Q.1 What is SDLC ?

A.1 Software Development Life Cycle (SDLC)

- 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 and support.

There are a number of different development models.

- A Software Development Life Cycle is essentially a series of steps, or

phases, that provide a model for the development and lifecycle management of an application or piece of software.

- The methodology within the SDLC process can vary across industries and organizations, but standards such as ISO/IEC 12207 represent processes that establish a lifecycle for software, and provide a mode for the development, acquisition, and configuration of software systems.

Q.2 What is software testing?

A.2 Software testing is a process of executing a program or application with

the intent of finding the software bugs.

● Software Testing is a process used to identify the correctness,

completeness, and quality of developed computer software.

● When asked, people often think that Testing only consists of running

tests, i.e. executing the software

● Test execution is only a part of testing, but not all of the testing

activities

● Test activities exist before and after test execution

● It can also be stated as the process of validating and verifying that a

software program or application or product:

● Meets the business and technical requirements that guided it's design and

development

● Works as expected

● Can be implemented with the same characteristic

Q.3 What is agile methodology?

A.3 Agile Model/Methodology

● 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 (small time frames) 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.

Pros

● Is a very realistic approach to software development

● Promotes teamwork and cross training.

● Functionality can be developed rapidly and demonstrated.

● Resource requirements are minimum.

● Suitable for fixed or changing requirements

● Delivers early partial working solutions.

● Good model for environments that change steadily.

● Minimal rules, documentation easily employed.

● Enables concurrent development and delivery within an overall planned context.

● Little or no planning required

● Easy to manage

● Gives flexibility to developers

Cons

● Not suitable for handling complex dependencies.

● More risk of sustainability, maintainability and extensibility.

● An overall plan, an agile leader and agile PM practice is a must without which it will not work.

● Strict delivery management dictates the scope, functionality to be delivered, and adjustments to meet the deadlines.

● Depends heavily on customer interaction, so if customer is not clear, team can be driven in the wrong direction.

● There is very high individual dependency, since there is minimum documentation generated.

● Transfer of technology to new team members may be quite challenging due to lack of documentation.

Q.4 What is SRS ?

A.4 Software Requirement Specification

● A software requirements specification (SRS) is a complete description

of the behavior 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 nonfunctional (or supplementary)

requirements.

● Non-functional requirements are requirements which impose constraints on the design or implementation (such as performance requirements, quality standards, or design constraints).

● Recommended approaches for the specification of software

requirements are described by IEEE 830-1998.

● This standard describes possible structures, desirable contents, and

qualities of a software requirements specification.

Q.5 What is oops ?

A.5 Object Oriented Programming System

• Object means a real word entity such as pen, chair, table etc.

• Object-Oriented Programming is a methodology or paradigm to

design a program using classes and objects.

Q.6 Write Basic Concepts of oops.

A.6 It simplifies the software development and maintenance by providing

some concepts:

- Object
- Class
- Inheritance
- Polymorphism
- Abstraction
- Encapsulation

Q.7 What is object?

A.7 Any entity that has state and behavior is known as an object. For example: chair, pen,table, keyboard, bike etc. It can be physical and logical.

Q.8 What is class?

A.8 Collection of objects is called class. It is a logical entity.

Q.9 What is encapsulation?

A.9 Binding (or wrapping) code and data together into a single unit is known as encapsulation.

For example: capsule, it is wrapped with different medicines.

A java class is the example of encapsulation. Java bean is the fully encapsulated class because all the data members are private here.

Q.10 What is inheritance?

A.10 Inheritance is a mechanism wherein a new class is derived from an existing class.

• In Java, classes may inherit or acquire the properties and methods of other classes.

• A class derived from another class is called a sub class, where as the class from which a subclass is derived is called a super class.

Q.11 What is polymorphism?

A.11 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 most important aspect of an object is its behaviour (the things it can do).

A behaviour is initiated by sending a message to the object (usually by calling a method).

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 to many. That is a single function or an operator functioning in many ways different upon the usage is called polymorphism.

E.g. the message displayDetails() of the Person class should give different results when send to a Student object (e.g. the enrolment number).

