**Software Testing Assignment**

**Module–1(Fundamental)**

1. **What is SDLC?**

🡪 The Software Development Life Cycle (SDLC) is a structured process that enables the production of high-quality, low-cost software, in the shortest possible production time.

🡪 SDLC is a structure imposed on the development of a software product that defines the process for planning, implementation, testing, documentation, deployment, and ongoing maintenance.

1. **What is agile methodology?**

🡪 The Agile methodology is a way to manage a project by breaking it up into several phases. It involves constant collaboration with stakeholders and continuous improvement at every stage. Once the work begins, teams working simultaneously on various areas like process of planning, requirements analysis, design, coding, unit testing, and acceptance testing.

🡪 Agile Model is combination of iterative and incremental process.

🡪 Agile methods break the product into small incremental builds.

🡪 These builds are provided in iterations.

🡪 Agile model believes that every project needs to be handled differently and the existing methods need to be tailored to best suit the project requirements. In agile the tasks are divided into small time frames to deliver specific features for a release.

**Agile Manifesto Principles**

* Individuals and interactions
* Working software
* Customer collaboration
* Responding to change

1. **What is SRS?**

**🡪** A software requirements specification is document that describes what the software will do and how it will be expected to perform. It also describes the functionality the product needs to fulfill all stakeholders needs.

**Types of Requirements**

Requirements are categorized in several ways.

* Customer Requirements
* Functional Requirements
* Non-Functional Requirements

**Customer Requirements**

The customers are those that perform the eight primary functions of systems engineering, with special emphasis on the operator as the key customer.

**Functional Requirements**

Functional Requirements are very important system requirements in the system design process. These requirements are the technical specifications, system design parameters and guidelines, data manipulation, data processing, and calculation modules etc.… of the proposed system.

**Non-Functional Requirements**

Non-functional requirements are requirements that specify criteria that can be used to judge the operation of a system, rather than specific behaviors. Non-functional requirements are qualities or standards that the system under development must have or comply with, but which are not tasks that will be automated by the system.

Nonfunctional requirements can be divided into following categories:

∙ Usability

∙ Reliability

∙ Performance

∙ Security

1. **What is oops?**

* Object-oriented programming (OOP) is a computer programming model that organizes software design around data, or objects, rather than functions and logic.

1. **Write Basic Concepts of oops**

**🡪**  Basic concepts of OOP:

1. Object
2. Class
3. Encapsulation
4. Inheritance
5. Abstraction
6. Polymorphism

∙ Overriding

∙ Overloading

1. **What is an object?**

🡪 Objects are the things you think about first in designing a program and they are also the units of code that are eventually derived from the process.

🡪 An object is an instance of a class. A class is a template or blueprint from which objects are created. So, an object is the instance of a class.

🡪 Everything in this world is an object:

A tree, a man, a flower, a city, the world, an animal, a bird, a vehicle, a student, etc.…

There are two parts of an Object.

Object = Data + Methods

1. **What is class?**

🡪 A class is a blueprint that defines the variables and the methods common to all objects of a certain kind.

🡪 A class represents an abstraction of the object and abstracts the properties and behavior of that object.

🡪 Ex. School is an object, but in school a teacher, a student, a classroom, a bench all are objects.

1. **What is encapsulation?**

**🡪** Encapsulation is a process of wrapping code and data together into a single unit.

🡪 Encapsulation enables data hiding, hiding irrelevant information from the users of a class and exposing only the relevant details required by the user.

🡪 Ex. a capsule which is mixed with several medicines. We can create a fully encapsulated class by making all the data members of the class private.

1. **What is inheritance?**

**🡪** Inheritance is the process by which genetic information is passed on from parent to child.











🡪 A subclass is also the type of its superclass.

🡪Inheritance is the process of creating a new Class, called the Derived Class, from the existing class, called the Base Class.

🡪 When a class derives from another class. The child class will inherit all the public and protected properties and methods from the parent class. In addition, it can have its own properties and methods. An inherited class is defined by using the extends keyword.

1. **What is polymorphism?**

🡪 Polymorphism means many forms of single thing.

🡪 The ability to change form is known as Polymorphism.

🡪 Polymorphism is allowing a specific routine to use variables of different types at different times.

🡪 Polymorphism is the ability of a programming language to present the same interface for several different underlying data types.

There are two types of polymorphism.

