Software Testing Assignment

Module–1(Fundamental)

1. What is SDLC

- A Software Development Lifecycle 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.
- software development life cycle, is a methodology that defines the entire procedure of software development step-by-step.
    - Requirements collection/ gathering
    - Analysis
    - Design
    - Implementation
    - Testing
    - Maintenance

2. What is software testing?

- Testing is the process of evaluating a system or its components with the intent to find that whether it satisfies the specified requirements or not.
- Software Testing is a process used to identify the correctness, completeness, and quality of developed computer software.
- 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?

- There are various methodologies present in agile testing and those are listed below:
    - Scrum
    - extreme Programming
- Below listed methodologies are used less frequently
    - Dynamic System Development Method (DSDM)
        - This is an Iterative and incremental approach that emphasizes on the continuous user involvement.
    - Test Driven Development (TDD)
        - This is a technique which has short iterations where new test cases covering the desired improvement or new functionality are written first.

- Feature Driven Development
    - This is an iterative and incremental software development process and this can aim depends on the features.
- XBreed
    - Agile enterprise previously known as XBreed. It is agile way of managing, architecting and monitoring the enterprise.
- Crystal
    - Crystal is an adaptive technique mainly used for software development methodologies.

4. What is SRS

- A software requirements specification (SRS) 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 (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.

5. What is oops

- Object Oriented Programming (function)
- Object = data(properties) + functionality(methods)
- Object is real world entity.
- Identifying objects and assigning responsibilities to these objects.
- Objects communicate to other objects by sending messages.
- Messages are received by the methods of an object
- An object is like a black box.
- The internal details are hidden.

6. Write Basic Concepts of oops

- Object
- Class
- Encapsulation
- Inheritance

- Polymorphism
    - Overriding
    - Overloading
- Abstraction

7. What is object

- That is both data and function that operate on data are bundled as a unit called as object.
- Object = Data + Methods

8. What is class

- Blue print for an object.
- A class represents an abstraction of the object and abstracts the properties and behavior of the object.
- An object is a particular instance of a class which has actual existence and there can be many objects for a class.

9. 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.
- Encapsulation In java is the process of wrapping up of data (properties) and behavior (methods) of an object into a single unit.

