Software Testing Assignment

Module – 1(Fundamental)

1. **What is software testing?**

- **Software Testing is a process used to identify the correctness, completeness, and quality of developed computer software.**

2. **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.**

3. **What is agile 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.**

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.**

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 an object**
- **An object is like a black box. The internal details are hidden.**

6. **Write Basic Concepts of oops?**

- **Class : Class is a collection of data member & member functions.**
- **Object : Object gives permission to access functionality of class.**
- **Encapsulation : Wrapping of data in single unit.**
- **Inheritance : Deriving the attributes of some other class.**
- **Polymorphism : One name multiple form.**
- **Abstraction : Hiding details showing only essential information**

7. **What is object?**

- **Object gives permission to access functionality of class.**
- **An "object" is anything to which a concept applies.**
- **Tangible Things Roles Incidents Interactions Specifications**

8. **What is class?**

- **Class is a collection of data member & member functions.**
- **A class represents an abstraction of the object and abstracts the properties and behaviour of that object.**
- **When you define a class, you define a blueprint for an object.**

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.**

10. What is inheritance?

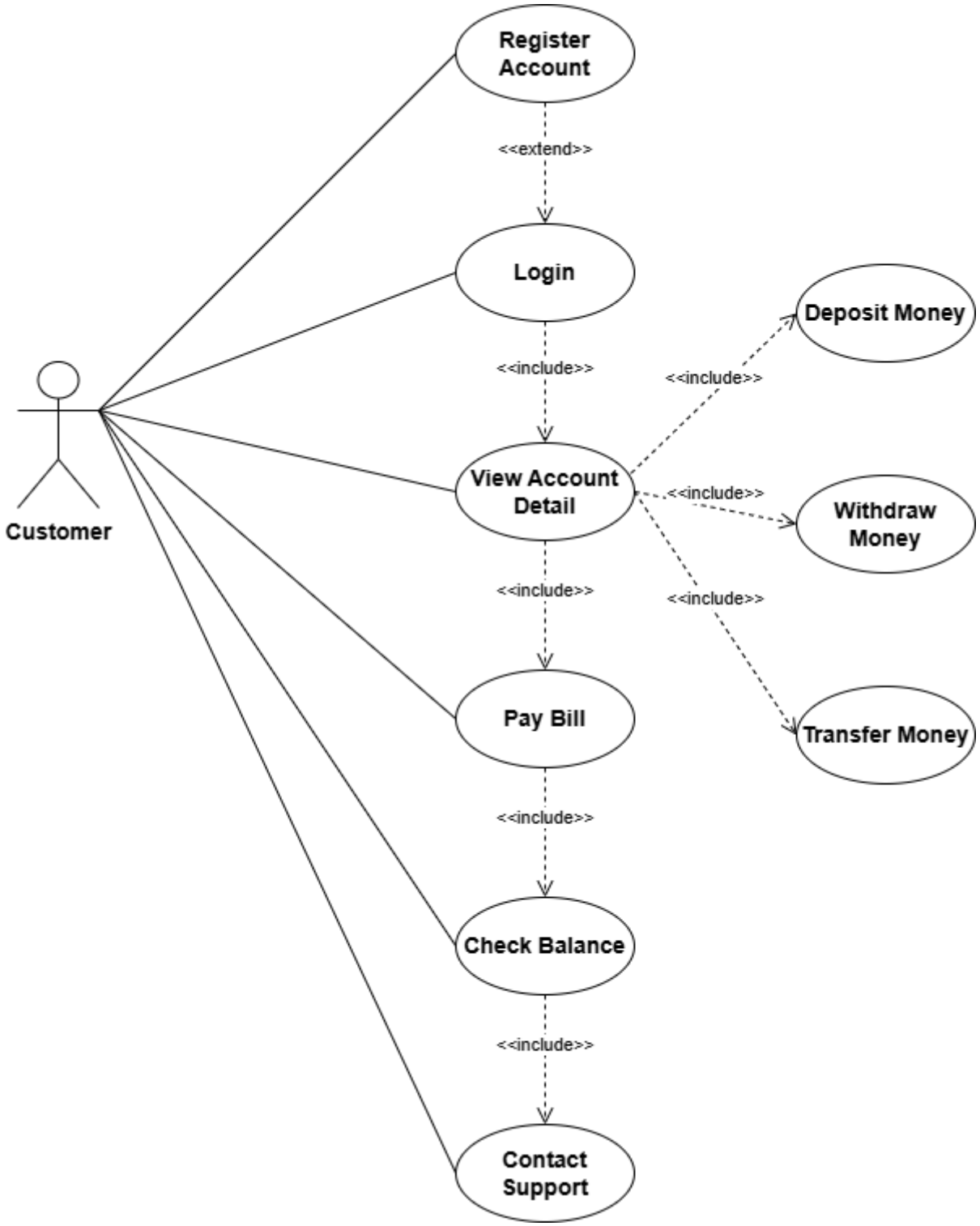
- **Inheritance means that one class inherits the characteristics of another class. This is also called a "is a" relationship.**

11. What is Polymorphism?

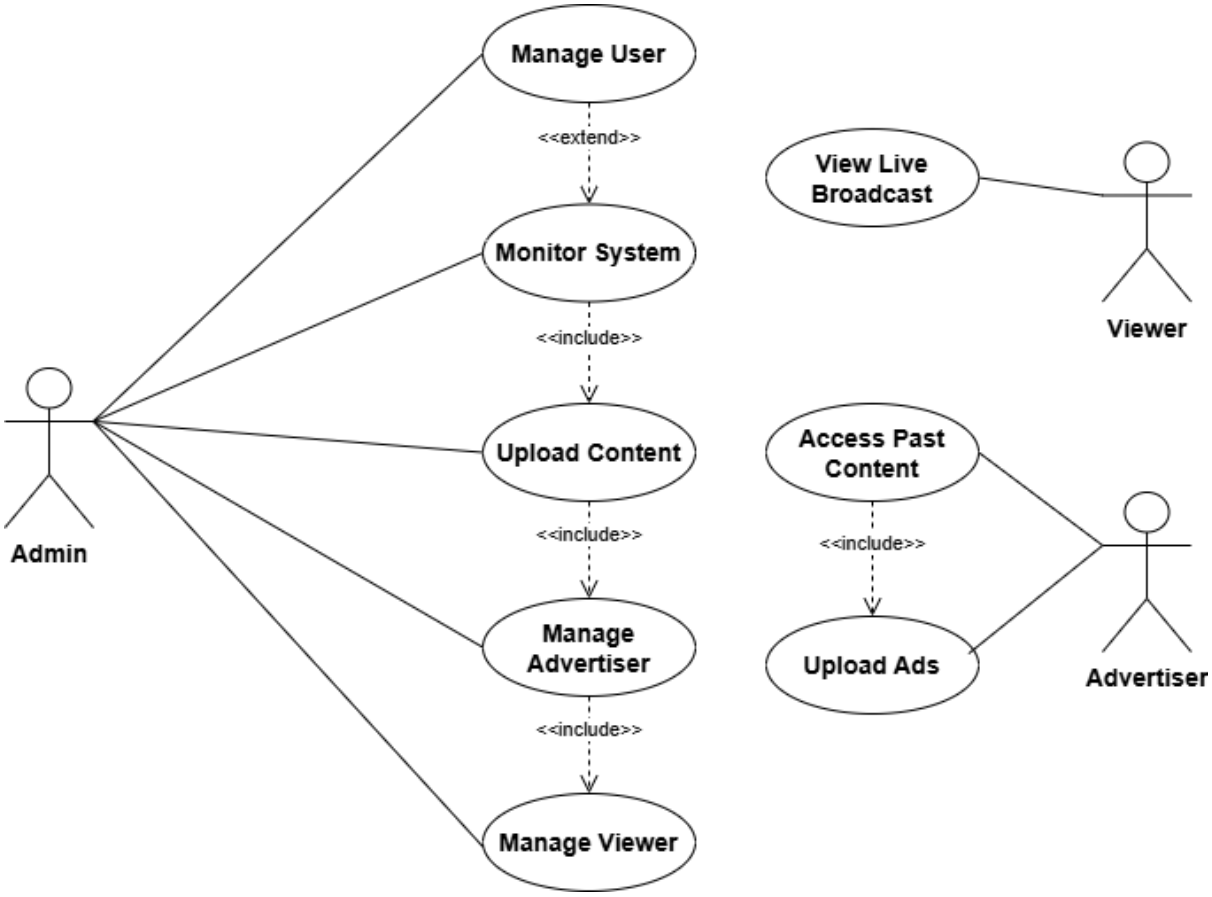
- **Polymorphism means "having many forms".**
- **It allows different objects to respond to the same message in different ways, the response specific to the type of the object.**
- **The ability to change form is known as polymorphism.**