1) Compile time polymorphism (Overloading)

2) Runtime polymorphism (Overriding)

1. **Compile time polymorphism (Overloading)**

🡪 Static Binding

🡪 Name same behavior different place same it’s called overloading.

🡪 The concept of overloading is also a branch of polymorphism. When the exiting operator or function is made to operate on new data type, it is said to be overloaded.

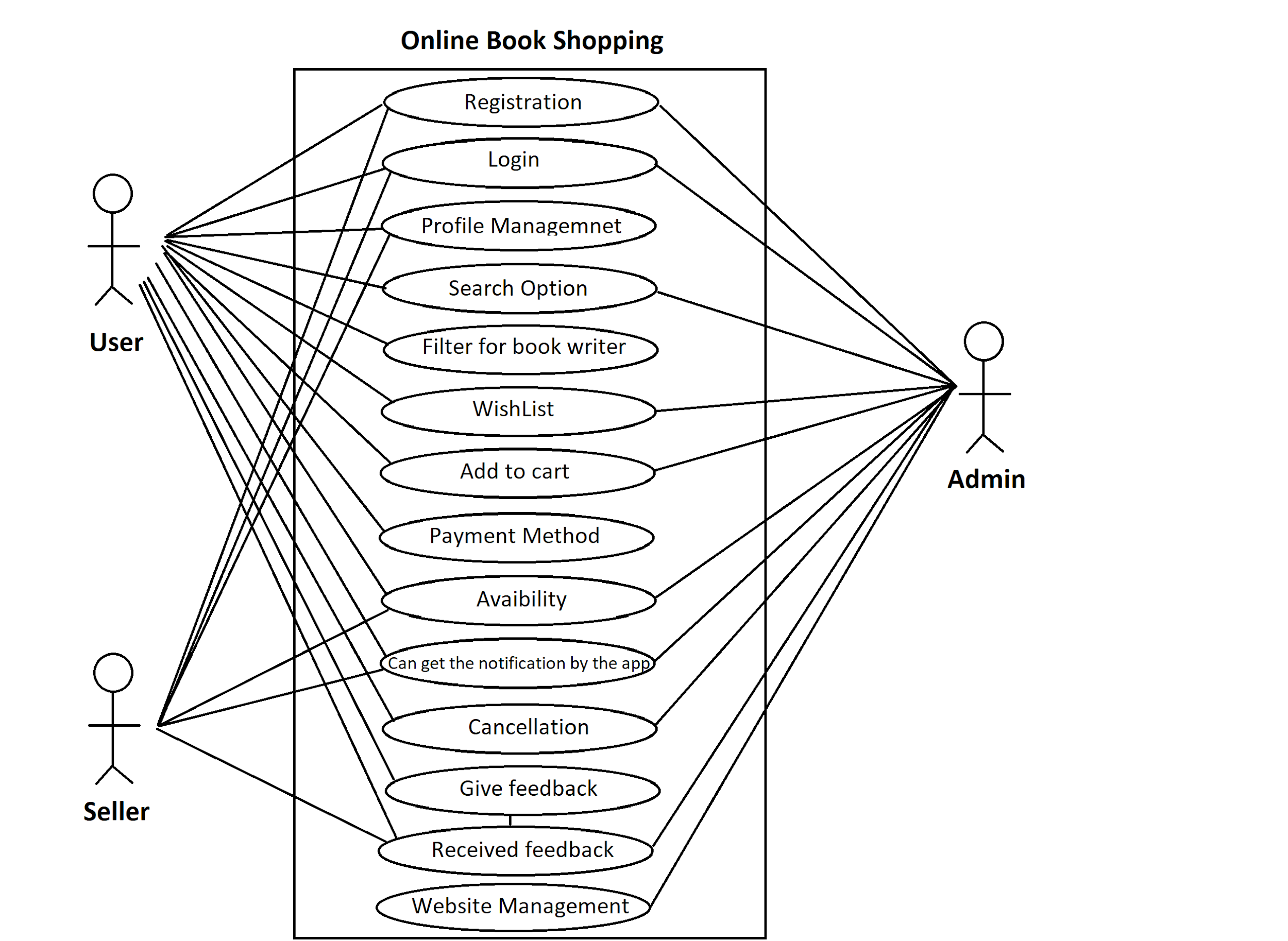
1. **Runtime polymorphism (Overriding)**

🡪 Dynamic Binding

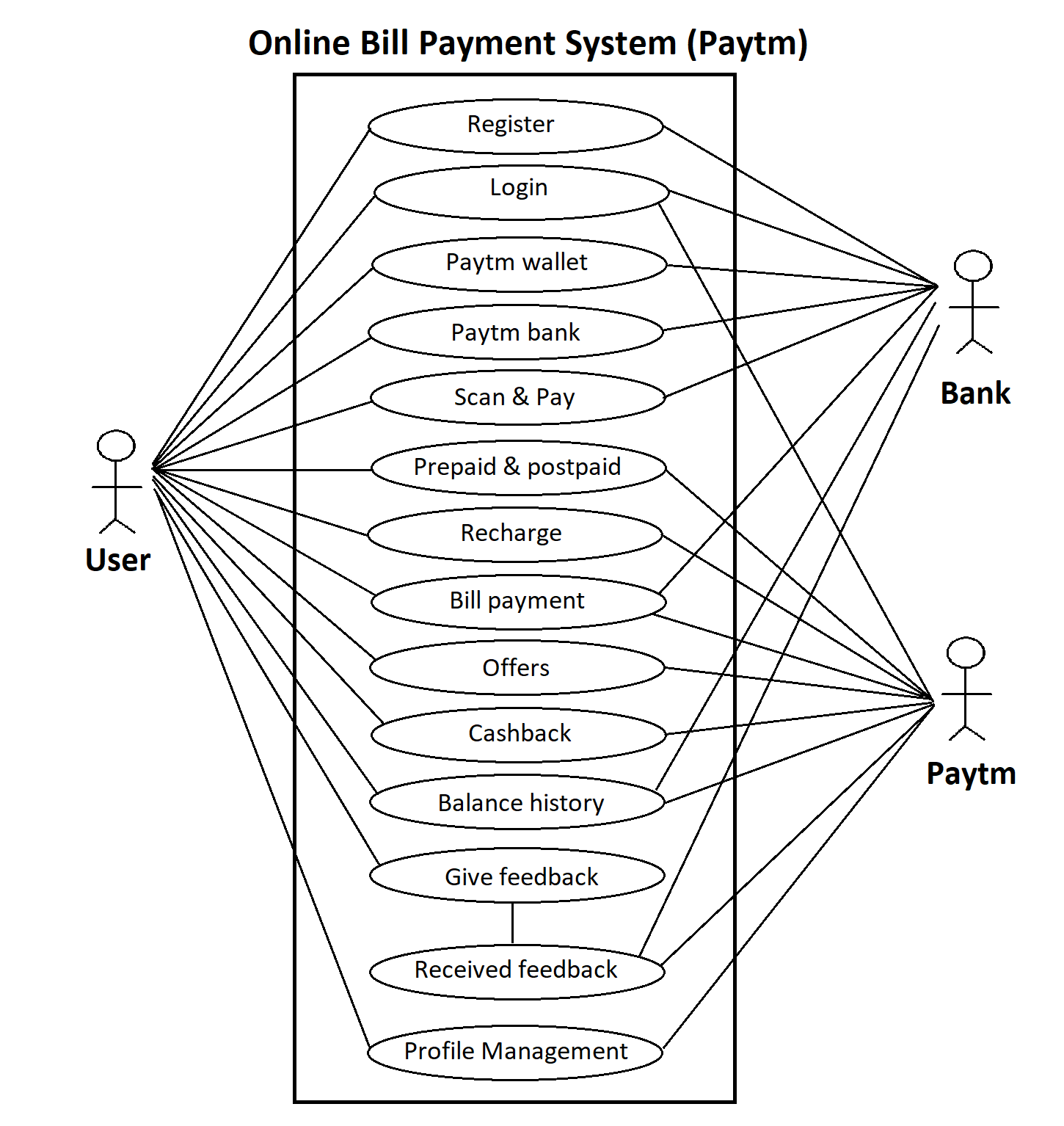
🡪 Name same behavior same place different it’s called overriding.

🡪 Overriding is defining a method in a subclass with the same name and type signature as a method in its super class and when this subclass instance appears in the super class context.

1. **What is RDBMS?**
2. **What is SQL?**
3. **Write SQL Commands**
4. **Draw Use case on Online book shopping**



1. **Draw Use case on online bill payment system (Paytm)**



1. **Write SDLC phases with basic introduction**

**🡪 SDLC Phases**

Analysis: In this phase analyst interact with stakeholders to develop the model and specify the requirements documents.

Design: During this phase developer and technical designer create initial level design for the software and system.

Implementation: Developers construct a solution in software.

Testing: Tester test the software to validate the solution against the requirements to make sure that the software is solving the needs during the planning

Maintenance: After software is ready or on live environment the system is in maintenance mode and repair defects and adapt the solution to the new requirements.

1. **Explain Phases of the waterfall model**

🡪 The Waterfall model is the earliest SDLC approach that was used for software development.

🡪 The waterfall model is a classical model used in system development life cycle to create a system with a linear and sequential approach. It is termed as waterfall because the model develops systematically from one phase to another in a downward fashion.

🡪 "The Waterfall" approach, the whole process of software development is divided into separate phases.

🡪 In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially.

🡪 There are five-stage of waterfall model, which is based on the requirements.

**Analysis, Design, Implementation, Testing, and Maintenance**.





**When it is used?**

* Requirements are very well documented, clear and fixed.
* Product definition is stable.
* Technology is understood and is not dynamic.
* There are no ambiguous requirements.
* Ample resources with required expertise are available to support the product.
* The project is short.

**Advantages:**

* Simple structure through clearly defined project phases.
* Easy to manage due to the rigidity of the model.
* Each phase has specific deliverables and a review process.
* Phases are processed and completed one at a time.
* Works well for smaller projects where requirements are very well understood.
* Clearly defined stages.
* Good documentation of the development process thanks to clearly defined milestones.
* Easy to arrange tasks.
* Process and results are well documented.
* Costs and workload can be estimated at the start of the project.
* Projects structured according to the waterfall model can be easily mapped on the time axis.

**Disadvantages:**

* No working software is produced until late during the life cycle.
* High amounts of risk and uncertainty.
* Not a good model for complex or multi-shift projects can rarely be divided into clearly defined project phases.
* Poor model for long and ongoing projects.
* It is difficult to measure progress within stages.
* Cannot accommodate changing requirements.
* No working software is produced until late in the life cycle.
* Little scope for adjustments to the project process due to changing requirements.
* Final user is only integrated into the production process after programming.
* Errors are sometimes only recognized at the end of the development process.

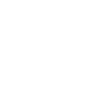
1. **Write phases of spiral model**

🡪 Spiral Model is a risk-driven software development process model.

🡪 It is a combination of waterfall model and iterative model. Spiral Model helps to adopt software development elements of multiple process models for the software project based on unique risk patterns ensuring efficient development process.

**🡪** The spiral model has four phases:

**Planning, Risk Analysis, Engineering and Customer Evaluation.**



1. **Planning:** In this phase, determine objectives and find alternate solutions. This phase includes requirement gathering and analysis. Based on the requirements, objectives are defined and different alternate solutions are proposed.
2. **Risk Analysis:** In this phase, all the proposed solutions are analyzed and any potential risk is identified, analyzed, and resolved.
3. **Engineering:** In this phase, includes the actual implementation of the different features. All the implemented features are then verified with thorough testing.
4. **Customer evaluation:** In this phase, Review and planning of the next phase.The software is evaluated by the customer. It also includes risk identification and monitoring like cost overrun or schedule slippage and after that planning of the next phase is started.
5. **Write agile manifesto principles**

🡪 Agile manifesto principles are:

* **Individuals and interactions** - in agile development, self-organization and motivation are important, as are interactions like co-location and pair programming.
* **Working software** - Demo working software is considered the best means of communication with the customer to understand their requirement, instead of just depending on documentation.
* **Customer collaboration** - As the requirements cannot be gathered completely in the beginning of the project due to various factors, continuous customer interaction is very important to get proper product requirements.
* **Responding to change** - agile development is focused on quick responses to change and continuous development.

1. **What is join?**
2. **Write type of joins.**
3. **Explain working methodology of agile model and also write pros and cons.**

**🡪** Agile methodology is a project management framework, used by teams to iteratively and incrementally complete tasks and projects. In most cases agile is implemented in the form of a working framework known as scrum, over short work beats called sprints.

