**1. What is SDLC?**

* In Short, SDLC(Software Development Life Cycle)
* Enables the development of high quality , low–cost Software with the shortest possible development time.
* A major objective of the SDLC is to produce high-quality
* Software that meets and exceeds the expectations of customers
* SDLC provides a detailed plan with a series of
* Stages, or phases ,that encompass their own processes and deliverables.

**Phases of SDLC:**

|  |  |
| --- | --- |
| 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 |

**2. What is Software Testing?**

* Software testing involves evaluating and verifying software product’s functionality.
* Basically, it checks whether the software product matches anticipated requirements and makes sure it is Defect free.
* It can be said that testing enhances the quality of the product by preventing bugs, reducing development costs ,and reducing performance issues.

**Types of Software Testing:**

1. **Unit Testing**
2. **Integration Testing**
3. **System Testing**
4. **Performance Testing**
5. **User Acceptance Testing**
6. **Stress Testing**
7. **Usability Testing**
8. **Security Testing**

**1. Unit Testing**: A programmatic test that test the internal working of a unit of code, such as a method or function.

**2. Integration Testing:** Ensures that multiple components of systems work as expected when they are combined to produce a result.

**3.System Testing:** Complete end to end testing is done on the complete software to make sure the whole system works as expected.

**4.Performance Testing:** Ensures that the software performs according to the user’s expectations by checking response time and throughput under specific load and environment.

**5.User Acceptance Testing:** Ensures the software meets the requirements of the clients or users.

**6.Stress Testing:** Ensures that the performance of the software doesn’t degrade when the load increase.

**7.Usability Testing:** Measures how usable the software is.

**8. Security Testing:** Now more important than ever. This testing tries to break software’s security checks, to gain access to confidential data.

1. **What is Agile Methodology?**

Agile methodology is a project managemement framework that breaks projects down into several dynamic phases, commonly known as sprint.

* Agile SDLC model is combination of iterative and incremental process models with focus on process adaptability and customer satisfaction by rapid delivery of a working software product.

**Fig: Lifecycle of Agile Methodology**

1. **What is SRS?**

* SRS stands for Software Requirements Specification.
* It’s **a** document that describes the requirements and expectations for a software projects.
* The SRS is a blueprint for the software, and it’s used by the development team and other stakeholders.
* **SRS can Include :**
* **Business Requirements:** The high –level goals for the project
* **Users Requirements:** The needs of the end users
* **Technical Requirements:**  The functionality of the product
* **Constraints:** limitations on the design, such as hardware or software limitations
* **Use cases :** Scenario that describes how the software will interact with users

**Characteristics of Software requirement Specification:**

1. Unambiguous
2. Complete
3. Verifiable
4. Consistence
5. Modifiable
6. Traceable
7. Usable during the operation and Maintenance phase

Specification Types

**Fig: Software requirement Specification**

1. **What is oops?**

* OOP is an **object –oriented programming**.
* Technique that combines data and instructions.
* For processing that data into an object that can bused within the program.
* Identifying the Objects and assigning responsibilities of these objects.
* Objects communicates to other objects by sending message.
* Messages are received by methods of an object.
* Object is derived from abstract data type.
* **Everything in the world is an object**
* A flower, a tree ,an animal
* A student, a professor
* A desk, a chair, a classroom, a building

A university, a city, a country

* The world ,the universe
* A subject such as CS,IS, Math , History,….

