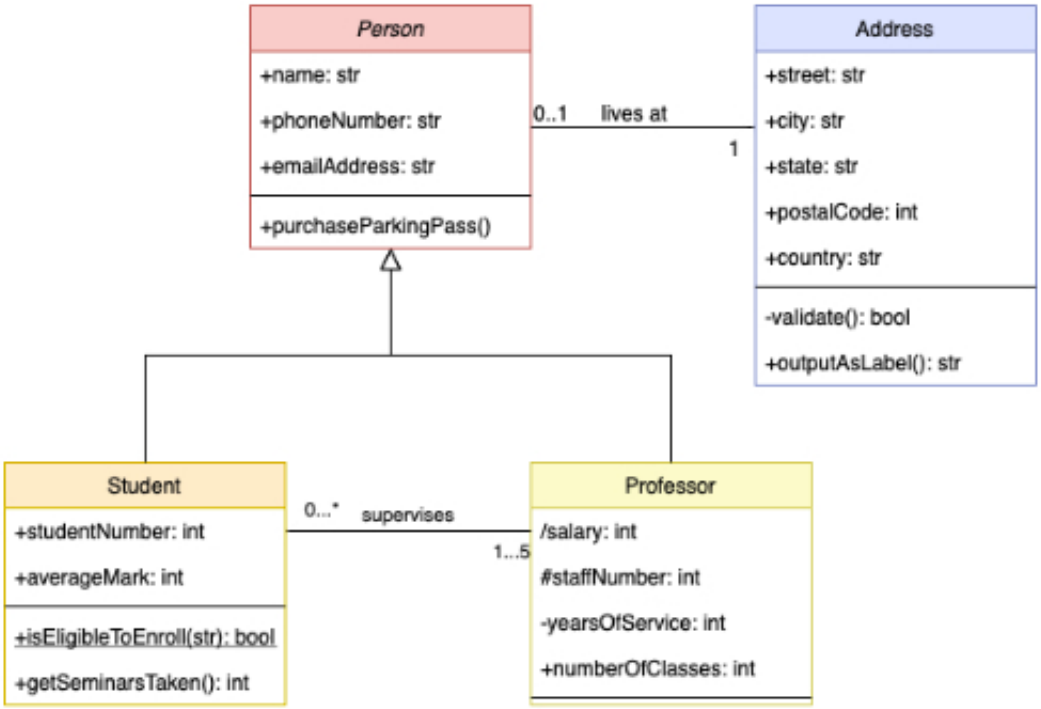
A special type of aggregation where parts are destroyed when the whole is destroyed.
Objects of Class2 live and die with Class1.

Class2 cannot stand by itself.
A solid line with a filled diamond at the association connected to the class of composite

Example: Class Home and Class Room ,if Home got destructed Room won't exist.

Exercise 3-5

Describe the following class diagram



Assume that Address has extra attributes such as buildingNo, floorNo, and apartmentNo.
What is the difference between + and - sign in class diagram ?

Solution:

+ : represents public attributes or methods.
- : represents private attributes or methods.
 # : represents protected attributes or methods.

What are benefits of inheritance dependency ?

Solution:

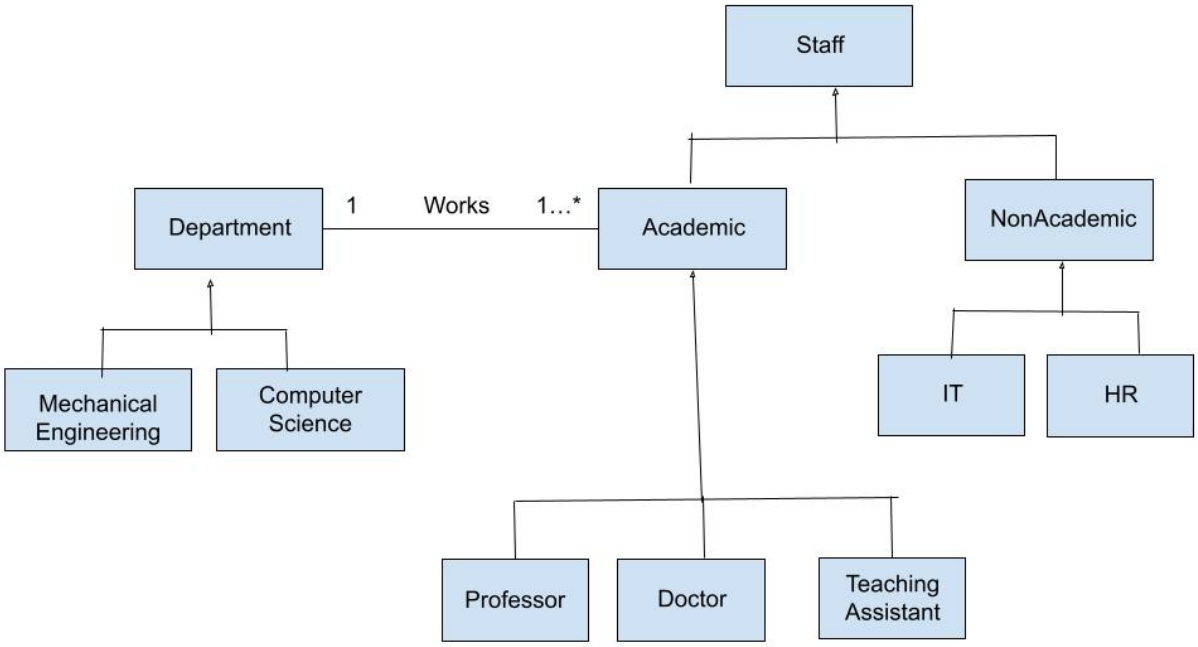
Inheritance helps in code reuse.
The child class may use the code defined in the parent class without re-writing it.
Inheritance can save time and effort as the main code need not be written again.
Inheritance provides a clear model structure which is easy to understand.
An inheritance leads to less development and maintenance costs.