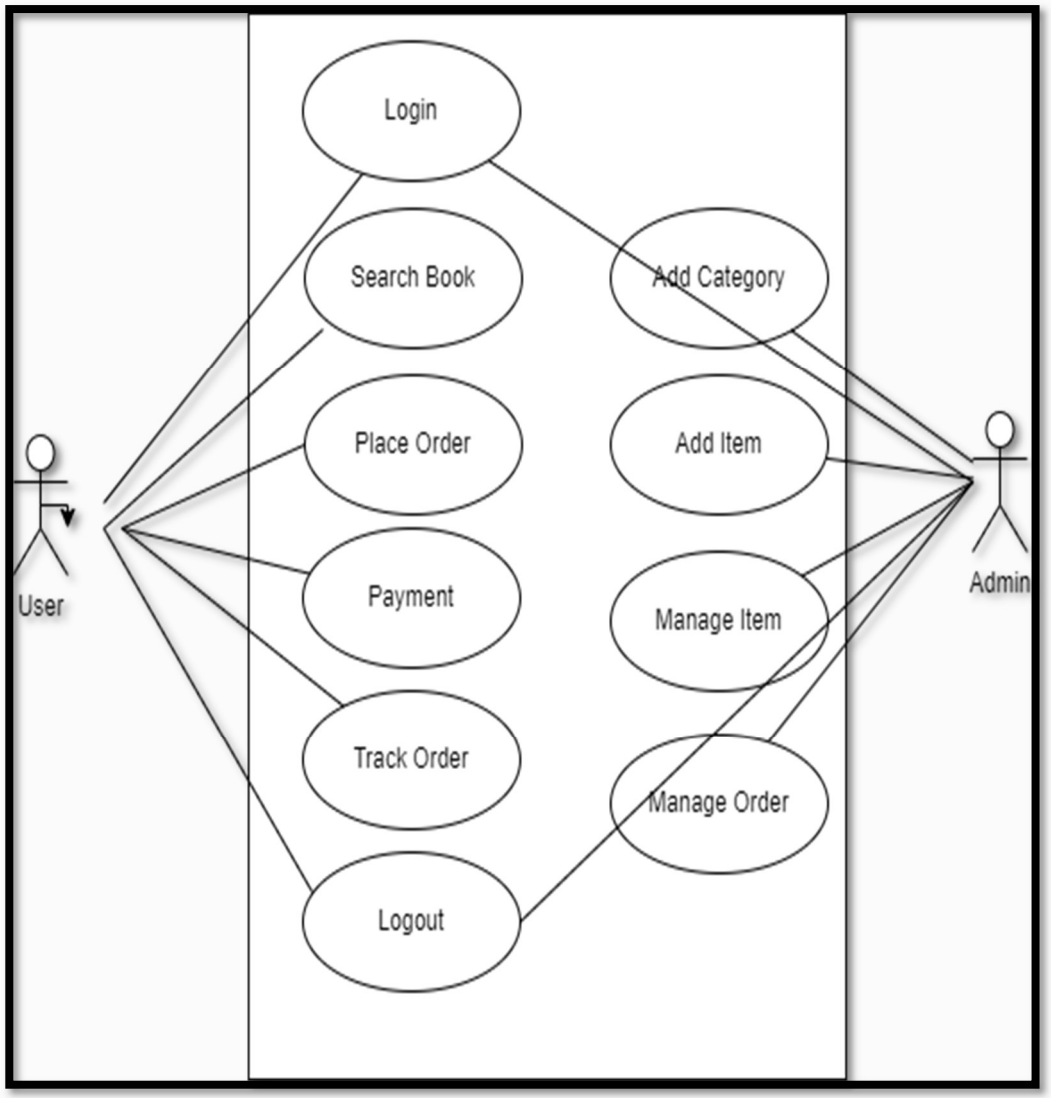
- **The ability to change form is known as polymorphism. There is two types of polymorphism in Java**

Compile time polymorphism(Overloading)

Runtime polymorphism(Overriding)