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

1. Object
2. Class
3. Encapsulation
4. Inheritance

5. Polymorphism

1. Overriding

2. Overloading

6. Abstraction

**1. Object:** It is a instance of class.

**2. Class:** Group of data and function.

**3. Encapsulation:** To bind a data and code in single unit.

**4. Inheritance:** The object of one class can acquire the properties of object of another class.

**5. Polymorphism:** It means “having many forms”.

**6. Abstraction:** Is a class that cannot be instantiated.

**7. What is Object?**

* Object **is** a class variable **or** an instance of class.
* It can represent a person, a bank account or any item that a program can handle.
* When program is executed, the objects interact by sending message to one another.
* **For example,** if ‘customer ‘ and ‘account’ are two objects in a program, then the customer object may send message to account object requesting for a bank Object can interact without having to know the details of each other’s data or code.
* It is sufficient to know the type of message accepted and the type of response returned by the objects.
* **The two parts of the object:**

**Object= Data + Methods**

Or

**To say the same differently**

**An object has the responsibility to know and the responsibility to do.**

**8. What is Class?**

* Class is collection of data members and member functions.
* Created using keyword class.
* A class declaration defines new user defined data type that links code and data.
* A class is a logical abstraction but an object has physical existence.
* Object is instance of a class.
* A class can be considered as the blueprint or definition or a temple for an object and describes the properties of and behavior of that object ,but without any existence
* **For example,** in the case of car or laptop, there will be a blueprint or design created first and then the actual car or laptop will be built based on that.
* We do not actually buy these blue prints but the actual objects.
* The basic syntax for the class definition:

Class Class Name[extends

SuperClassName]

{

[Fields declaration]

[Methods declaration]

}

**9. What is encapsulation?**

* Encapsulation is the practice of including in an object everything it needs hidden from other object
* The internal state is usually not accessible by other objects.
* Encapsulation is placing the data and the functions that work on that data in the same place.
* It is also referred to as a data hiding.
* Encapsulation is defined as the wrapping up the data under a single unit.
* Increased security of data.
* It’s just like capsule in which we can store multiple things and consume them.

Class

{

Data members

+

Methods (behavior)

}

Class

Variables methods

* Let’s take an example of a car. In a car we have so many functionalities like drive (), stop (), set speed (number) etc.
* Which help us to control the car while driving; apart from this we have various properties of car like model, speed, engine, speed Limit etc.
* Which gives us various information about the car?
* All these functions and properties are bound in a single unit called car.

**10.What is inheritance?**

* The object of one class can acquire the properties of object of another class is called inheritance.
* A parent child relationship between classes allows sharing of the behavior o the parent class in to its child classes.
* One of the major benefits of object-oriented programming (oop) is this code sharing between classes through inheritance.
* Child class can add new behavior or override existing behavior from parent.
* Reusability- building a new components by utilizing existing components –is yet another important aspect of OO paradigm.
* **Sub class, class, child class:** Terms to describe the child in the relationship, which accepts functionality

from its parent.

Class Subclass Name extends SuperClassName

{

Fields declaration;

Methods declaration;

}

* The keyword “extends ”signifies that the properties of super class are extended to

The subclass.

* That means, subclass contains its own members as well of those of the super class.
* The kind of situation occurs when we want to enhances properties of existing class without actually modifying it.

**Syntax of inheritance**

**{**

**//new variables or methods here**

**}**

* **Super class, base, class, parent class**: terms to describe the parent in the relationship, which shares its functionality.
* **Extend, inherit, and derive:** Become a subclass of another class.
* **Types of Inheritance**
* Single inheritance
* Multiple inheritance
* Hierarchical inheritance
* Multi-level inheritance
* Hybrid inheritance
* **Single inheritance:** single inheritance is the simple inheritance of all, when a class extends another class (only one class) then we call it as single inheritance.
* The below diagram represents

**Class A**

The single inheritance in java where

Class B extends only one class A

Here class B will be one and only super class

**Class B**

* **Multiple Inheritance:** multiple inheritance is nothing

But one class extending more than one class.

* Multiple inheritances is basically not supported by many Object oriented programming languages such as java, small talk, c# etc…
* As the child class has to manage the dependency of more than one parent class.
* But you can achieve multiple inheritances in java using inheritance**.**

**Class B**

**Class A**

**Class C**

* **Multilevel Inheritance:** In Multilevel Inheritance a derived class will be inheriting a parent class and as well as the derived class act as the parent class to other class.
* As seen in the below in the diagram.
* Class B inherits the property of class A and again class B act as a parent for class B and class B parent for class c.

**Class A**

**Class B**

**Class C**

* **Hierarchical Inheritance:** In Hierarchical inheritance one parent class will be inherited by many sub classes.
* As per the below example class A will be inherited by class B, class c and class D.
* Class A will be acting as a parent class for class B, class C and class D.

**Class A**

**Class D**

**Class C**

**Class B**

* **Hybrid Inheritance**: hybrid inheritance is the combination of both single and multiple inheritances.
* For example, class A and class B extends class C & another class D extends class A hybrid inheritance, because it is a combination of single and hierarchical inheritance.

**A**

**D**

**C**

**B**

**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 object.
* The most important aspect of an object is its behavior (the things it can do).
* A behavior 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 function to the operators or functions is called polymorphism.
* The ability of objects to take on different forms or behave in different ways depending on the context in which they are used.
* Polymorphism is achieved through the interfaces, inheritance, and method overriding.
* **For example,** a person who plays various roles likes being a father, brother, friend, or son, husband, employee.
* So the same person possesses different behaviors in different situations.
* This is called polymorphism
* They adjust their behavior based on the role they are playing at the moment.
* **Polymorphism include two steps:**
* **Methods of overloading(compile time):**
* Overloading in polymorphism is a technique that allows for multiple functions or methods with the same name but different parameters.
* The compiler automatically calls the correct method by matching the parameters passed.
* The functions must have the same return type but different arguments.
* Overloading can occur in both base and derived classes.
* Other names overloading is compile time polymorphism, static and also early binding.
* **Sample code**

Public class Program

{

Public int Add(int num1, int num2)

{

Return (num1+num2);

}

Public int Add (int num1, int num2, int num3)

{

Return (num1+num2+num3);

}

Public float Add (float num1,float num2)

{

Return (num1 + num2);

}

Public string Add (string value 1, string value2)

{

Return (value 1 +” “+ value 2);

}

Static void Main (Args\_args)

{

Program objprogram=New

Program ();

Info (“Add with two int parameter:” +objprogram. Add (3, 2));

Info (“Add with three int parameter:”

+objprogram. Add (3, 2,8));

Info (“Add with two float parameter:”

+objprogram. Add (3 f, 22 f));

Info (“Add with two string parameter:”

+objprogram. Add (“hello”, “world”));

}

}

In the above example, you can see that

There are four methods with the same name but the type of parameters is different. When you call Add(4,5), compiler automatically calls the method which has two integer parameters and when you call Add (“hello”,” world”), compiler calls the method which has two string parameters. So basically in method overloading compiler checks which method should be called at the time of compilation.

* **Methods of overriding(run time):**
* Method of overriding is a type of polymorphism
* It has several names like Run time polymorphism or Dynamic polymorphism and sometimes it is called late binding.
* Methods overriding means having two

Methods with the same name and same signatures {parameters}.

* One should be in a base class and another method should be in derived class(child class)
* You can override the functionality of a base class method to create the same name method with the same signature in a derived class.
* You can achieve method overriding using inheritance.
* Virtual and override keywords are used to achieve method overriding.

Class Base Class

{

Public virtual void Add (int num1, int num2)

{

//your logic will be here

}

}

Class Child Class: Base Class

{

Public override void Add (int num1, int num2)

{

// your logic will be here

;

}

}

Class Program

{

Static void Main (Args\_args)

{

BaseClass baseclassObj;

BaseClassObj = new Base Class();

info (“Base class method Add:” +

baseclassObj. Add (-3, 8));

baseclassObj=new ChildClass();

Info (“Child class method Add:” +

baseClassObj.Add (-2,2));

}

}

* In the above example, I have created two same name methods in the Base Class as well as in the child class.
* When you call the BaseClass Add method with less than zero value as parameters then it adds successfully.
* But when you call the child class Add method with less than zero value then it checks for a negative value.
* And the passing values are negative then it asks for a new value.
* So, here it is clear that we can modify the base class methods in derived class.

