[](https://topsint.com/careercenter/dashboard.php)

**Module: 1**

**Assignment: 1**

1. **What is SDLC?**

**Answer:**

SDLC is a structure imposed on the development of a software product that defines process for planning, implementation, testing, documentation, deployment and ongoing maintenance a support. There are a number of different development models.

1. **What is agile methodology?**

**Answer:**

Agile SDLC model is a combination of iterative and incremental process models with focus on process adaptability and customer satisfaction by rapid delivery of working software product.

-Agile methods breaks the product into small incremental builds. This build provided in iterations. Iteration typically lasts from about one to three weeks.

-Every iteration involves cross functional teams working simultaneously on various areas like planning, requirements analysis, design, coding, unit testing, and acceptance testing.

-At the end of the iteration a working product is displayed to the customer and important stakeholders.

-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 to time boxes to deliver specific features for a release.

-Iterative approach is taken and working software build is delivered after each iteration. Each build is incremental in terms of features; the final build holds all the features required by the customer. • Agile thought process had started early in the software development and started becoming popular with time due to its flexibility and adaptability.

1. **What is SRS?**

**Answer:**

-A software requirements specification is a complete description of the behaviour of the system to be developed.

-It includes a set of use cases that describe all of the interactions that the users will have with the software.

-Use cases are also known as functional requirements. In addition to use cases, the SRS also contains non-functional requirements.

-This standard describes possible structures, desirable contents, and qualities of a software requirements specification.

1. **What is OOPS?**

**Answer:**

**OOP-** Object Oriented Programming.

-Identifying objects and assigning responsibilities to these objects. An object is like a black box. An object has the responsibility to know and the responsibility to do.

1. **Write basic concept of OOPS?**

**Answer:**

**Concepts of OO:**

• **Object**

-An object represents an individual, identifiable item, unit, or entity, either real or abstract, with a well-defined role in the problem domain.

• **Class**

**-**A class represents an abstraction of the object and abstracts the properties and behavior of that object. Class can be considered as the blueprint or definition or a template for an object and describes the properties and behaviour of that object, but without any actual existence.

• **Encapsulation**

**-**Encapsulation is the practice of including in an object everything it needs hidden from other objects. The internal state is usually not accessible by other objects.

-Encapsulation is placing the data and the functions that work on that data in the same place. While working with procedural languages, it is not always clear which functions work on which variables but object-oriented programming provides you framework to place the data and the relevant functions together in the same object

**• Inheritance**

-Inheritance means that one class inherits the characteristics of another class. This is also called a “is a” relationship. This is a very important concept of object-oriented programming since this feature helps to reduce the code size.

• **Polymorphism**

-Polymorphism means “having many forms”.

-It allows different objects to respond to the same message in different ways, the response specific to the type of the object.

-**Overriding**

-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

-**Overloading**

**-**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 like

**• Abstraction:**

**-**Abstraction is the representation of the essential features of an object. These are ‘encapsulated’ into an abstract data type.

-Abstraction in Object Oriented Programming refers to the ability to make a class abstract.

1. **What is object?**

**Answer:**

-Tangible Things as a car, printer, Roles as employee, boss, Roles as employee, boss, Incidents as flight, overflow, Interactions as contract, sale, Specifications as colour, shape.

- An object represents an individual, identifiable item, unit, or entity, either real or abstract, with a well-defined role in the problem domain. An "object" is anything to which a concept applies.

-This is the basic unit of object-oriented programming (OOP). That is both data and function that operate on data are bundled as a unit called as object.

-Object = Data + Method

1. **What is class?**

**Answer:**

-When you define a class, you define a blueprint for an object. This doesn't actually define any data, but it does define what the class name means, that is, what an object of the class will consist of and what operations can be performed on such an object.

-A class represents an abstraction of the object and abstracts the properties and behaviour of that object.

-Class can be considered as the blueprint or definition or a template for an object and describes the properties and behaviour of that object, but without any actual existence.

1. **What is encapsulation**

**Answer:**

-Encapsulation is the practice of including in an object everything it needs hidden from other objects. The internal state is usually not accessible by other objects.

-Encapsulation is placing the data and the functions that work on that data in the same place. While working with procedural languages, it is not always clear which functions work on which variables but object-oriented programming provides you framework to place the data and the relevant functions together in the same object.