Exercise 3-6

The GIU university mainly has academic staff and non academic staff. A Professor, Doctor, and Teaching Assistant are academic staff. Whereas HR and IT staff are non academic staff. Each department has at least one academic staff member and each academic staff member works only for one department. Also, GIU university has Computer Science Department and Mechanical Engineering Department.

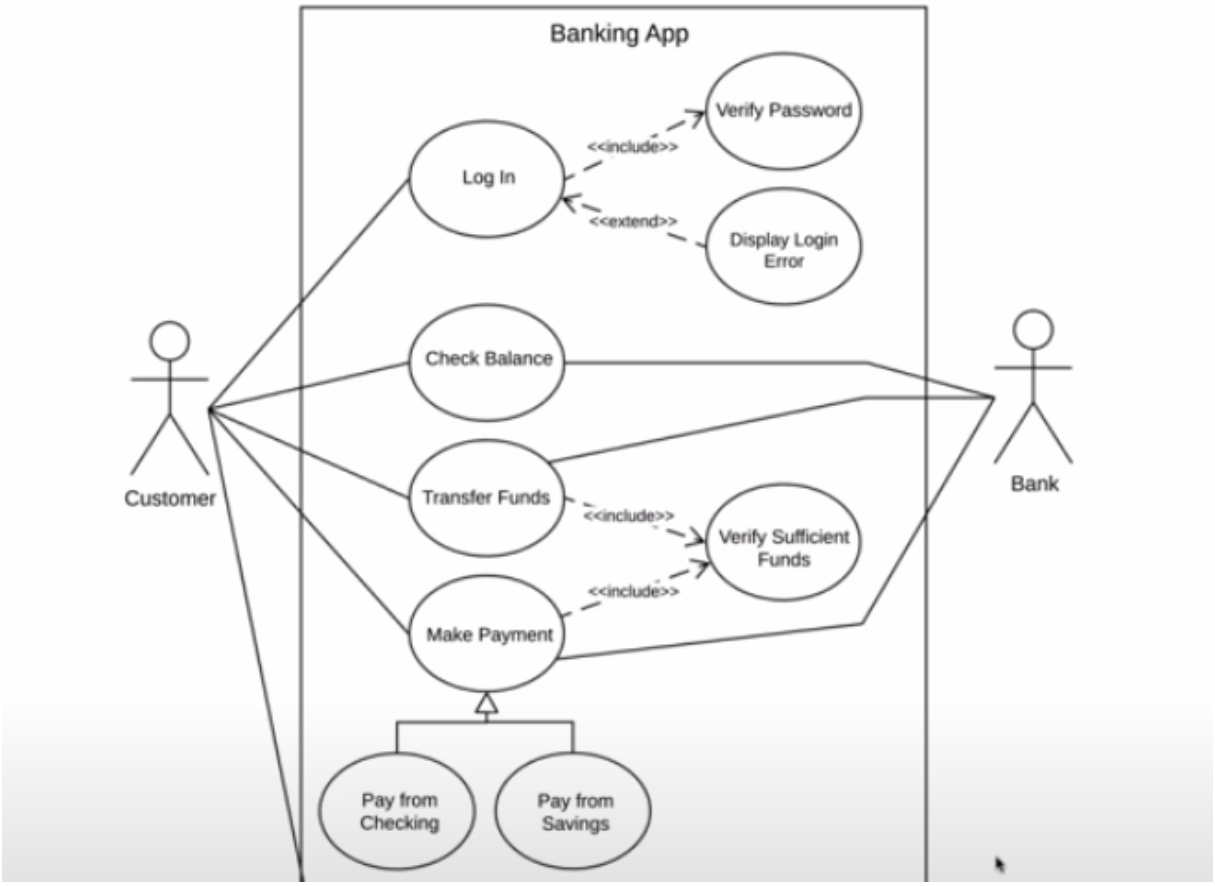
Design the following class diagram.
Don't add attributes and methods to each class.

Solution:



Exercise 3-7

Describe the following use case diagram



Solution:

The use case Diagram is mainly used to visualize actions which can be performed by the user . In this example, the customer (primary actor) can login, check Balance, transfer funds, and make payment through a banking app. The bank (secondary actor) provides information required for the customer to perform check balance, transfer funds and make payment. The make payment use case is the parent class of payment from checking and payment from savings. The verify password must be performed whenever the customer tries to login to the system. The display login error is shown whenever the user tries to enter wrong credentials. The verify sufficient funds must be performed whenever the customer tries to transfer funds and make payment.

Exercise 3-8

Draw a sequence diagram for an ATM withdrawal transaction use case.
The scenario is as the following:

- The user will insert the card into the ATM, The ATM will verify the card through the Bank server, and in case of a verified card the ATM will ask the user to enter the PIN code otherwise the card will be ejected.
- When the user replies to the ATM with the PIN code, the ATM will verify the PIN through the Bank server, and in case of a verified PIN, the ATM will ask the user to enter the required amount otherwise the card will be ejected.
- When the user replies with the amount, the ATM will ask the bank server to start the transaction.

- The Bank server will check if there are sufficient funds in the bank account.
- In case of sufficient funds, the bank server will ask to withdraw the amount from the bank account and the bank account will reply with withdraw successful message to the bank server, then the bank server will reply to the ATM with a transaction successful message and the ATM will dispense the cash to the user.
- But in case of insufficient funds, the bank account will reply insufficient funds message to the bank server, then the bank server will reply to the ATM with a transaction unsuccessful message.
- Finally the ATM will eject card to the user.

Solution: