Software Testing

Module: -1(Fundamentals)

1. What is SDLC

- SDLC stands for software development life cycle.
- SDLC is a process followed for software building within a software organization.
- SDLC is structured imposed on the development of software product
- It defines the process of planning, Implementation, testing Documentation, deployment, and maintenance.
- The software development lifecycle is the cost-effective and time -efficient process that development teams use design and High-quality software.

SDLC 6 Phases
1.Requirements Collection/Gathering
2.Analysis
3.Design
4.Implementation
5.Testing
6.Maintenance

2. What is software testing?

- Software testing is process used to identify the correctness, completeness, quality of developed computer software.
- Software Testing is the process of finding errors in the developed product.
- It also checks the real outcomes can match expected, results, defects, missing requirement gaps.
- Testing is executing a system in order to identify any gaps, errors or missing requirements in contrary to the actual desire or requirements

3. What is agile methodology?

- Agile SDLC model is 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 1 to 3 weeks.
- Every iterations involves cross functional teams working on various area like planning, requirement analysis, design, coding Unit testing, and acceptance testing.
- At the end of the iterations working product is displayed to the customer and important stakeholders.

4. What is SRS

- SRS Stands for software requirement specification.
- A software requirements specification (SRS) is a complete description of the behavior of the system to be developed.
- It includes set of use cases that describe all of the interactions that users will have with the software.
- Use cases are also known as functional requirement which impose constraints on the design or implementations (such as performance requirement, quality standards or design)
- These standards describe possible structures and qualities of software requirement specifications.
- Requirements are categorized in several ways
  Customer Requirements
  Functional Requirement
  Non-Functional Requirement

Customer Requirements

- Customer Requirements are the specific needs, wants or expectations of customer that a product or service must meet to be successful.

- These requirements can include functional specifications performance, design and more subjective such as aesthetics and user experience.

Functional Requirement

- Functional Requirement are very important system requirement in system design process.
- These Requirement are technical specifications, system design Parameters and guidelines, data manipulation, data processing and calculations modules etc.
- Functional Requirement are product features or functions that Developers must implement to enables users to their tasks.

Non-Functional Requirement

- Non-Functional Requirement 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.
- Non-Functional Requirement are a set of specifications that describe the system quality, constraints, or external interface.
- NFRs are referred to as quality attributes or software quality Requirements as product works.

5. What is Oops

- Oops stands for object oriented programming language.
- OOP is a programming technique in which program are Written on the basis of objects.
  C++, PHP, Java
- An object based programming language is one which easily Supports object-orientation.

- Object oriented programming style of programming that focus On using objects to design and build applications.

6. Write Basis concepts of oops

1) Object
2) Class
3) Encapsulation
4) Inheritance
5) Polymorphism
   i) overriding ii) overloading
6)Abstraction

1) Object: Any entity which has own state
   Example: Pen, Paper

2) Class: Class is a collection of object
   Example: human body

3) Abstraction: Hiding internal details and showing Functionalities.
   Example: login page

4) Encapsulation: Wrapping up of data or binding of data
   Example: capsule

5)Inheritance: When one object acquires all the properties and behavior of parent class
   Example: Parent-Child

6)Polymorphism: The ability to change form is known as polymorphism

7. 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.
- That is both data and function that operates

8. What is class

- Class is a collection of data members and members function.
- A class represents an abstraction of the object and abstract the properties and behavior of that object.
- An object is a particular instance of class which has actual existence and there can be many object for a class.

9. What is Encapsulation

- Encapsulation is the practice of including in an object everything it needs from other objects.
- The internal state is usually not accessible by other objects.
- Encapsulation is placing the data the function that work on that data in the same place.

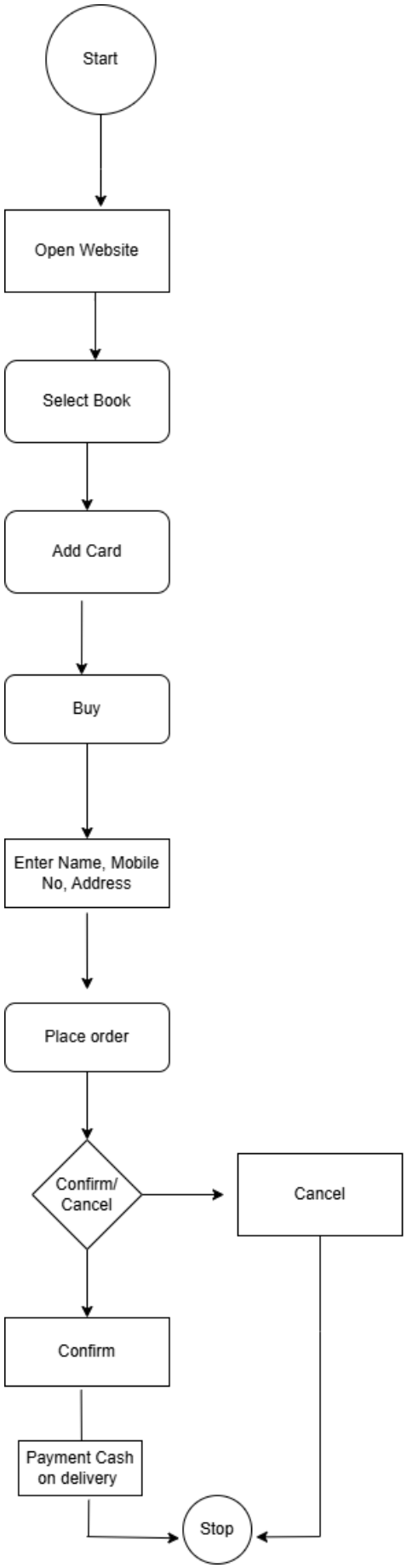
10. What is Inheritance

- Inheritance means that one's class inherits the characteristics of another class. This is also called as a relationship.
- Inheritance describes the relationship between two classes.
- A class can get some of it characteristics from a parent class and then add unique features of its own.

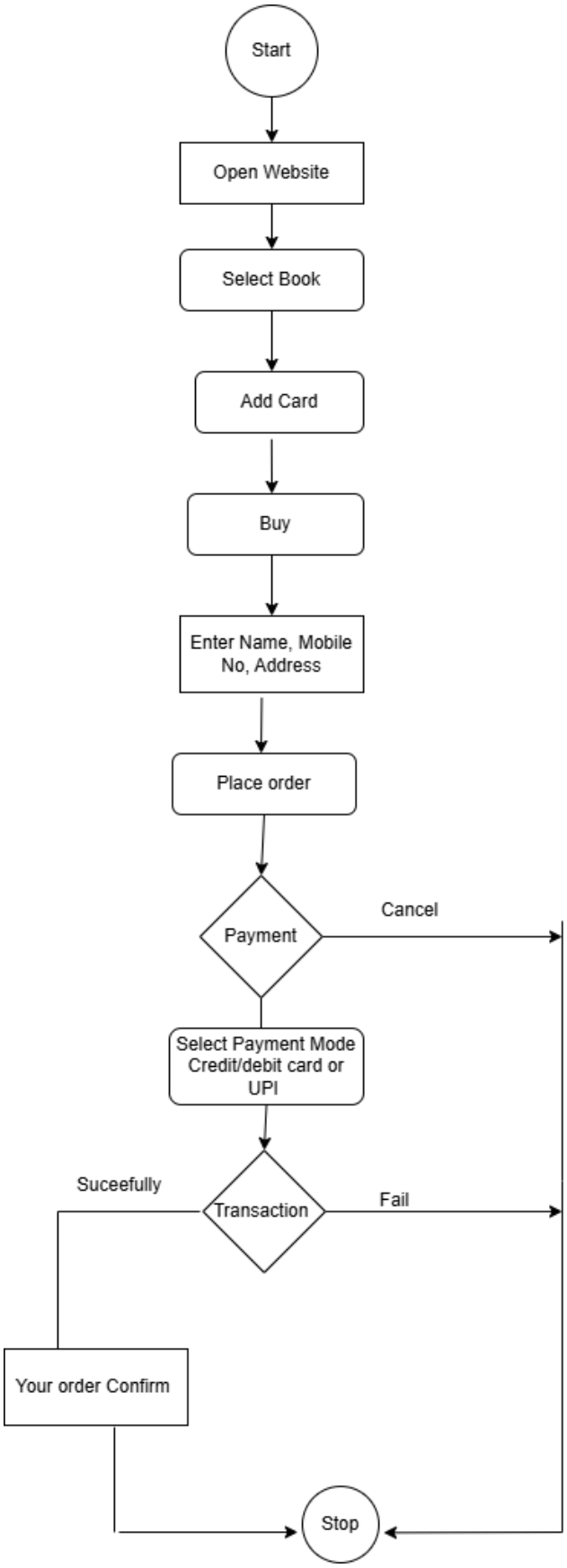
11. What is Polymorphism

- 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 ability to use an operate or function in different ways in other words giving different meaning or function to the operators or functions is called polymorphism.
- The ability to change form is known as polymorphism.
- There are two types of polymorphism.
    1. Compile time polymorphism (Overloading)
    2. Runtime time polymorphism (Overriding)

12. Draw Use case on Online Book Shopping.



13. Draw Usecase On Online bill payment system(paytm)



14.Write SDLC phases with basic Introduction

- There are six phases of SDLC

1. Requirement Gathering

- Requirement Analysis is a process used to determine the needs and expectations of a new product.
- It involves frequent communication with stakeholders and end-users of the product to define expectations, resolve, conflict, and document all key requirement.

2. Analysis

- The analysis phase defines the requirement of the system, independent of these requirement will be accomplished.
- This phase defines the problem that the customer is trying to solve.
- This phase starts with the requirement document delivered by the requirement phase and maps the requirement into architecture.

3. Design

- The design phase of SDLC is a critical step in developing the conceptual of software project.
- The design phase is a stage of software developers of the product depending on the project, the can include screen designs, databases, sketches, system interfaces and prototypes.
- Client use these details to make final product design choices.

4.Implementation

- This phases is initiated after the system has been tested and accepted by the user.

- The implementation phase deals with issue of quality, performance, baselines, libraries, and debugging.
- The end deliverable is the product itself. There are already many established techniques associated with implementation.

5. Testing

- The testing phase is a different team after the implementation is completed.
- The developers build the software then its deployed in the testing environment.
- Then the testing team tests the functionality of the entire system, the testing is done to ensure that the entire applications works according to the customer requirements.

6. Maintenance

- Software maintenance is one of the activities in software engineering and is the process of enhancing and optimizing deployed software as well as fixing defects.
- Software maintenance is also one of the phases in the system development life cycle, it applies to software development.
- The maintenance phase is the phase which comes after deployment of the software into the field.

There are three types in maintenance

1. Correctiveness maintenance
2. Adaptive maintenance
3. Perfective maintenance

15. Explain phases of waterfall model

- The classical software lifecycle models the software development as a step-by-step “waterfall” between the Various development phases.
- The waterfall is unrealistic for many reasons especially.

1) Requirement Analysis

- The collect complete needs and then analysis and define needs that must be met by the program to be built.
- In this phase all requirement of the project are analysis and document I a specification document and a feasibility analysis is done to check if the requirement is valid.

2) System Design

- The goal of this phase is a convert the requirement acquired in the SRS into a format that can be coded in a programming language.
- It includes high-level and detailed design as well as the overall software architecture.
- A software design document is used to document all of this effort (SDD)

3) Implementation

- The implementation phase is when programmers assimilates the requirement and specification from the previous phase and produce actual code.

4) Testing

- Testing is a type pf software testing in which the different testing levels are performed one after the other.
- Testing is waterfall development is sequential and through, consisting of unit testing, integration testing.

5) Deployment

- Once the function and non-functional testing is done; the product is deployed in the customer environment or released into the market.

6) Maintenance

- Maintenance is the most important phase of a software life cycle.
- The effort spent on maintenance is 60% of the total effort spent to develop a full software.
- There are basically three types of maintenance

1. Corrective Maintenance

- This type of maintenance is carried out of to correct errors that were not discovered during the product development phase.

2. Perfective Maintenance

- This type pf maintenance is carried out to enhance the functionalities of the system based on the customer's request.

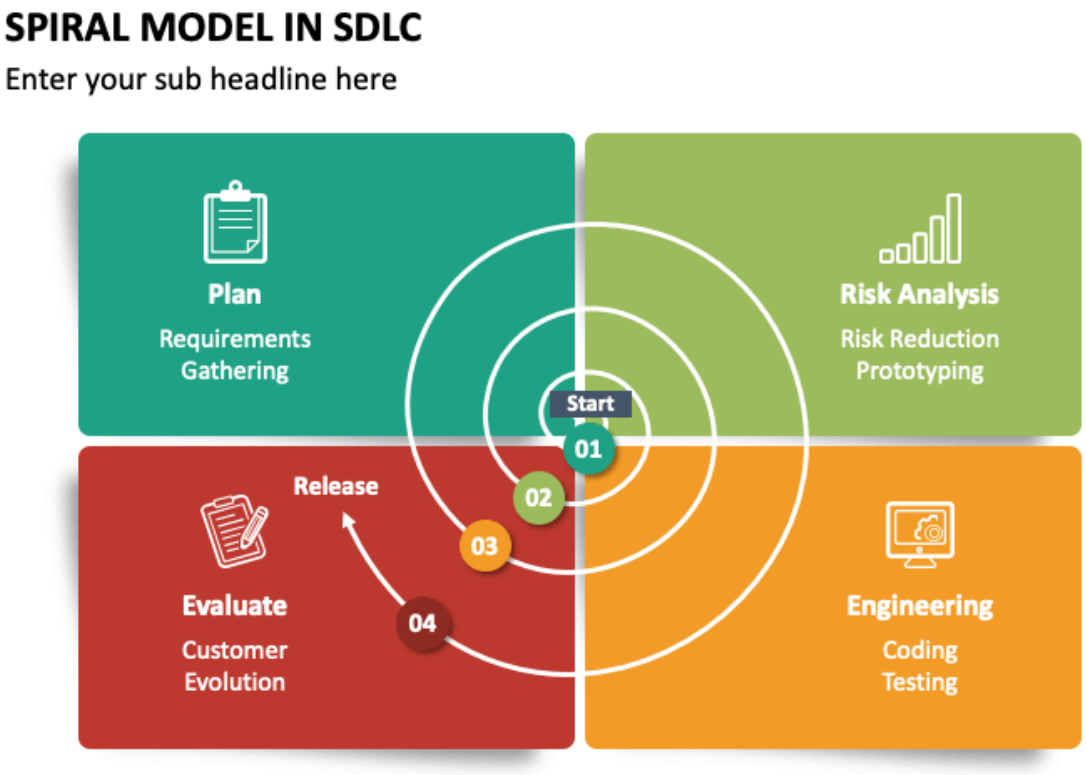
3. Adaptive Maintenance

- Adaptive maintenance is usually required for porting the software to work in a new environment such as working in a new computer platform or with a new operating system.

16. Write Phases of Spiral Model

- The spiral model is one of most important software development life cycle models, which provides support for risk handling.
- The exact number of loop of the spiral is known and can vary from project to project.
- Each loop of the spiral is called a phase of the software development process.
- The spiral model is a software development life cycle (SDLC) model that provides a systematic and iterative approach to software development.

- The spiral model is a risk-driven model, meaning that the focus is on managing risk through multiple iterations of the software development process.



1. Planning: The First phase of the spiral model is the planning phase, where the scope of the project is determined and a plan is created for the next iteration of the spiral.
2. Risk Analysis: In Risk Analysis phase, the risks associated with the project are identified and evaluated.
3. Evaluation: In the Evaluation phase, the software is evaluated to determine if it meets the customer's requirement and if it is of high quality.
4. Engineering: In the engineering phase, the software is developed based on the requirement gathered in the previous iteration.

17. Explain Working Methodology of 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 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.

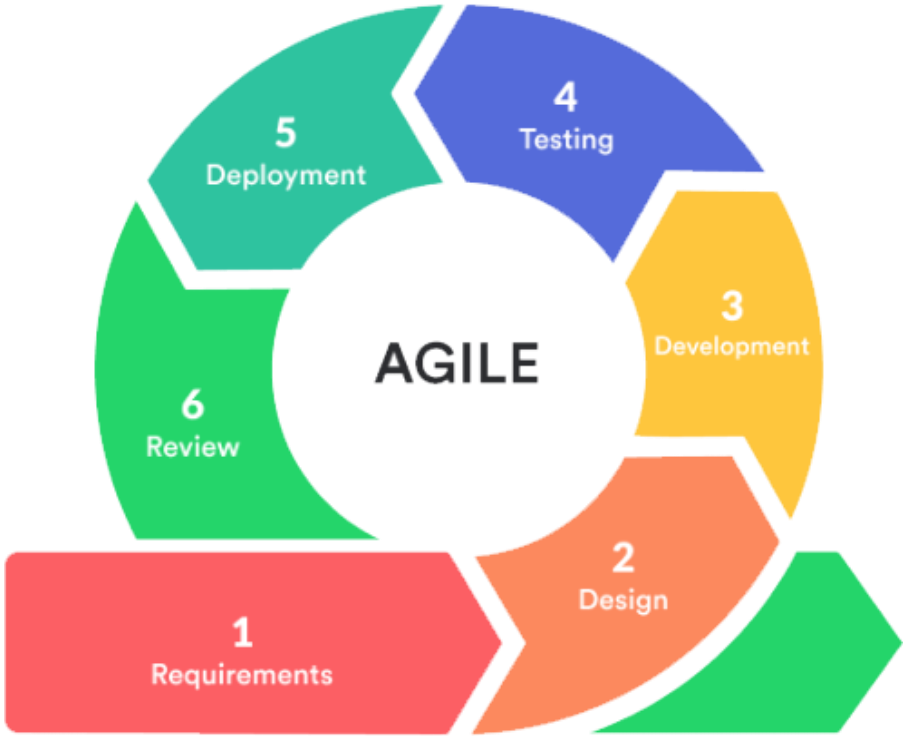
Pros:

- 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.

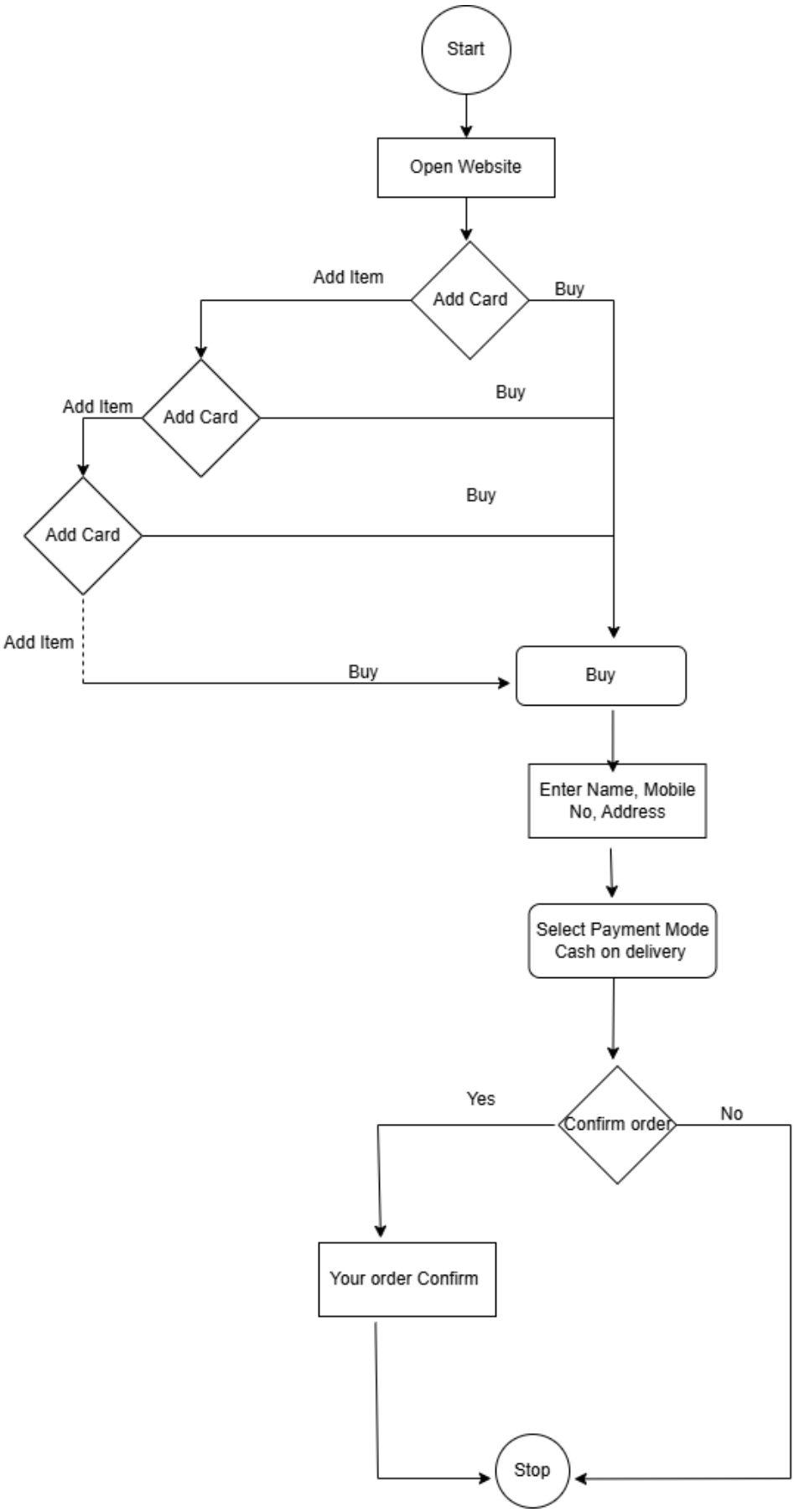
Cons:

- 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.



18. Draw Usecase On Online Shopping product using COD.



19. Draw Usecase On Online Shopping using payment gateway.