Software testing

Name: Patel janvi

Assessment:1

1 . what is 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 model
– 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.
– SDLC is talking about the complete software development from beginning to end like requirements, understanding, designing, coding, testing everything will be part of software development life cycle.

2. what is software testing?

- 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

3. What is agile methodology?

- Agile SDLC model is combination of iterative and incremental process model with focus on process.
- 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.

4. What is SRS ?

- software requirements specification (SRS) is a complete description of the behaviour of the system to be developed**.**
- Use cases are also known as functional requirements. In addition to use cases, the SRS also contains nonfunctional requirements
- This standard describes possible structures, desirable contents, and qualities of a software requirements specification.
- Requirements are categorised in several ways. The following are common categorizations of requirements that relate to technical management .

  1 Customer requirements

  2 function requirements

  3 non function requirements

5. What is oops?

- Identifying objects and assigning responsibilities to these objects.
- Objects communicate to other objects by sending messages.
- Messages are received by the methods of the project.
- An object is like a black box.
- The internal details are hidden.
- Object-oriented programming has a web of interacting
- objects, each house-keeping its own state.
- Objects of a program interact by sending messages to each other.

6. Write basic Concepts of OOPS :

⇒ Inheritance

⇒ Encapsulation

⇒ Polymorphism

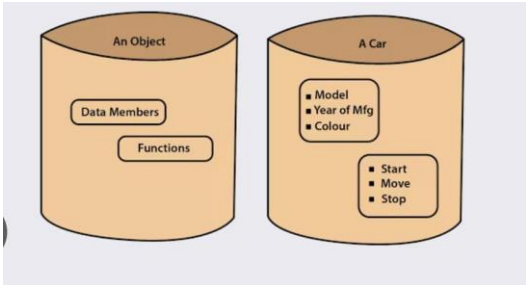
⇒ Data abstraction

6. what is object

- 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.
- Objects are the basic run time entities of an object oriented system.
- They may represent a person, a place or any item that the program must handle.
- Example



7. what is class

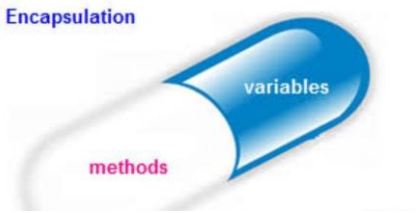
- When you define a class, you define a blueprint for an object.
- This doesn't actually define any data, but it does define what the class name means, that is, what an object of the class will consist of and what operations can be performed on such an object.
- A class represents an abstraction of the object and abstracts the properties and behaviour of that object
- An object is a particular instance of a class which has actual existence and
  there can be many objects for a class.

8. what is 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.

- Encapsulate in plain English means to enclose or be enclosed in or as if in a capsule. In Java, a class is the capsule .

Example: capsule



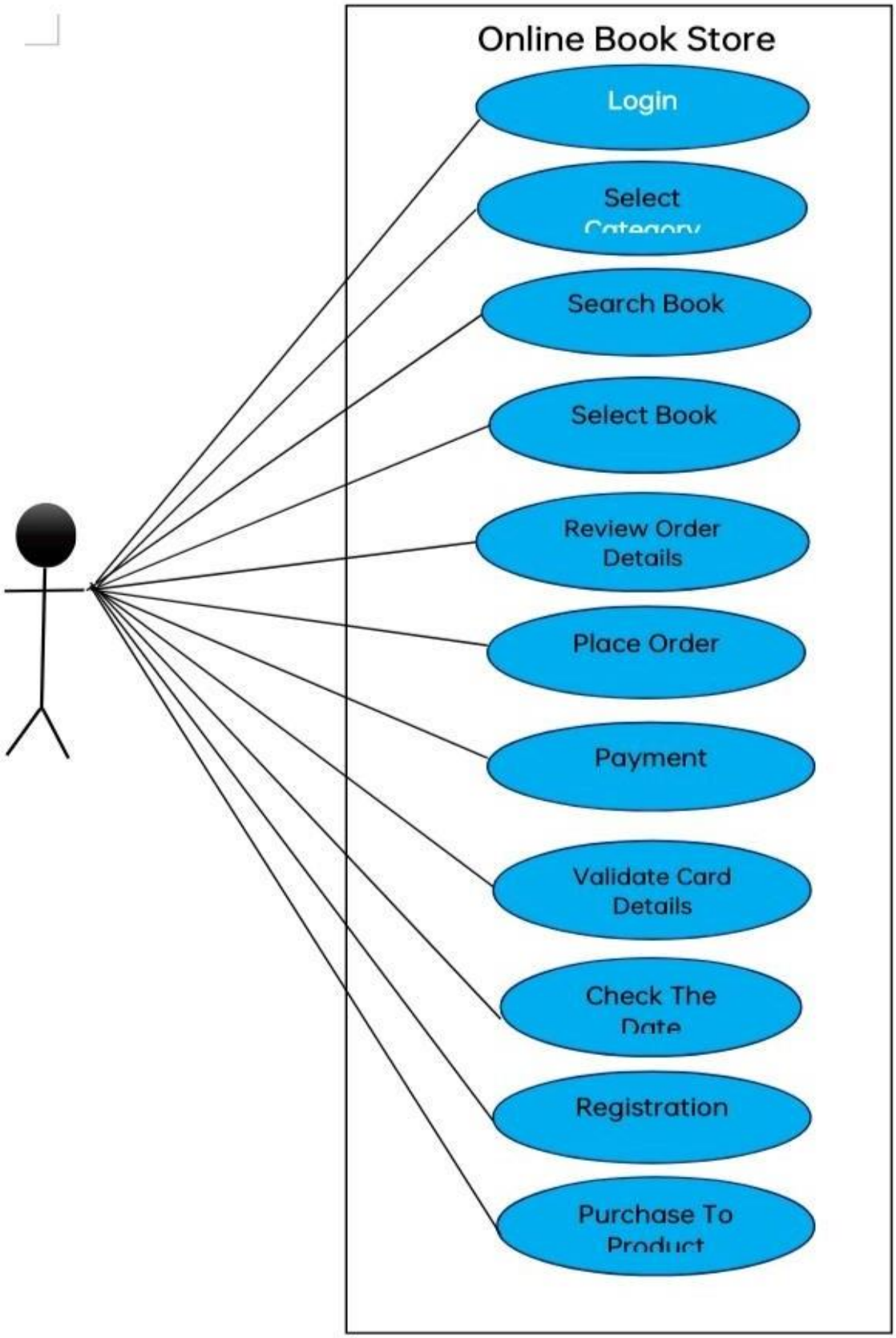
9. What is inheritance

- Inheritance means that one class inherits the characteristics ofanother 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.
- 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.
- Inheritance
- In general, Java supports single-parent, multiple-children inheritance
- and multilevel inheritance (Grandparent-> Parent -> Child) for classes and interfaces. Java supports multiple inheritances (multiple parents, single child) only through interfaces.
- In a class context, inheritance is referred to as implementation inheritance, and in an interface context, it is also referred to as interface inheritance.

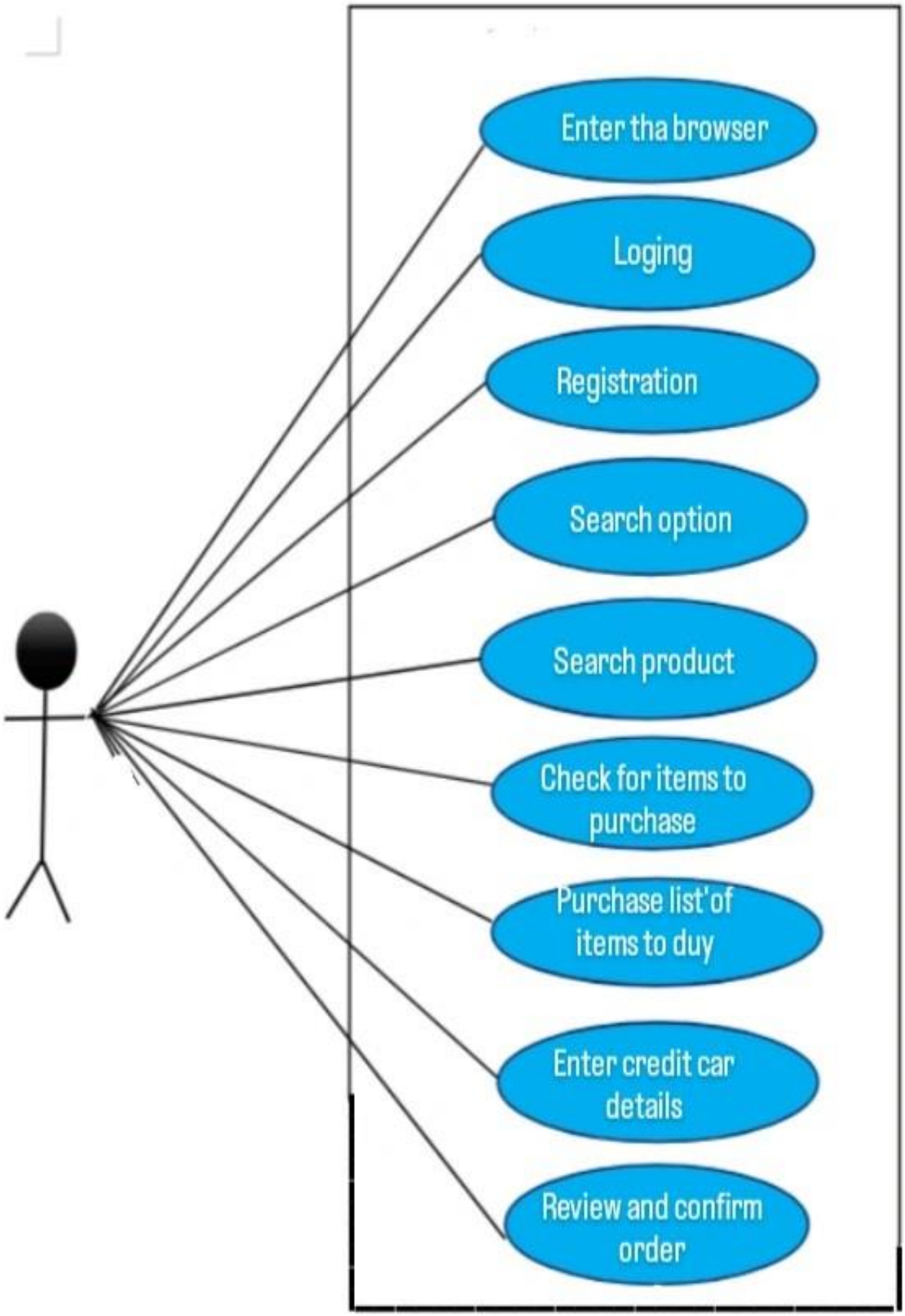
10. what is polymorphism

- 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.
- The most important aspect of an object is its behaviour
- A behavior is initiated by sending a message to the object.
- 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 display Details() of the Person class should give different results when send to a Student object .
- There is two types of polymorphism in Java .

- Draw Usecase on Online book Shopping



- Draw a Usecase on online bill payment system (Paytm)