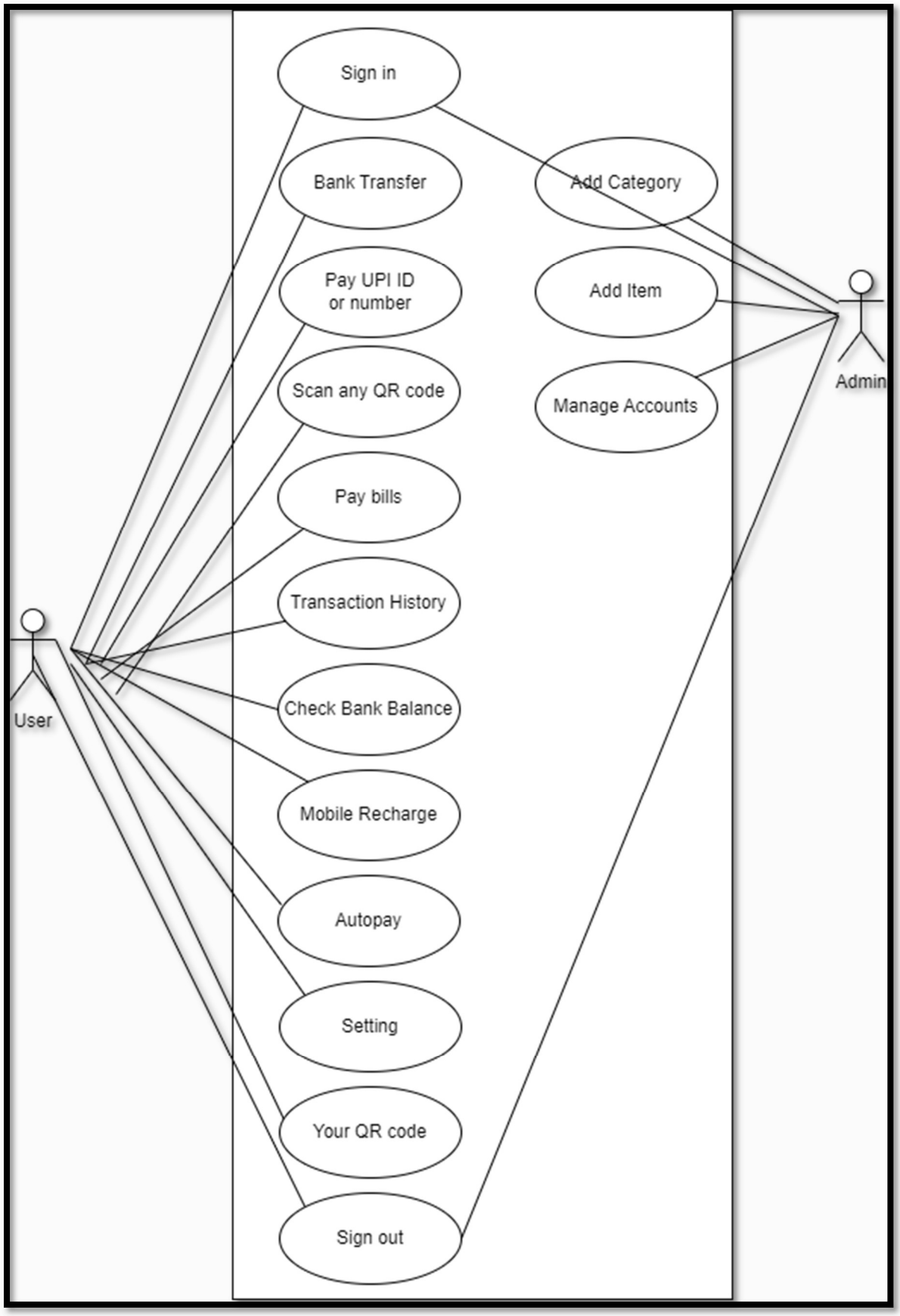
Q.12 Draw Usecase on Online book shopping.

A.12



Q.13 Draw Usecase on online bill payment system (paytm).

A.13


Sign in
Bank Transfer
Pay UPI ID
or number
Scan any QR code
Pay bills
Transaction History
Check Bank Balance
Mobile Recharge
Autopay
Setting
Your QR code
Sign out
Add Category
Add Item
Manage Accounts
Admin
User

Q. 14 Write SDLC phases with basic introduction.

A. 14 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 and support. There are a number of different development models.

- **A Software Development Life Cycle is essentially a series of steps, or phases, that provide a model for the development and lifecycle management of an application or piece of software.**
- **The methodology within the SDLC process can vary across industries and organizations, but standards such as ISO/IEC 12207 represent processes that establish a lifecycle for software, and provide a mode for the development, acquisition, and configuration of software systems.**

Phases of SDLC :