-Encapsulation in Java is the process of wrapping up of data and behaviour of an object into a single unit; and the unit here is a Class.

1. **What is inheritance**

**Answer:**

-Inheritance means that one class inherits the characteristics of another class. This is also called a “is a” relationship.

-One of the most useful aspects of object-oriented programming is code reusability. As the name suggests Inheritance is the process of forming a new class from an existing class that is from the existing class called as base class, new class is formed called as derived class.

-This is a very important concept of object-oriented programming since this feature helps to reduce the code size.

-Inheritance describes the relationship between two classes. A class can get some of its characteristics from a parent class and then add unique features of its own. In a class context, inheritance is referred to as implementation inheritance, and in an interface context, it is also referred to as interface inheritance.

1. **What is polymorphism**

**Answer:**

-Polymorphism means “having many forms”. It allows different objects to respond to the same message in different ways, the response specific to the type of the object.

-The ability to use an operator or function in different ways in other words giving different meaning or functions to the operators or functions is called polymorphism.

-Poly refers too many. That is a single function or an operator functioning in many ways different upon the usage is called polymorphism.

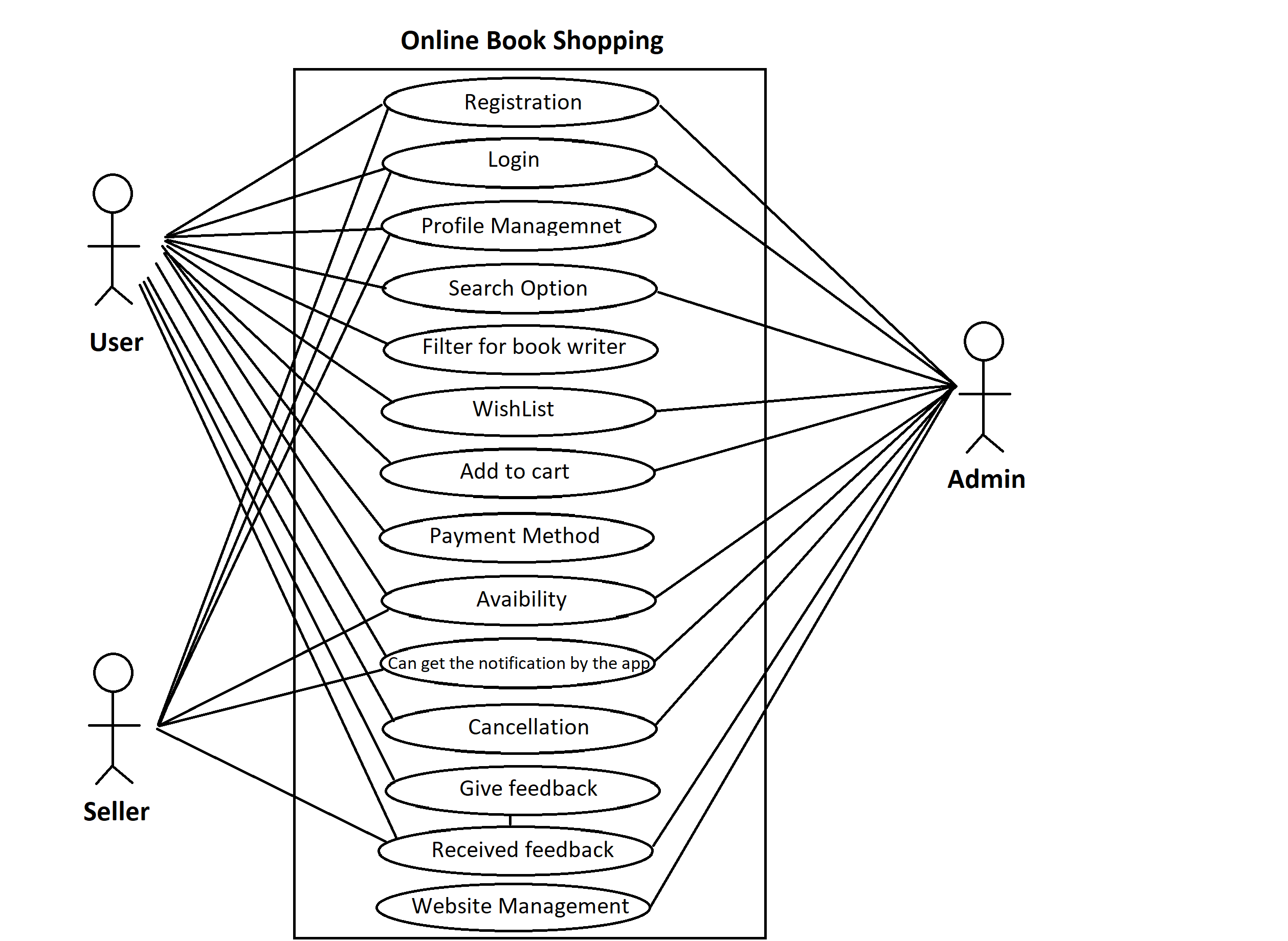
-There is two types of polymorphism in Java :