11. Write SDLC phases with basic introductions

SDLC Phases

1 Requirements

2 Collection/Gathering

3 Analysis

4 Design

5 Implementation

6 Testing

7 Maintenanc

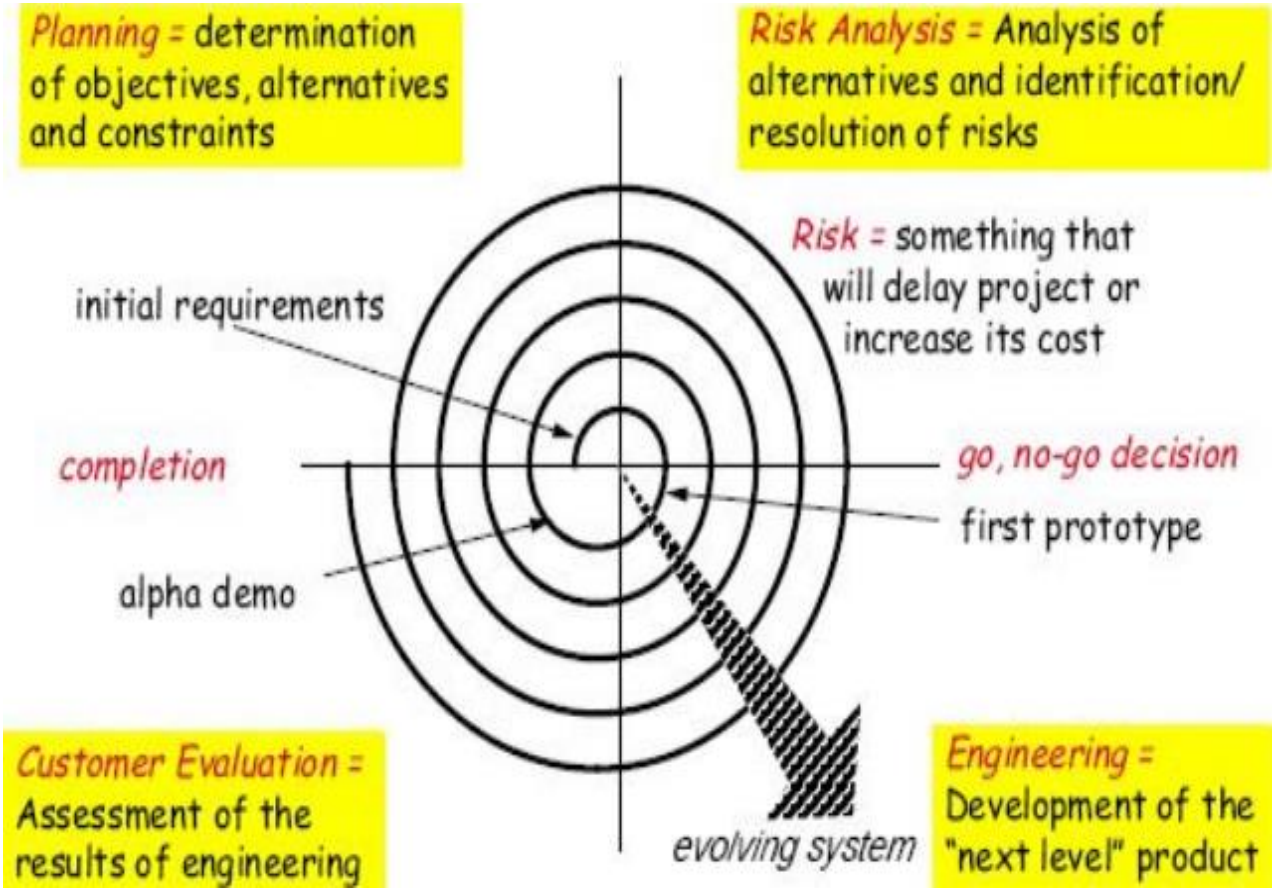
12. write phases of the waterfall model

- The classical software lifecycle model the software development as a step by step "waterfall" between the various development phases.
- The Waterfall model is the earliest SDLC approach that was used for software development.
- Requirements must be “frozen” to early in the life cycle
- Requirements are validated too late
- The waterfall Model illustrates the software development process in a linear sequential flow. This means that any phase in the development process begins only if the previous phase is complete. In this waterfall model, the phases do not overlap.

Phases of the waterfall model

1. Requirements /collaboration
2. Analysis
3. Design
4. Implementation
5. Testing
6. Maintena

13. Write phases of spiral model



- Spiral Model is very widely used in the software industry as it is in synch with the natural development process of any product
- Long-term project commitment because of potential changes to economic priorities as the requirements change with time.
- Requirements are complex and need evaluation to get clarity.

Pros:

1. Changing requirements can be accommodated.
2. Allows for extensive use of prototypes
3. Requirements can be captured more accurately.
4. Users see the system early.

Cons:

1. Management is more complex.
2. End of project may not be known early.
3. Process is complex
4. Spiral may go indefinitel

14. Write agile manifesto principal

Agile there are 4 manifesto

1. Individual and infections
2. Working software
3. Customer collaboration
4. Responding change

17 . Explain working methodology of the agile model and also write pros and cons.

- 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 products.
- 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.