10. What is inheritance

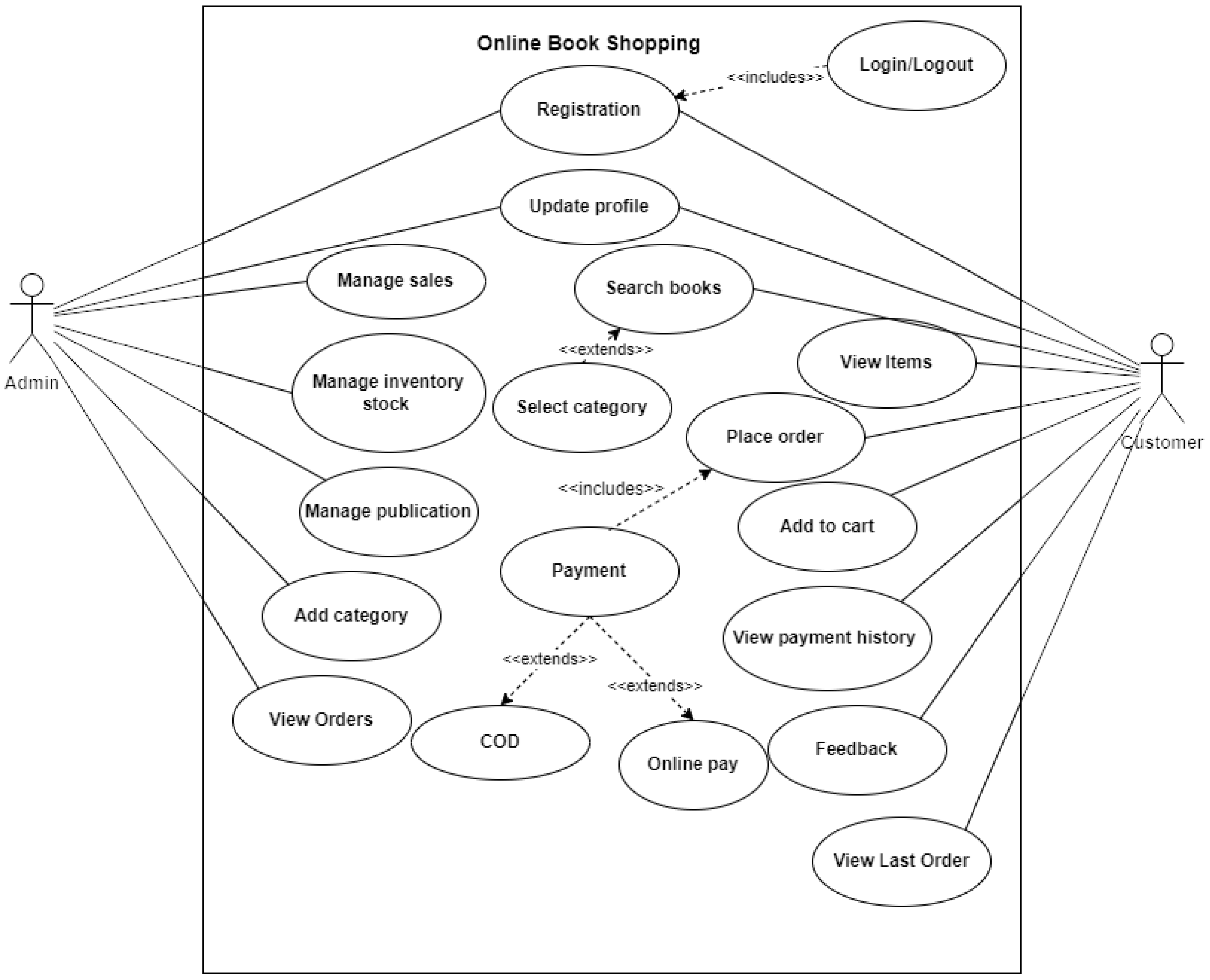
- Inheritance means that one class inherits the characteristics of another class. This is also called a “is a” relationship.
- A programming technique that is used to reuse an existing class to build a new class is known as inheritance.

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.

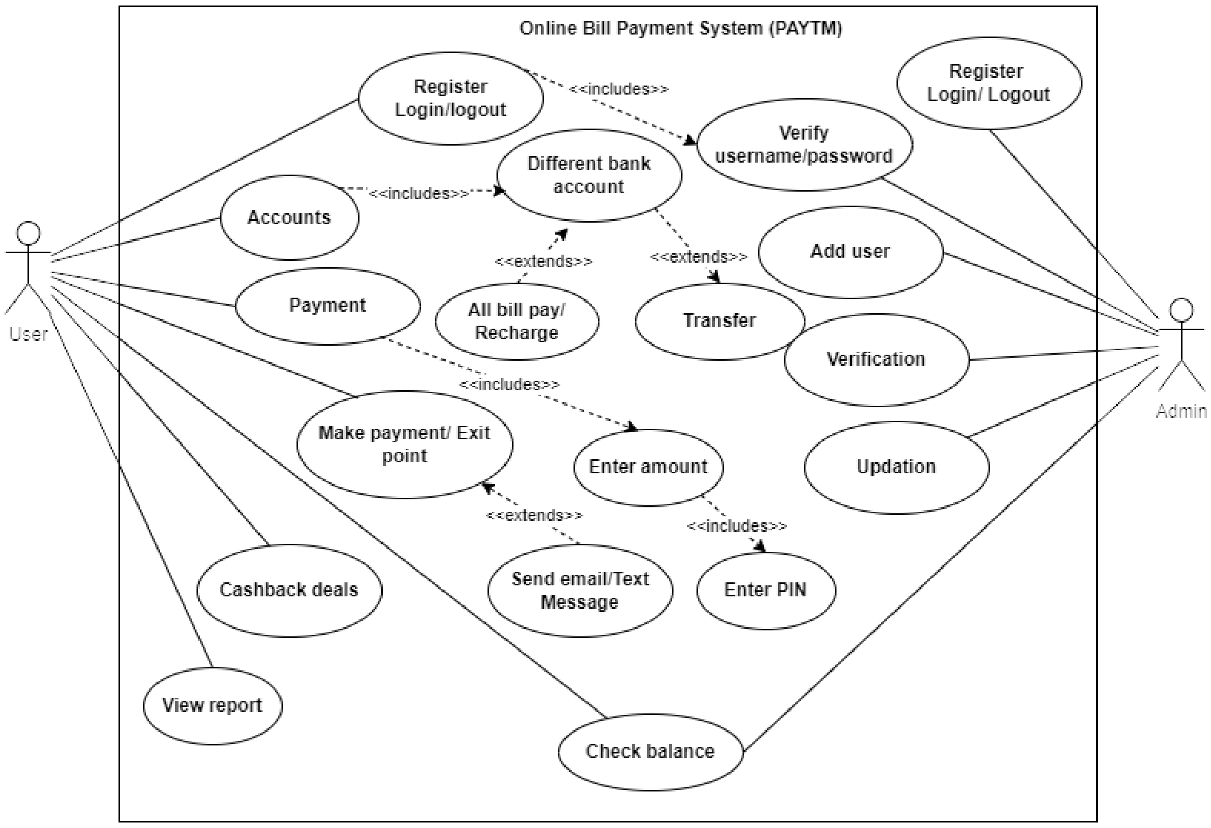
12. Draw Use case on Online book shopping

➔



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

➔



14. Write SDLC phases with basic introduction

- software development life cycle, is a methodology that defines the entire procedure of software development step-by-step.

  - Requirements collection/ gathering
  - Analysis
  - Design
  - Implementation
  - Testing
  - Maintenance

- **Requirements collection/ gathering:**
  - Establish Customer Needs
  - Although requirements may be documented in written form, they may be incomplete, unambiguous, or even incorrect.
  - Types of requirements:
    - Functional requirements:

            - Describe system services or functions.
            - Compute sales tax on purchase.
            - Update the database on the server.
        - Non-functional requirements:
            - Are constraints on the system or the development process.
            - Non-functional requirements may be more critical than functional requirement.
            - If these are not met, the system is useless.

- **Analysis:**
    - Model and Specify the requirements- “What”.
    - This Phase defines the problem that the customer is trying to solve.
    - This analysis represents the “what” phase.
    - The analysis phase defines the requirements of the system, independent of how these requirements will be accomplished.
    - This phase represents the “how” phase.
    - Software Requirements Specification (SRS) (documentation).
    - 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.

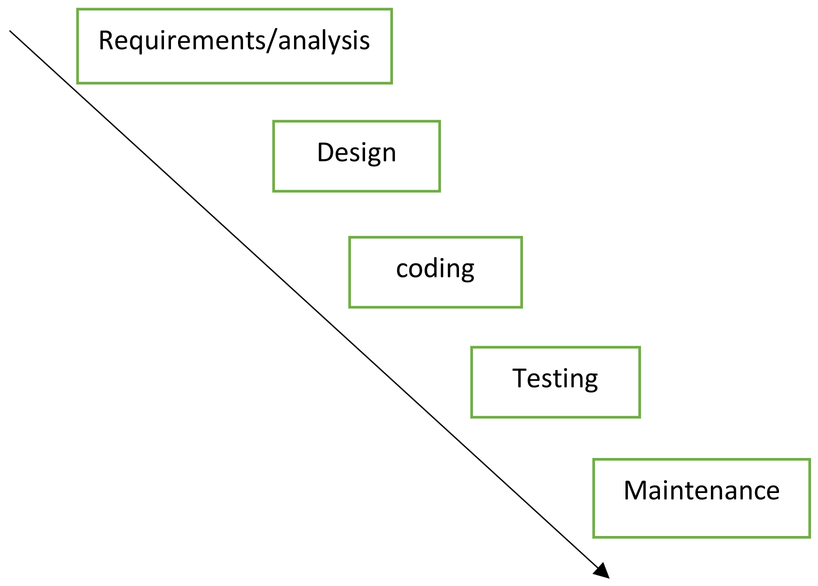
- **Design Phase:**
    - SRS is a reference for software designers to come up with the best architecture for the software.
    - UI/UX design
    - Higher level design
    - Implementation plan
    - Test Plan
    - Analysing the trade-offs of necessary complexity allows for many things to remain simple which, in turn, will eventually lead to a higher quality product. The architecture team also converts the typical scenarios into a test plan.
    - Critical priority analysis

- **Implementation Phase (coding):**
  - This Phase is initiated after the system has been tested and accepted by the user. In this phase, the system is installed to support the intended business functions. System performance is compared to performance objectives established during the planning phase.
  - 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.
    - Implementation - Code
    - Critical Error Removal
  - 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

- **Maintenance Phase**: (deployment phase)
  - Software maintenance is one of the activities in software engineering, and is the process of enhancing and optimizing deployed software (software released), as well as fixing defects.
  - Corrective maintenance
    - Identifying and defects
  - Adaptive maintenance
    - New platform
  - Perfective maintenance
    - New requirement/ New features(add)

15. Explain Phases of the waterfall model



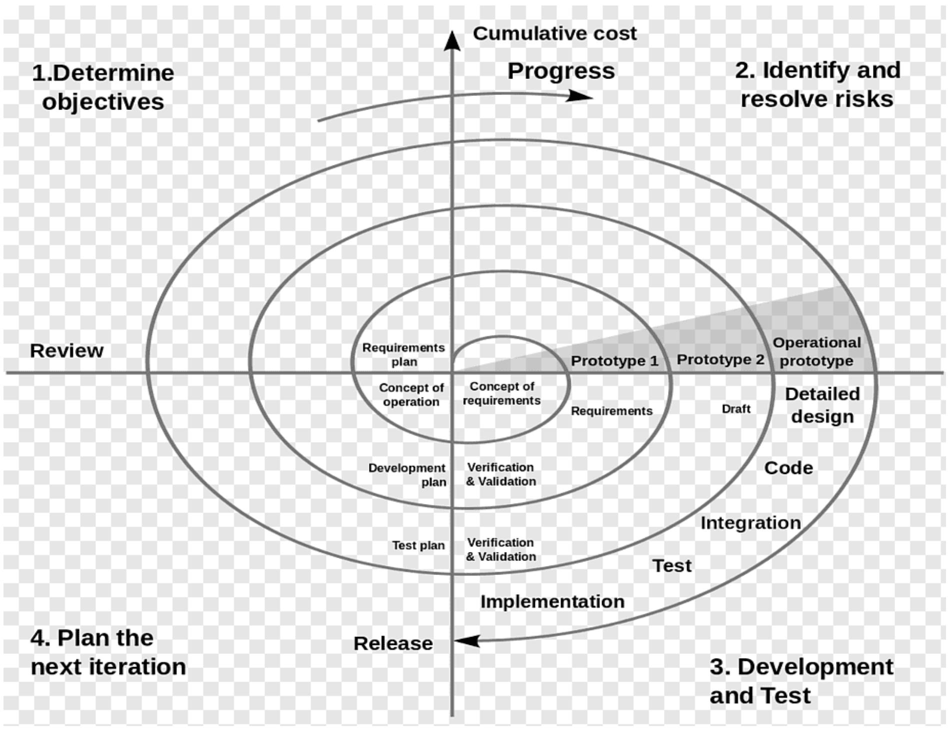
Pros:

- Simple and easy to understand and use.
- Easy to manage because each phase has specific deliverables and a review process.
- Each phase is processed and completed one at a time. Phases do not overlap.
- Works well for smaller projects where requirements are clearly defined and very well understood.

Cons:

- Once an application is in the testing stage, it is very difficult to go back and change something.
- No working software is produced until late during the lifecycle.
- Not a good model for complex and object- oriented projects.
- Poor model for long and ongoing projects.
- Not suitable for the projects where requirements are at a moderate to high risk of changing.

16. Write phases of spiral model



- The spiral model is a systems development lifecycle method used for risk management that combines the iterative development process model with elements of the waterfall model. The spiral model is used by software engineers and is favoured for large, expensive and complicated projects.

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.

17. Write agile manifesto principles

- **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.

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:

- Lack of documentation
- 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

18. 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:

- In Agile methodology the delivery of software is unremitting.
- The customers are satisfied because after every Sprint working feature of the software is delivered to them.
- Customers can have a look of the working feature which fulfilled their expectations.
- If the customers have any feedback or any change in the feature then it can be accommodated in the current release of the product.
- In Agile methodology the daily interactions are required between the business people and the developers.
- In this methodology attention is paid to the good design of the product.
- Changes in the requirements are accepted even in the later stages of the development.
- An Agile/Scrum approach can improve organizational synergy by breaking down organizational barriers and developing a spirit of trust and partnership around organizational goals.

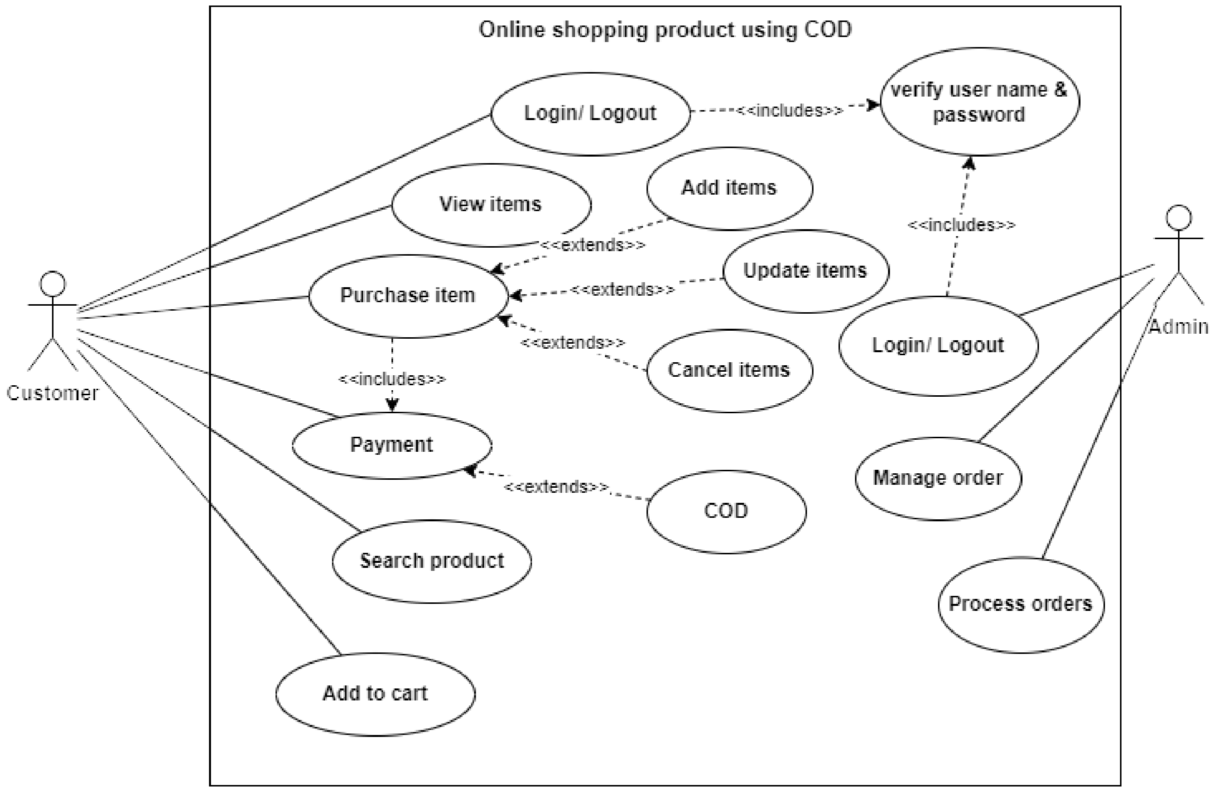
Cons:

- In Agile methodology the documentation is less.
- Sometimes in Agile methodology the requirement is not very clear hence it’s difficult to predict the expected result.
- In few of the projects at the starting of the software development life cycle it’s difficult to estimate the actual effort required.
- Because of the ever-evolving features, there is always a risk of the ever-lasting project.