1. Compile time polymorphism (Overloading)

2. Runtime polymorphism (Overriding)

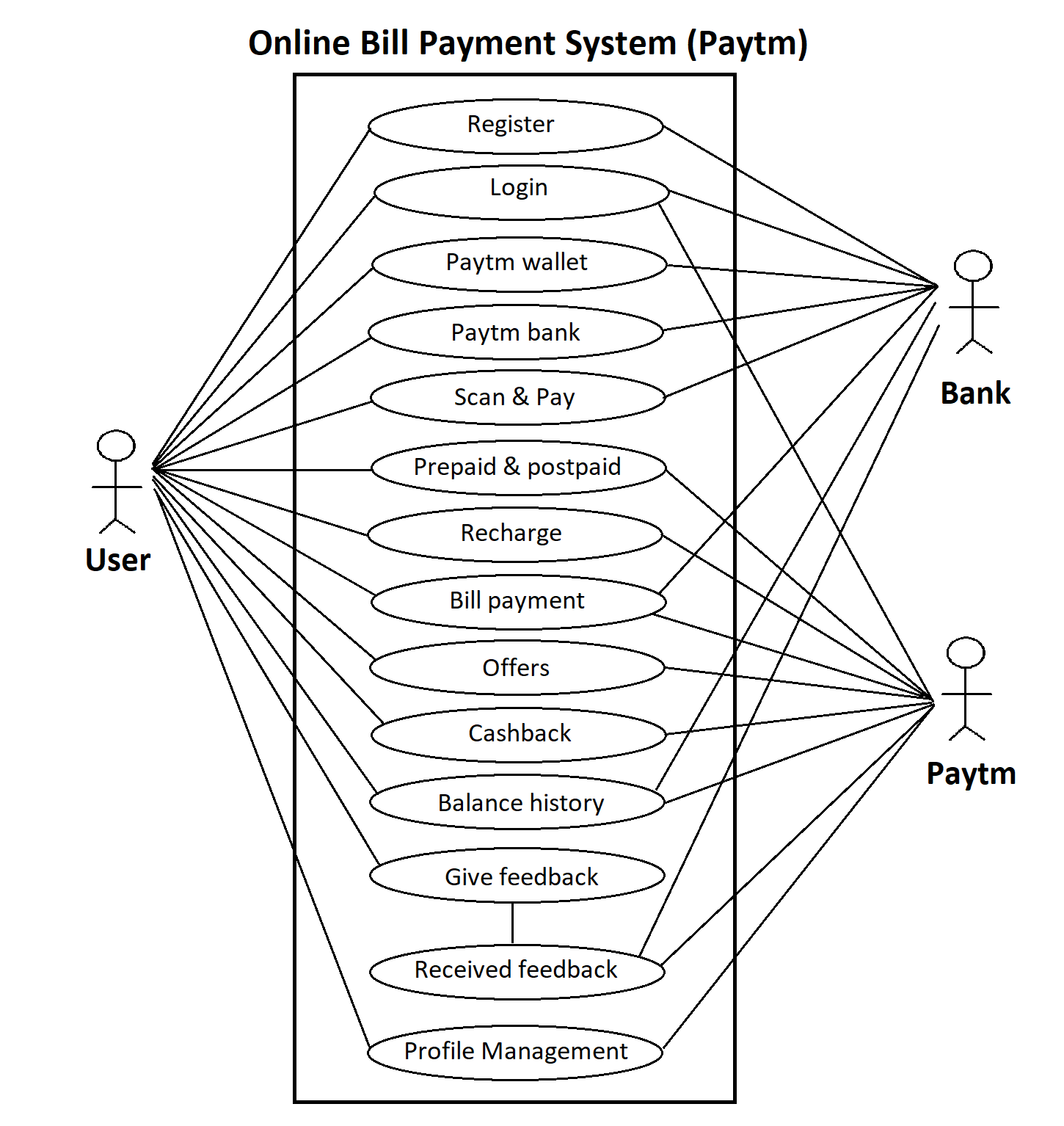
1. **Draw Use case on Online book shopping**

**Answer:**



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

**Answer:**



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

**Answer:**

SDLC: - Software Development Lifecycle Process.

**Analysis phase:** - The analysis phase defines the requirements of the system, independent of how these requirements will be accomplished.

-This phase defines the problem that the customer is trying to solve. The deliverable result at the end of this phase is a requirement document. Ideally, this document states in a clear and precise fashion what is to be built. Details on computer programming languages and environments, machines, packages, application architecture, distributed architecture layering, memory size, platform, algorithms, data structures, global type definitions, interfaces, and many other engineering details are established.

**Design phase:**

**-**Design Architecture Document

**-**Implementation Plan

-Critical Priority Analysis

-Performance Analysis

**-**Test Plan

-The Design team can now expand upon the information established in the requirement document.

-The requirement document must guide this decision process.

-Analyzing the trade-offs of necessary complexity allows for many things to remain simple which, in turn, will eventually lead to a higher quality product. The architecture team also converts the typical scenarios into a test plan.

**Implementation phase:**

**-**In the implementation phase the team builds the components either from scratch or by composition.

-Given the architecture document from the design phase and the requirement document from the analysis phase, the team should build exactly what has been requested, though there is still room for innovation and flexibility.

-For example, a component may be narrowly designed for this particular system, or the component may be made more general to satisfy a reusability guideline. The end deliverable is the product itself. There are already many established techniques associated with implementation.

**Testing phase:**

-Simply stated, quality is very important. Many companies have not learned that quality is important and deliver more claimed functionality but at a lower quality level.

-It is much easier to explain to a customer why there is a missing feature than to explain to a customer why the product lacks quality.

-A customer satisfied with the quality of a product will remain loyal and wait for new functionality in the next version.

-Quality is a distinguishing attribute of a system indicating the degree of excellence, Regression Testing, Internal Testing, Unit Testing, Application Testing, Stress Testing.

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

**Answer:**

The waterfall is unrealistic for many reasons, especially: (1) Requirements must be “frozen” to early in the life cycle. (2) Requirements are validated too late.

-It is used when requirement is well documented, clear and fixed. Product definition is stable. Technology is understood and is not dynamic. There is no ambitious requirements. Ample resources with required expertise are available to support the product. The project is short.

-Phases Of Waterfall Model:

Requirement collection Analysis Design Implementation Testing

-In Practice, development is always iterative, and all activates progress in parallel.

1. **Write phases of spiral model**

**Answer:**

-Spiral Model is very widely used in the software industry as it is in synch with the natural development process of any product i.e. learning with maturity and also involves minimum risk.

-When costs there are a budget constraint and risk evaluation is important.

-For medium to high-risk projects. Long-term project commitment because of potential changes to economic priorities as the requirements change with time.

-Customer is not sure of their requirements which are usually the case.

-Requirements are complex and need evaluation to get clarity.

-New product line which should be released in phases to get enough customer feedback.

-Significant changes are expected in the product during the development cycle.

**Pros:**

**-**Changing requirements can be accommodated.

-Allows for extensive use of prototypes

-Requirements can be captured more accurately.

-Users see the system early.

-Development can be divided into smaller parts and more risky parts can be developed earlier which helps better risk management.

**Cons:**

-Management is more complex.

- End of project may not be known early.

-Not suitable for small or low risk projects and could be expensive for small projects.

-Process is complex.

-Spiral may go indefinitely.

-Large number of intermediate stages requires excessive documentation.

1. **Write agile manifesto principles.**

**Answer:**

-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. **Explain working methodology of agile model and also write pros and cons.**

**Answer:**

**Agile Model:**

Iteration

Warm up

construction

Release end games

production

Retirement

-Agile SDLC model is a combination of iterative and incremental process models with focus on process adaptability and customer satisfaction by rapid delivery of working software product.

-Agile Methods break the product into small incremental builds. These builds are provided in iterations. Each iteration typically lasts from about one to three weeks.