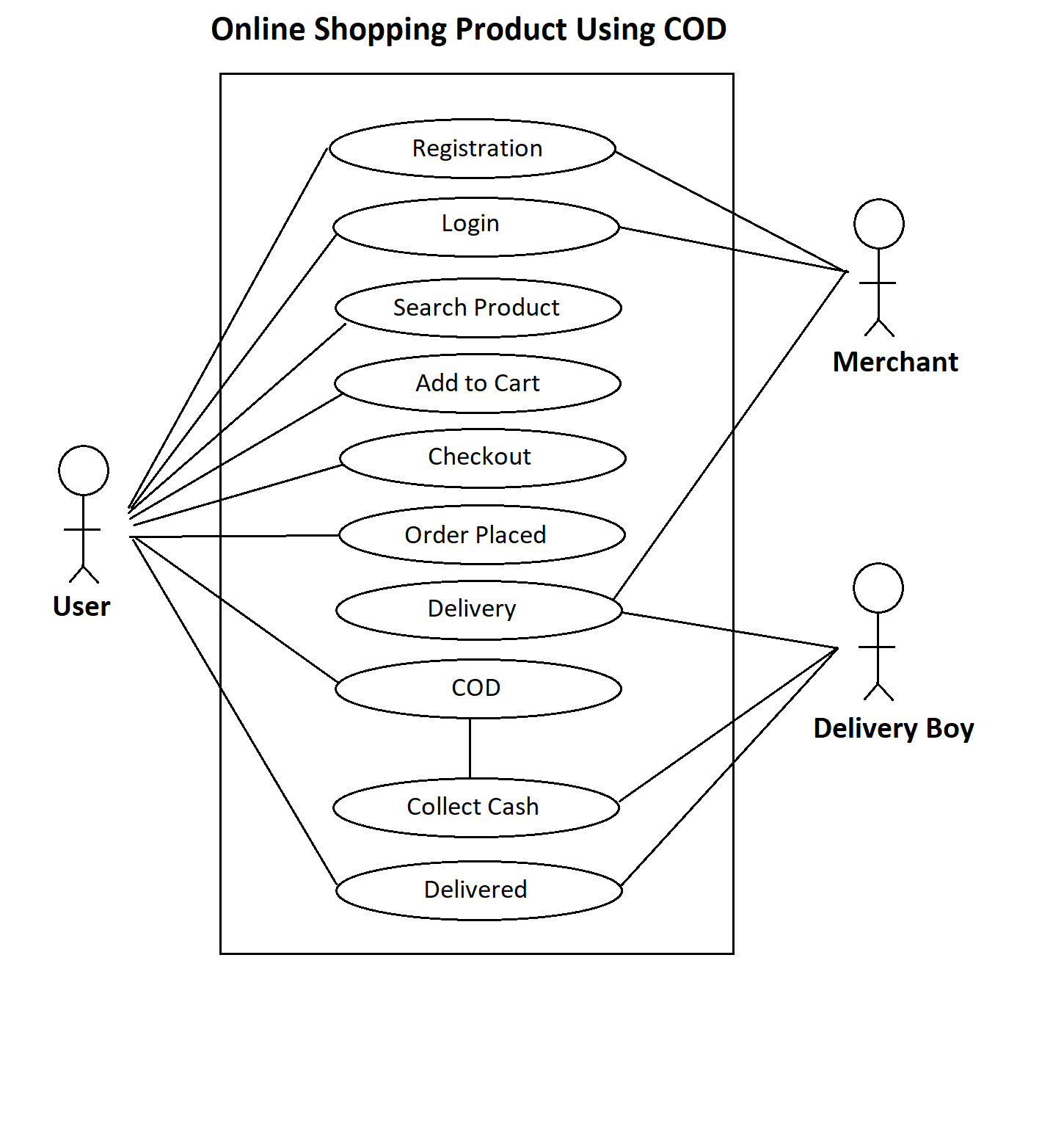
**Pros**

* Customer satisfaction is rapid, continuous development and delivery of useful software.
* Customer, Developer, and Product Owner interact regularly with each other.
* Product is developed fast and frequently delivered.
* A face-to-face conversation is the best form of communication.
* It continuously gave attention to technical excellence and good design.
* Regular adaptation to changing circumstances.
* Even late changes in requirements are welcomed.

**Cons**

* It is not useful for small development projects.
* There is a lack of intensity on necessary designing and documentation.
* It requires an expert project member to take crucial decisions in the meeting.
* Cost of Agile development methodology is slightly more as compared to other development methodology.
* The project can quickly go out off track if the project manager is not clear about requirements and what outcome they want.
* Not suitable for handling complex dependencies.

1. **Draw use case on Online shopping product using COD.**



1. **Draw use case on Online shopping product using payment gateway**