**12. Write SDLC phases with basic introduction?**

* The software Development Life Cycle (SDLC) is a process

Used by software development organizations to plan, design, develop, test, deploy, and maintain software applications.

* The lifecycle defines a method for improving the quality of software and the all around.

* **Requirement gathering:**
* In this stage, all the requirements for the target software are specified.
* These requirements get approval from customers, market analysis, and stake holders.
* This is fulfilled by utilizing SRS (Software requirements Specification).
* This is a sort of document that specifies all those things that need to be defined and created during the entire project cycle.
* **Planning and Analysis:**
* Planning is crucial step in everything, Justas in software development.
* In this same stage, requirements analysis is also performed by the developers of the organization.
* This is attained from customer inputs, and sales department /market surveys.
* The information from this analysis forms the building blocks o a basic project.
* The quality of the project is a result of planning.
* Thus, in this stage, the basic project is designed with all the available information.

**3. Designing Architecture:**

* **SRS** is a reference for software designers to come up with the best architecture for the software .
* Hence , with the requirements defined in SRS,

Multiple designs for the product architecture are present in the Design Document Specification (DDS).

* This DDS is assessed by market analysis and stakeholders
* After evaluating all the possible factors, the most practical and logical design is chosen for development.

**4. Implementation:**

* At this stage, the fundamental developing of the product starts.
* For this developers use a specific programming code as per the design in the DDS.
* Hence, it is important for the coders to follow the protocols set by the association.
* Conventional programming tools like compilers, interpreters, debuggers, etc. Are also put into use at this stage.
* Some popular languages like c/c++, python, java, etc. are put into use as per the software regulations.

1. **Testing and Integration:**

* After the development of product, testing of the software is necessary to ensure its smooth execution.
* Although, minimal testing is conducted at every stage of SDLC.
* Therefore, at this stage, all the probable flaws are tracked, fixed, and retested.
* This ensures that the product confronts the quality requirements of SRS.
* 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

**6. Maintenance phase:**

* After detailed testing, the conclusive product is released in phases as per the organization’s strategy.
* Then, it is tested in a real industrial environment.
* It is important to ensure its smooth performance.
* If it performs well, the organization sends out the product as a whole.
* After retrieving beneficial feedback, the company releases it as it is or with auxiliary improvements to make it further helpful for the customers.
* However, this alone is not enough.
* Therefore, along with the deployment, the product’s supervision.

**13. Explain the phases of waterfall model?**

* The waterfall model is a software development model used in the context of complex projects, typically in the field of information technology.
* It is characterized by a structured, sequential approach to project management and software development.
* This model is useful in situations where the project requirements are well defined and the project goals are clear.
* It is often used for large scale projects with long timelines, where there is little room for error and the project stakeholders need to have a high level of confidence in the outcome.
* **Features:**

1. Sequential approach: The model involves a sequential approach to software development, where each phase of the project is completed before moving on to next one.
2. Document –Driven: The model depends on documentation to ensure that the project is well defined and the project team is working towards a clear set of goals.
3. Quality Control: The model places a high emphasis on quality control and testing at each phase of the project, to ensure that the final product meets the expectations of the stakeholders.
4. Rigorous planning: The model involves a careful planning process, where the project scope, timelines, and deliverables are carefully defined and monitored throughout the project lifecycle.

Overall the waterfall model is used in situations where there is a need for a highly structured and systematic approach to software development.

* **Phases of waterfall Model:**
  1. **Requirements:** The first phase involves gathering requirements from stakeholders and analyzing them to understand the scope of the project.