1.Requirements Collection/Gathering- Establish Customer Needs

2. Analysis- Model and Specify the requirements-"What"

3. Design- Model and Specify a Solution – "Why"

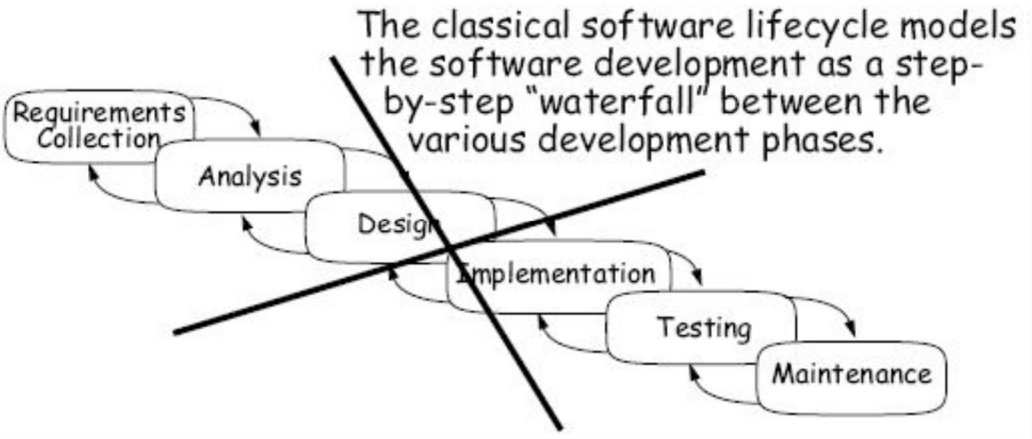
4. Implementation- Construct a Solution in Software

5. Testing- Validate the solution against the requirements

6. Maintenance- Repair defects and adapt the solution to the new requirements.

Q.15 Explain Phases of the waterfall model.

A.15



The waterfall is unrealistic for many reasons, especially:

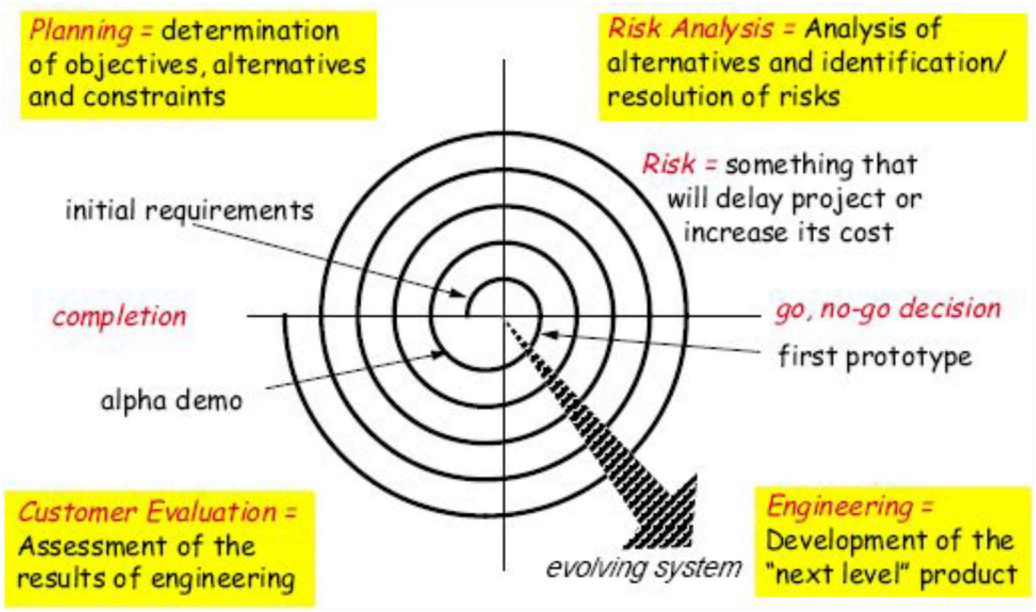
- **Requirements must be "frozen" to early in the life cycle**
- **Requirements are validated too late**

Applications (When to use?)

- **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.**

Q.16 Write phases of spiral model.

A.16



Q.17 Write agile manifesto principles.

A.17

- **Individuals and interactions over processes and tools.**
- **Working software over comprehensive documentation.**
- **Customer collaboration over contract negotiation.**
- **Responding to change over following a plan.**

Q.18 Explain working methodology of agile model and also write pros and cons.