- For complex projects, the resource requirement and effort are difficult to estimate

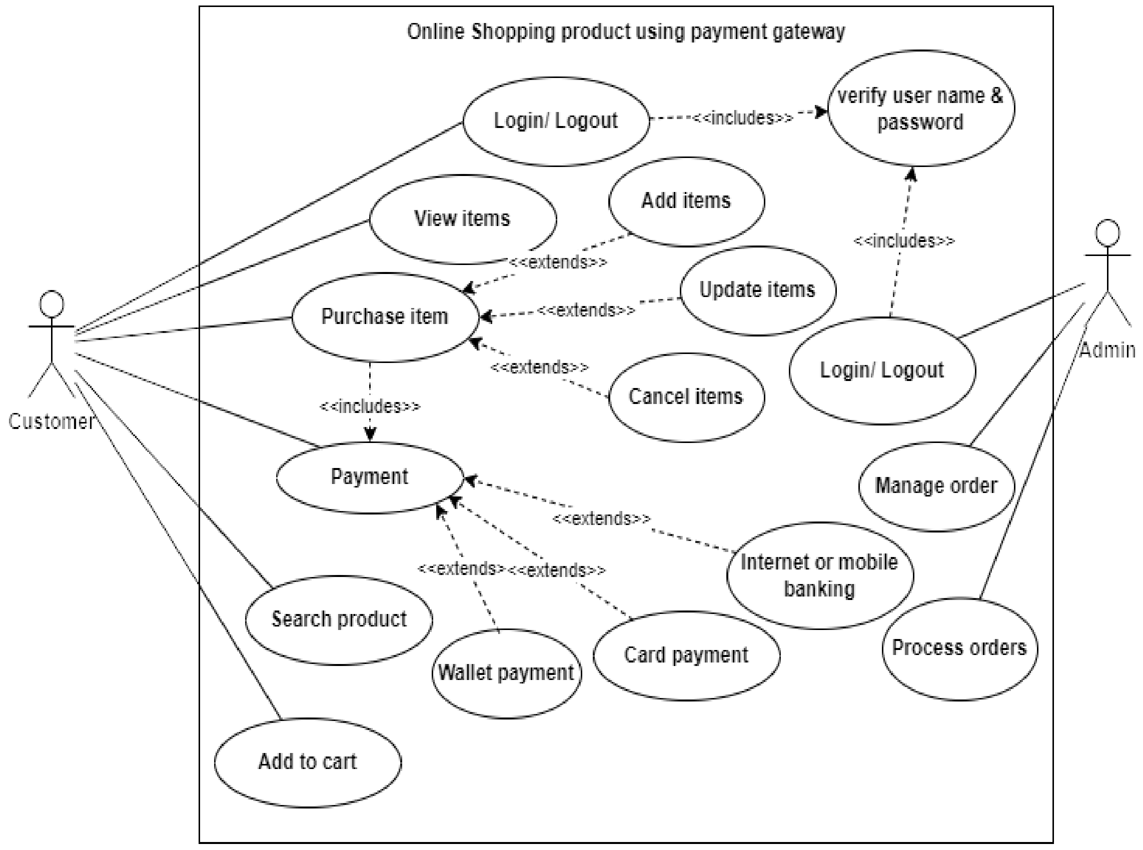
19. Draw use case on Online shopping product using COD.

➔



20. Draw use case on Online shopping product using payment gateway

➔


Online Shopping product using payment gateway
Login/ Logout
<<includes>>
verify user name & password
View items
Add items
<<extends>>
Update items
Purchase item
<<extends>>
<<extends>>
Cancel items
<<includes>>
Login/ Logout
<<includes>>
Customer
Admin
Payment
<<extends>>
Manage order
Internet or mobile banking
<<extends>>
<<extends>>
Search product
Wallet payment
Card payment
Process orders
Add to cart