**2. Analysis:** This phase starts with the requirement document delivered by the requirement phase and maps the requirement in to architecture. The architecture defines the components, their interfaces and behaviors .This phase represents the “how” phase.

**3. Design:** once the requirements are understood, the design phase begins. This involves creating a detailed design document outlines the software architecture, user interface, and system components.

**4. Development:** The development phase includes implementation involves coding the software based on the design specifications. This phase also includes unit testing to ensure that each component of the software is working as expected.

**5. Testing:** In the testing phase, the software is tested as a whole to ensure that it meets the requirements and is free from defects.

**6. Deployment:** once the software has been tested and approved, it is deployed to the production environment.

**7. Maintenance:** The final phase of waterfall model is maintenance, which involves fixing any issues that arise after the software has been deployed and ensuring that it continuous to meet the requirements over time.

**Waterfall model**

Requirement Gathering

Analysis

Design

Implementation

Testing

Eploy ment

Maintenance

**14. Write the phases of spiral model?**

* Spiral model is one of the most important SDLC models.
* This model is combination of waterfall model and iterative model.
* It provides support for risk handling. The spiral model was first proposed by Barry Boehm.
* That model provides a systematic and iterative approach to software. In its diagrammatic representation, looks like a spiral with many loops.
* The exact number of loops of the spiral is unknown and can vary from project to project.
* Each loop of the spiral is called a phase of the software development cycle.
* It is based on the idea of the spiral with each iteration of the spiral representing a complete software development cycle, from requirements gathering and analysis, design implementation, testing and maintenance.
* The model is a risk –driven model.

**Phases of the spiral model:**

1. **Objectives Defined:** In first phase of the spiral model we clarify what the project aims to achieve, including functional and non functional requirements.
2. **Risk Analysis:** In the risk analysis phase risks associated with the project are identified and evaluated.
3. **Engineering:** In the engineering phase, the software is developed based on the requirements gathered in the previous iteration.
4. **Evaluation:**  In the evaluation phase, the software is evaluated to determine if it meets the customer‘s requirements and if it is of high quality.
5. **Planning:** The next iteration of spiral begins with anew planning phase, based on the result of the evaluation.

**15. Write agile manifesto principles?**

* The agile manifesto is a set of foundational principles for agile software development that emphasizes flexibility, collaboration, and customer satisfaction.
* **12 principles of agile manifesto:**

**1. Customer satisfaction through early and continuous Delivery:** This principle concentrates on the importance of customer satisfaction by providing information to customers early on time and also with consistency throughout the development process.

**2. Welcome changing requirements, Even Late in Development:** Agile processes tackle change for the customer’s competitive advantage. Even late in development, changes in requirements are welcomed to ensure the delivered software meets the evolving requirements of the customer.

**3. Deliver Working Software Frequently:**

This principle encourages the regular release of functional software increments in short iterations. This enables faster feedback and adaption to changing requirements.

**4. Collaboration between Business Stakeholders and Developers:**

This says the businesspeople and developers must work together the daily throughout the project .there should be communication and collaboration between stakeholders and the development team regularly. This is crucial for understanding and prioritizing requirements effectively.

**5. Build projects around motivated individuals:** This promotes in giving developers the environment and support they need and trusts them to complete the job successfully. Motivated and empowered individuals are more likely to produce work with quality and make valuable contributions to the project.

1. **Face to Face communication is the Most Effective:**

This is the most effective method of discussion and conveying information. This principle depicts the importance of direct interaction which helps minimize misunderstandings, and hence effective communication is archived.

1. **Working Software Is the Primary Measures of Progress:** This principle emphasizes delivering functional and working software as the primary metric for project advancement .It encourages teams to prioritize the continuous delivery of valuable features, so it ensures that good progress is consistently achieved throughout the process.
2. **Maintain a sustainable Pace of work:**

Agile promotes sustainable development .all people involved: The sponsors , developers, and users should be able to maintain a constant pace indefinitely. This helps in avoiding burnout and ensures long-term project success.