12. Draw Use-case on online bill payment system (paytm)

13. Draw Use-case on banking system for customers.



14. Draw Use-case on Broadcasting System.



15. Write SDLC phases with basic introduction?

Requirements Collection/Gathering	Establish Customer Needs
Analysis	Model And Specify the requirements- “What”
Design	Model And Specify a Solution – “Why”
Implementation	Construct a Solution In Software
Testing	Validate the solution against the requirements
Maintenance	Repair defects and adapt the solution to the new requirements

- **Requirement Gathering :-**

- **Features**
- **Usage scenarios**
- **Although requirements may be documented in written form, they may be incomplete, unambiguous, or even incorrect.**
- **Requirements will Change!**

- **Requirement Gathering(Cont…) :-**

- **Requirements definitions usually consist of natural language, supplemented by (e.g., UML) diagrams and tables.**
- **Three types of problems can arise:**

1) **Lack of clarity: It is hard to write documents that are both precise and easy-to-read.**
2) **Requirements confusion: Functional and Non-functional**
3) **requirements tend to be intertwined.**
4) **Requirements Amalgamation: Several different requirements**

- **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.**

- **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.**

- **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.**

- **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.**

- **Maintenance Phase** :-

  - **Software maintenance is one of the activities in software engineering, and is the process of enhancing and optimizing deployed software (software release), as well as fixing defects.**
  - **Software maintenance is also one of the phases in the System Development Life Cycle (SDLC), as it applies to software development. The maintenance phase is the phase which comes after deployment of the software into the field.**

16. Explain Phases of the waterfall model.

- **The waterfall is unrealistic for many reasons, especially :-**
- **Requirements must be "frozen" to early in the life cycle Requirements are validated too late 60**

- **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.**

- **Pros (Why Waterfall Model) :-**
  - **Simple and easy to understand and use**
  - **Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.**
  - **Phases are processed and completed one at a time.**
  - **Works well for smaller projects where requirements are very well understood.**
  - **Clearly defined stages.**

- **Cons (Why not Waterfall Model) :-**
  - **No working software is produced until late during the life cycle. High amounts of risk and uncertainty.**
  - **Not a good model for complex and object-oriented projects. Poor model for long and ongoing projects.**
  - **Not suitable for projects with a moderate to high risk of changing requirements. So risk and uncertainty are high**

with this process model.

- **It is difficult to measure progress within stages. Cannot accommodate changing requirements.**

17. Write phases of spiral model

- **Application :-**

  - **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 for the customer as well as the development firms. Following are the typical uses of Spiral model:**
  - **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.**

- **Pros (Why It works) :-**

  - **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 (Why It doesn't work) :-**

  - **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.**

18. Write agile manifesto principles

- **What is Agile? :-**

  - **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**

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

- **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.**

- **Pros :-**

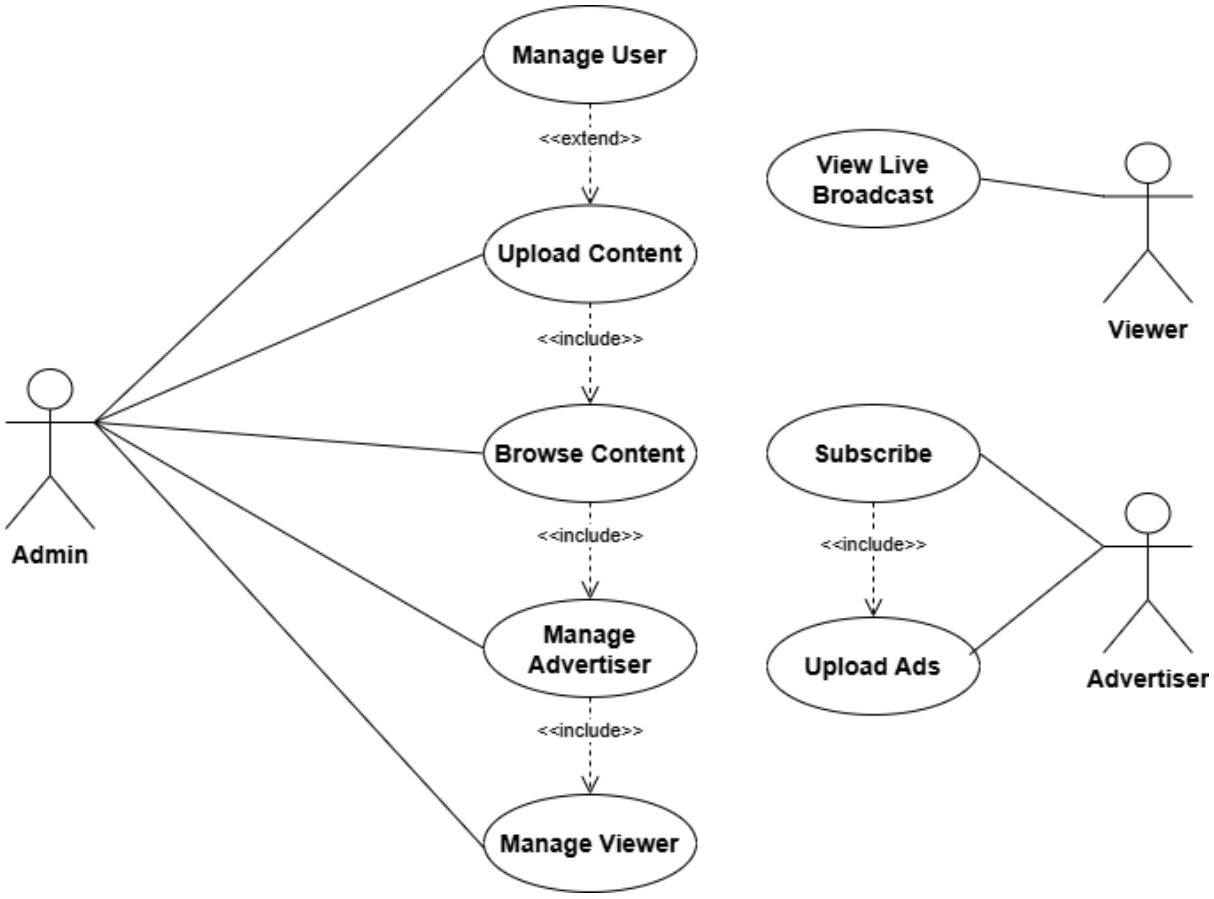
  - **Is a very realistic approach to software development Promotes teamwork and cross training.**
  - **Functionality can be developed rapidly and demonstrated. Resource requirements are minimum.**
  - **Suitable for fixed or changing requirements Delivers early partial working solutions.**
  - **Good model for environments that change steadily. Minimal rules, documentation easily employed.**
  - **Enables concurrent development and delivery within an overall.**