-Every iteration involves cross functional teams working simultaneously on various areas like planning, requirements analysis, design, coding, unit testing, and acceptance testing.

-At the end of the iteration a working product is displayed to the customer and important stakeholders.

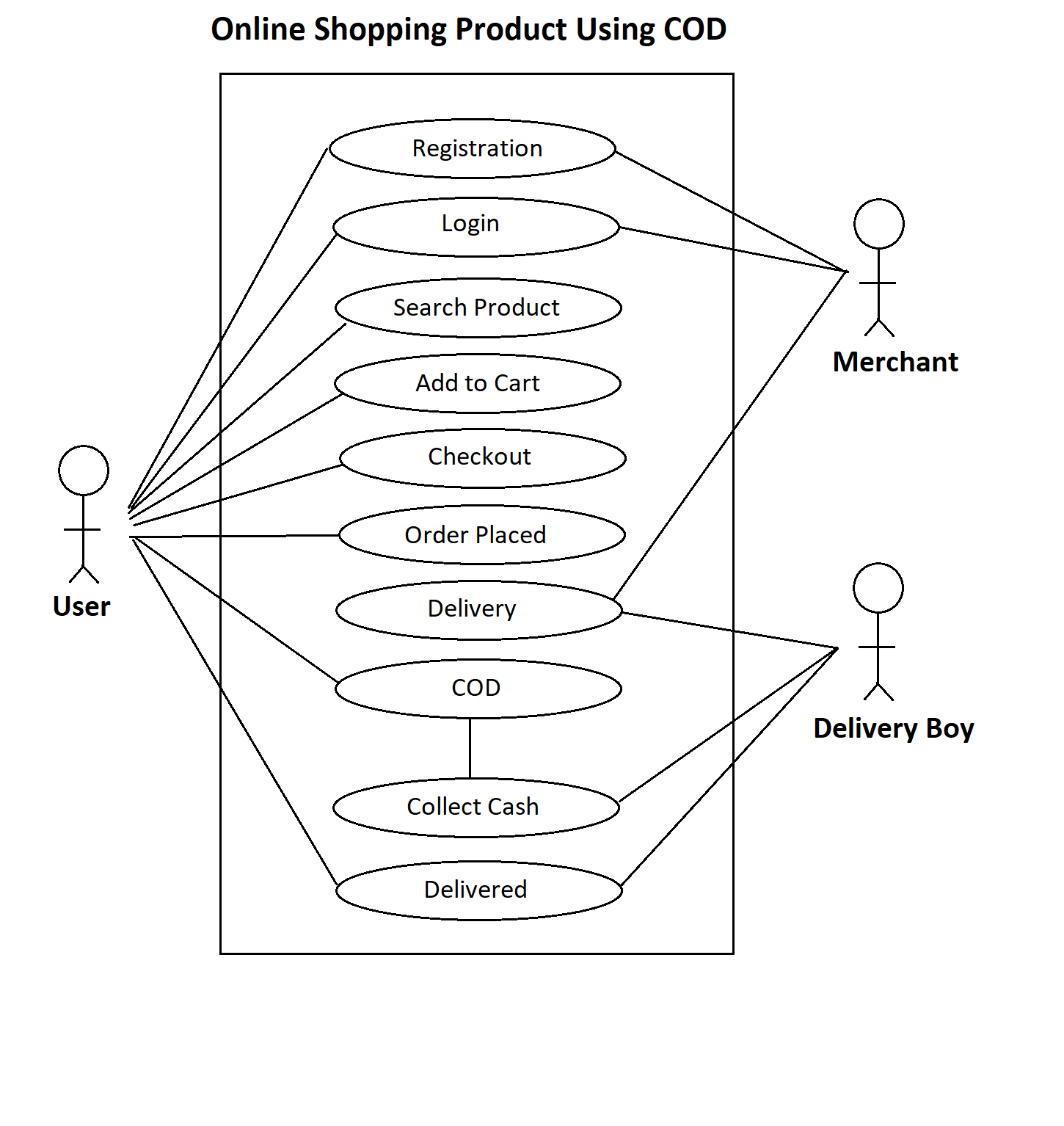
-Agile model believes that every project needs to be handled differently and the existing methods need to be tailored to best suit the project requirements. In agile the tasks are divided to time boxes to deliver specific features for a release.

-Iterative approach is taken and working software build is delivered after each iteration. Each build is incremental in terms of features; the final build holds all the features required by the customer.