**9. Continuous Attention to Technical Excellence and Good design:** This principle is on the importance of maintaining high standards of technical craft and design, so it ensures the long –term ability in maintenance and adaptability of the software.

**10. Simplicity –the Art of Maximizing the Amount of Work Not Done:** The simplicity essential. The objective here is concentrate on the most valuable features and tasks and avoids unnecessary complexity as the art of maximizing the amount of work not done is crucial.

**11. Self-Organizing Teams:** self- organizing teams providing the best architectures, requirements, and designs. These help in empowering teams to make decisions and organize to optimize efficiency and creativity.

**12. Regular Reflection on Team Effectiveness:** This makes the team reflect on how to become more effective at regular intervals and then adjust accordingly

**16. Explain the 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 iterations 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 best suit the project requirements.

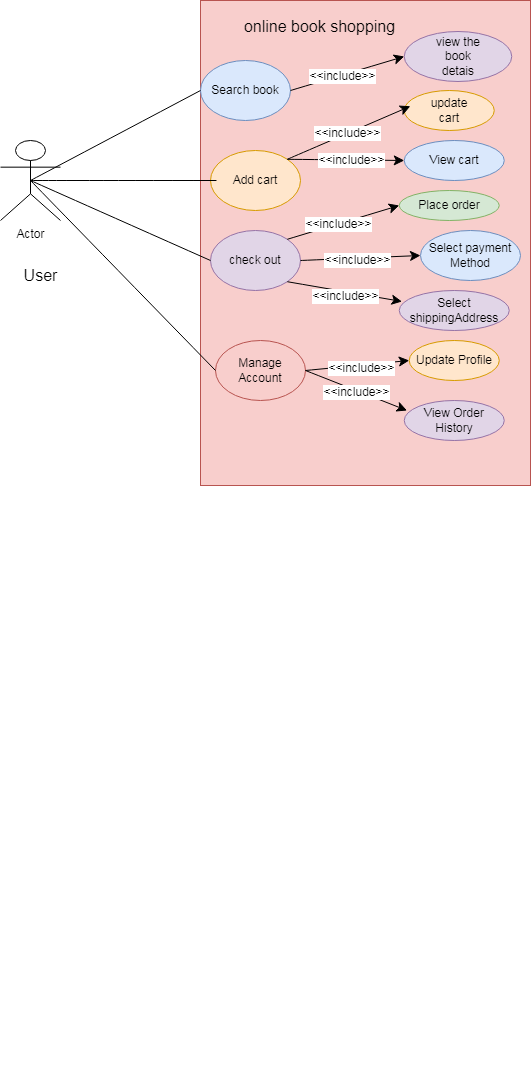
* **Pros of Agile methodology:**

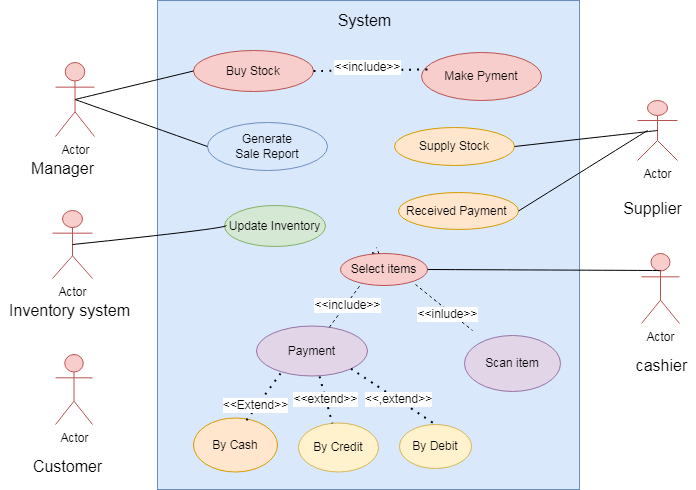
1. Is very a realistic approach to software development.
2. Promotes teamwork and cross training.
3. Functionality can be developed rapidly and demonstrated.
4. Resources requirements are minimum.
5. Suitable for fixed or changing requirements.
6. Minimal rules, document easily employed.
7. Easy to manage.
8. Little or no planning required.
9. Gives flexibility to developer.