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

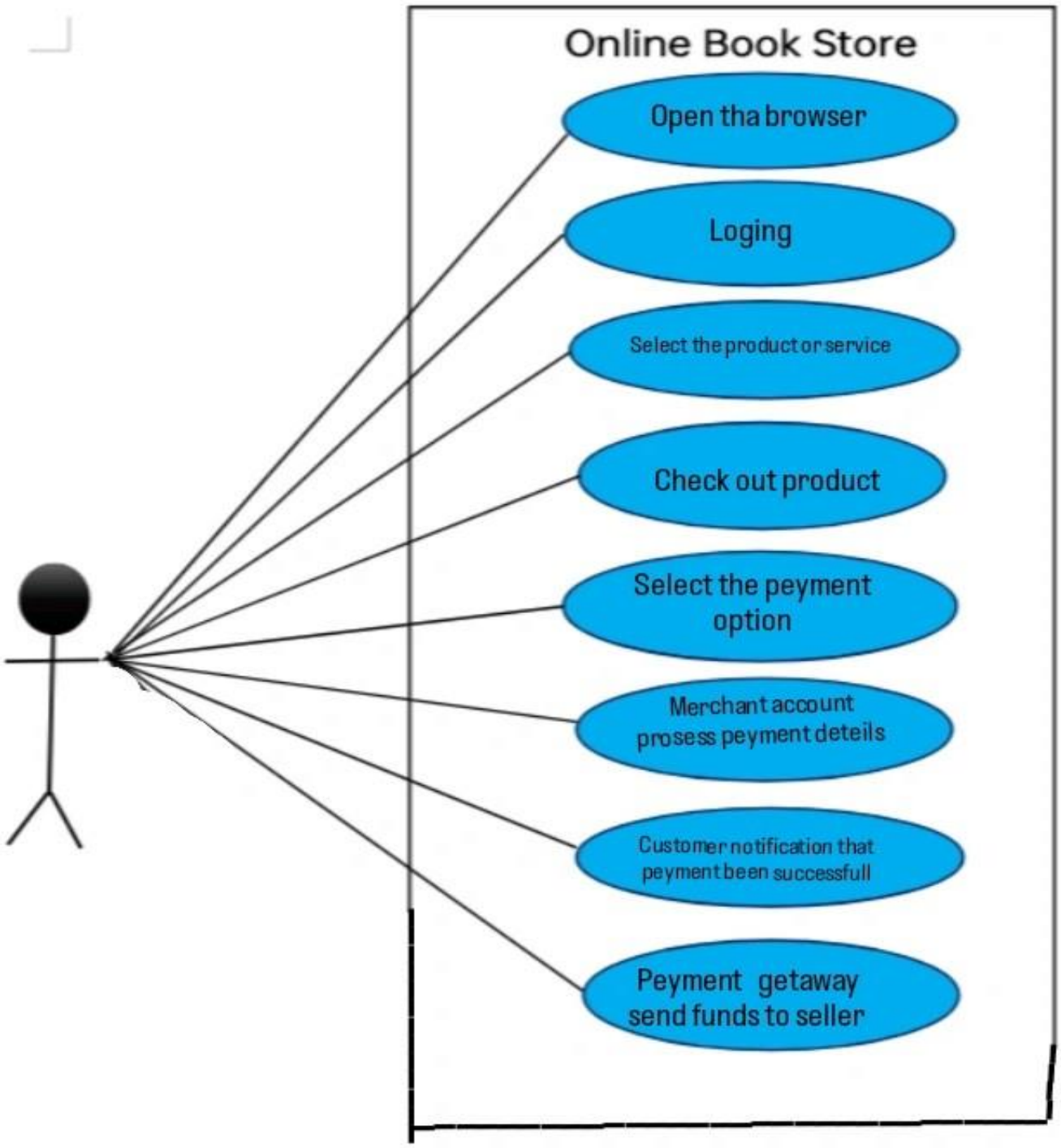
Pros:

1. Is a very realistic approach to software development
2. Promotes teamwork and cross training.
3. Functionality can be developed rapidly and demonstrated.
4. Resource requirements are minimum.

Cons:

1. Not suitable for handling complex dependencies.
2. More risk of sustainability, maintainability and extensibility
3. Strict delivery management dictates the scope, functionality to be delivered, and adjustments to meet the deadlines.

18. Draw Use case on online shopping products using DOC.



19. Draw Use case on online shopping product using payment gateway