- **Cons :-**

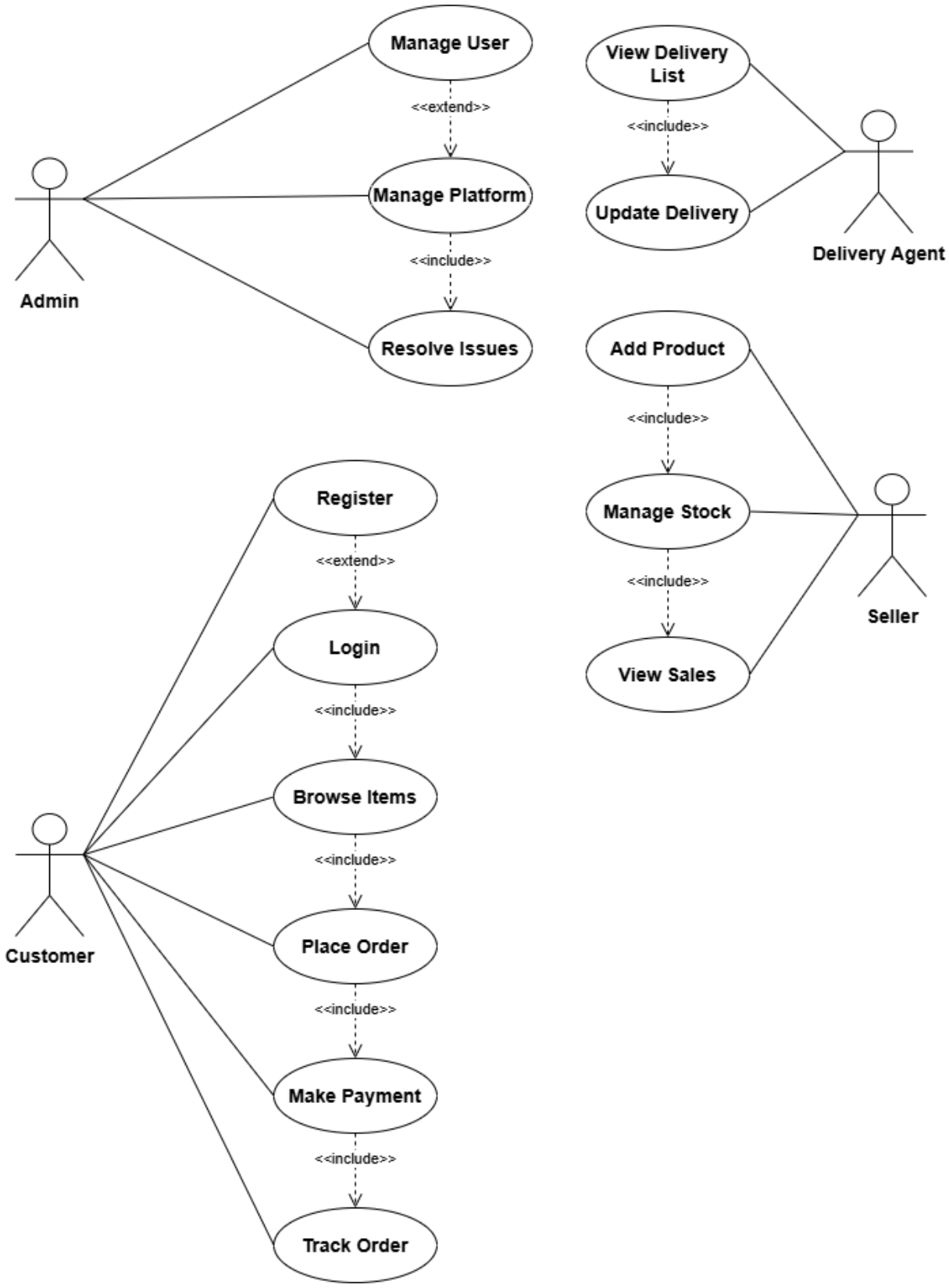
  - **Not suitable for handling complex dependencies.**

- **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.**

20. Draw use-case on OTT Platform.



21. Draw use-case on E-commerce application



22. Draw use-case on Online shopping product using payment gateway.