-Agile thought process had started early in the software development and started becoming popular with time due to its flexibility and adaptability.

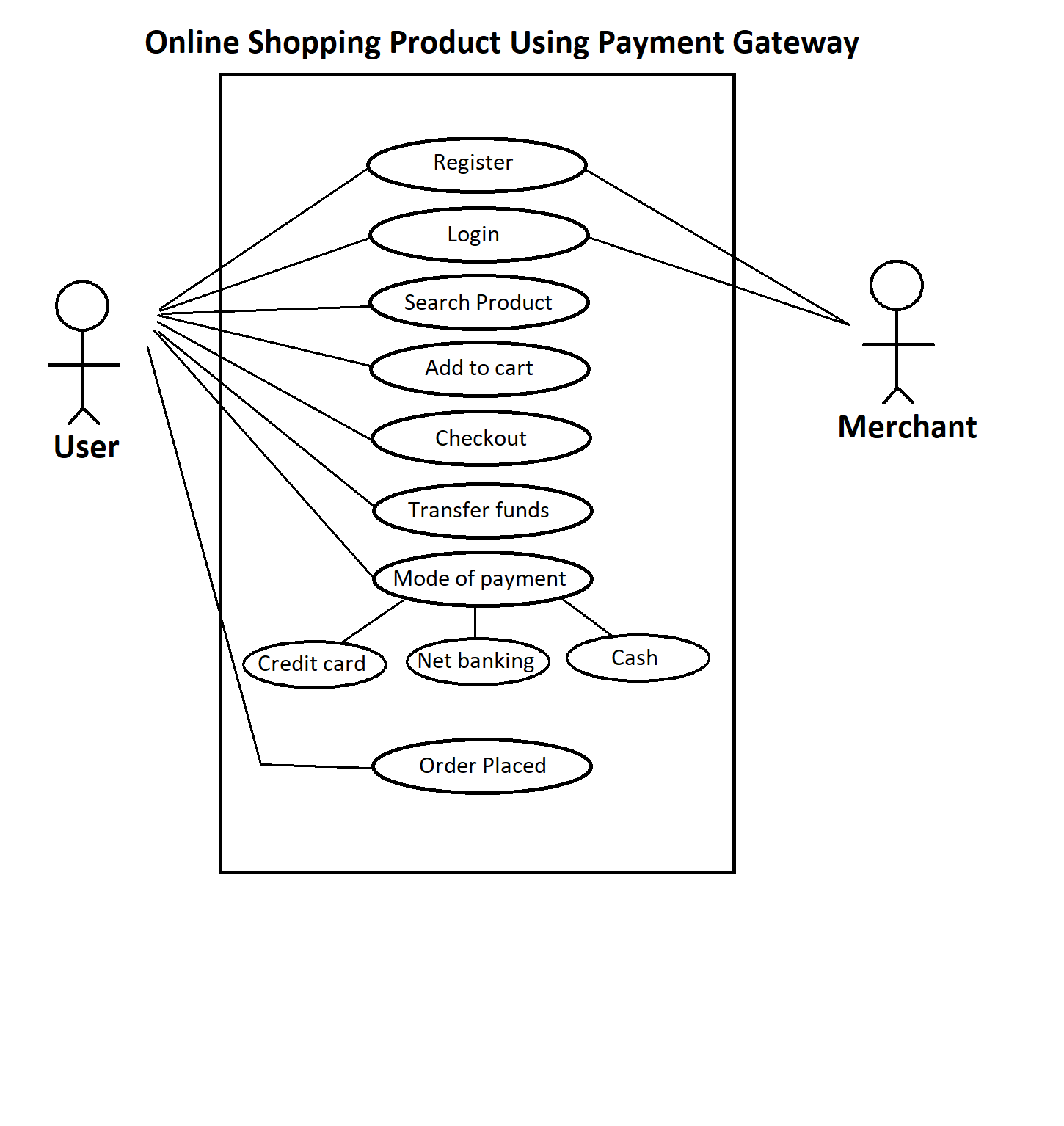
1. **Draw Use case on Online shopping product using COD.**

**Answer:**



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

**Answer:**



­­

**Remain­­­­ing Question: For Next module**

1. What is RDBMS
2. What is SQL
3. Write SQL Commands
4. What is join?
5. Write type of joins.