* **Cons of Agile methodology:**

1. Not suitable for handling complex dependencies.
2. More risk of sustainability, Maintainability and extensibility.
3. An overall plan, an agile leader and agile PM practice is a must without which it will not work.
4. Depends heavily on customer interaction, so if customer is not clear, team can be driven in the wrong direction.
5. There is very high individual dependency, since there is minimum documentation generated.
6. Transfer of technology to new team members maybe quite challenging due to lack of documentation.

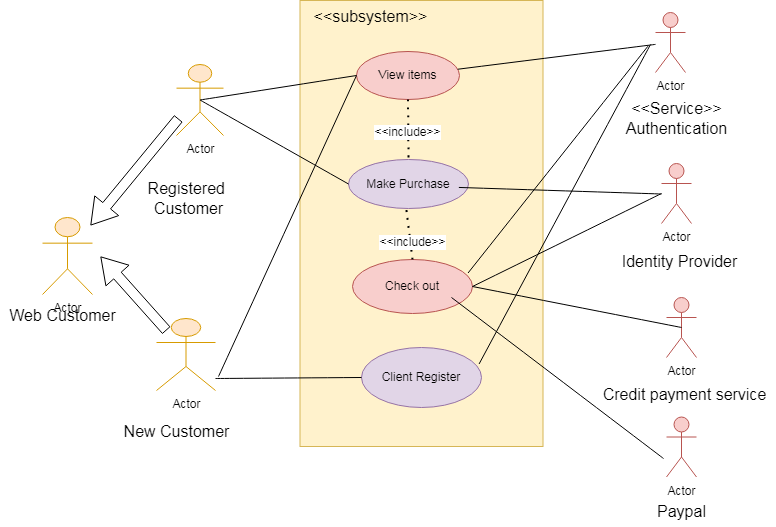
**17. Draw Use case diagram for online book shopping:**

****

****

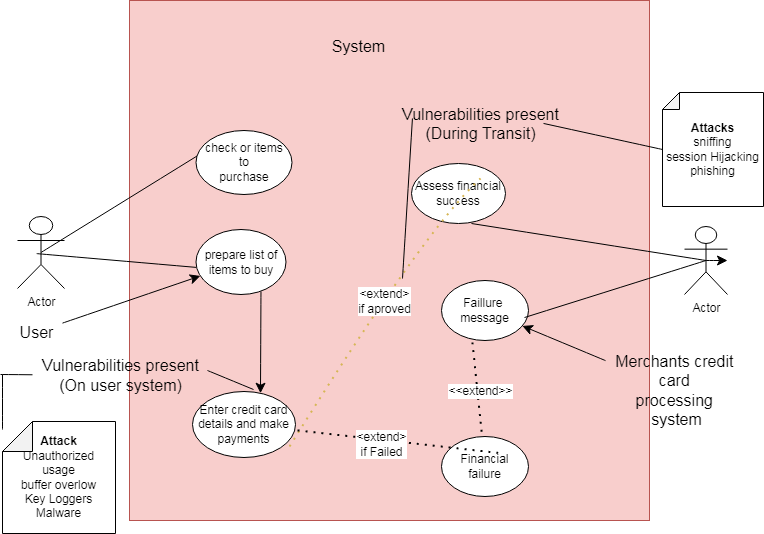
**Fig:**

**18. Use case diagram for online shopping product using payment gateway system:**

****

**19. Fig: Use case diagram for online shopping product using COD.**

**20. Draw the use case diagram for online Bill payment method?**

****

Maintenance and Deployment

Testing

Implementation