A.18

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 (small time frames) 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.**

Pros

- **Is a very realistic approach to software development**

- **Promotes teamwork and cross training.**
- **Functionality can be developed rapidly and demonstrated.**
- **Resource requirements are minimum.**
- **Suitable for fixed or changing requirements**
- **Delivers early partial working solutions.**
- **Good model for environments that change steadily.**
- **Minimal rules, documentation easily employed.**
- **Enables concurrent development and delivery within an overall planned context.**
- **Little or no planning required**
- **Easy to manage**
- **Gives flexibility to developers**

Cons

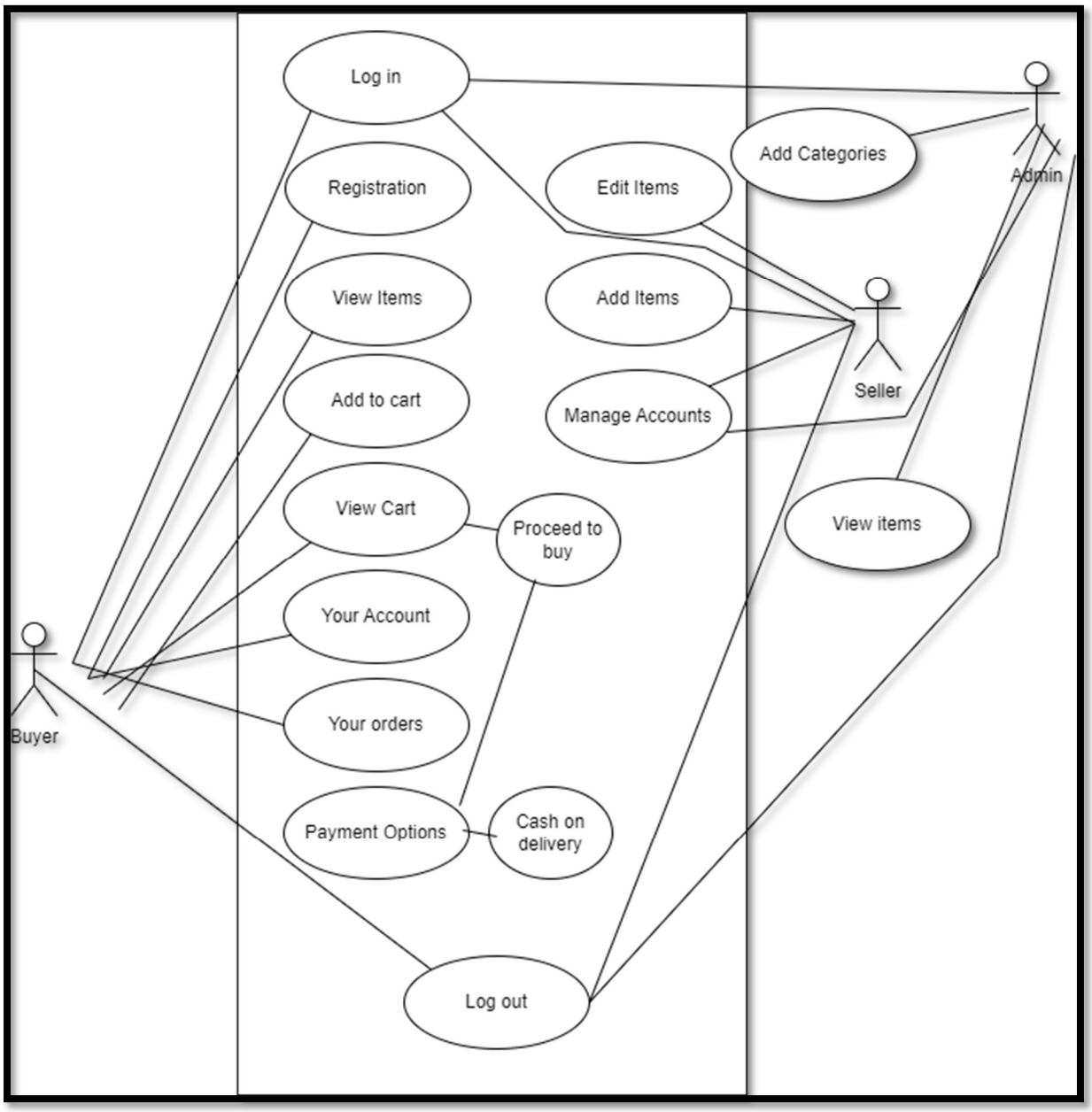
- **Not suitable for handling complex dependencies.**
- **More risk of sustainability, maintainability and extensibility.**
- **An overall plan, an agile leader and agile PM practice is a must without which it will not work.**
- **Strict delivery management dictates the scope, functionality to be delivered, and adjustments to meet the deadlines.**
- **Depends heavily on customer interaction, so if customer is not clear, team can be driven in the wrong direction.**
- **There is very high individual dependency, since there is minimum**

documentation generated.

● Transfer of technology to new team members may be quite challenging due to lack of documentation.

Q.19 Draw usecase on Online shopping product using COD.

A.19



Q. 20 Draw usecase on Online shopping product using payment gateway.

A.20