1:What is SDLC

A Software Development Life Cycle is essentially a series of steps,or phases, that provide a model for the development and life cycle management of an application or piece of software.

2:What is software testing?

SoftwareTestingisaprocessusedtoidentifythecorrectness,Completeness,andqualityofdevelopedcomputersoftware.

3:What is an agile methodology?

The Agile methodology is a way to manage a project by breaking it up into several phases

4:What is SRS

A Software Requirements Specification (SRS) incomplete description of the behavior of the system one developed

5:What is oops

Identifying objects and assigning responsibilities to these objects.

6:Write Basic Concepts of oops

There are four fundamental concepts of Object-oriented programming – Inheritance, Encapsulation, Polymorphism, and Data abstraction. It is very important to know about all of these in order to understand OOPs.

7:What is object

Anobjectrepresentsanindividual,identifiable item, unit, or entity,either alor abstract,with a well-defined roles the problem domain. An"object"is anything to which concept applies.

8:What is class

A class Represents Abstraction of the object and abstract the property and behavior of that object

9:What is encapsulation

Encapsulationisthepracticeofincludinginanobjecteverythingit needshiddenfrom other objects. The internal state is usually not accessible by other objects.

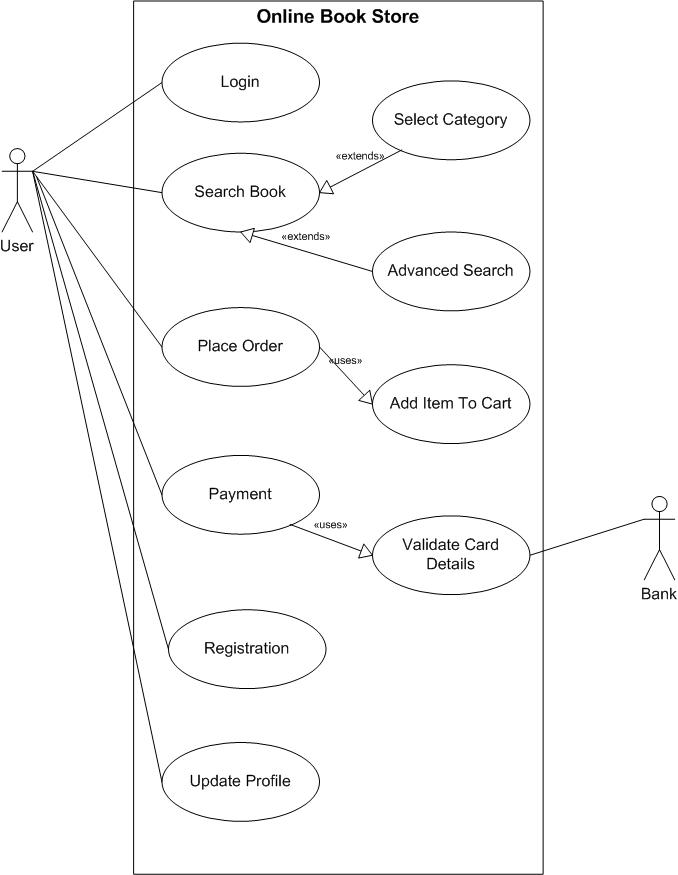
10:What is inheritance

Inheritance means that one class inherits the characteristics of another class. This is also called a“is a”relationship.

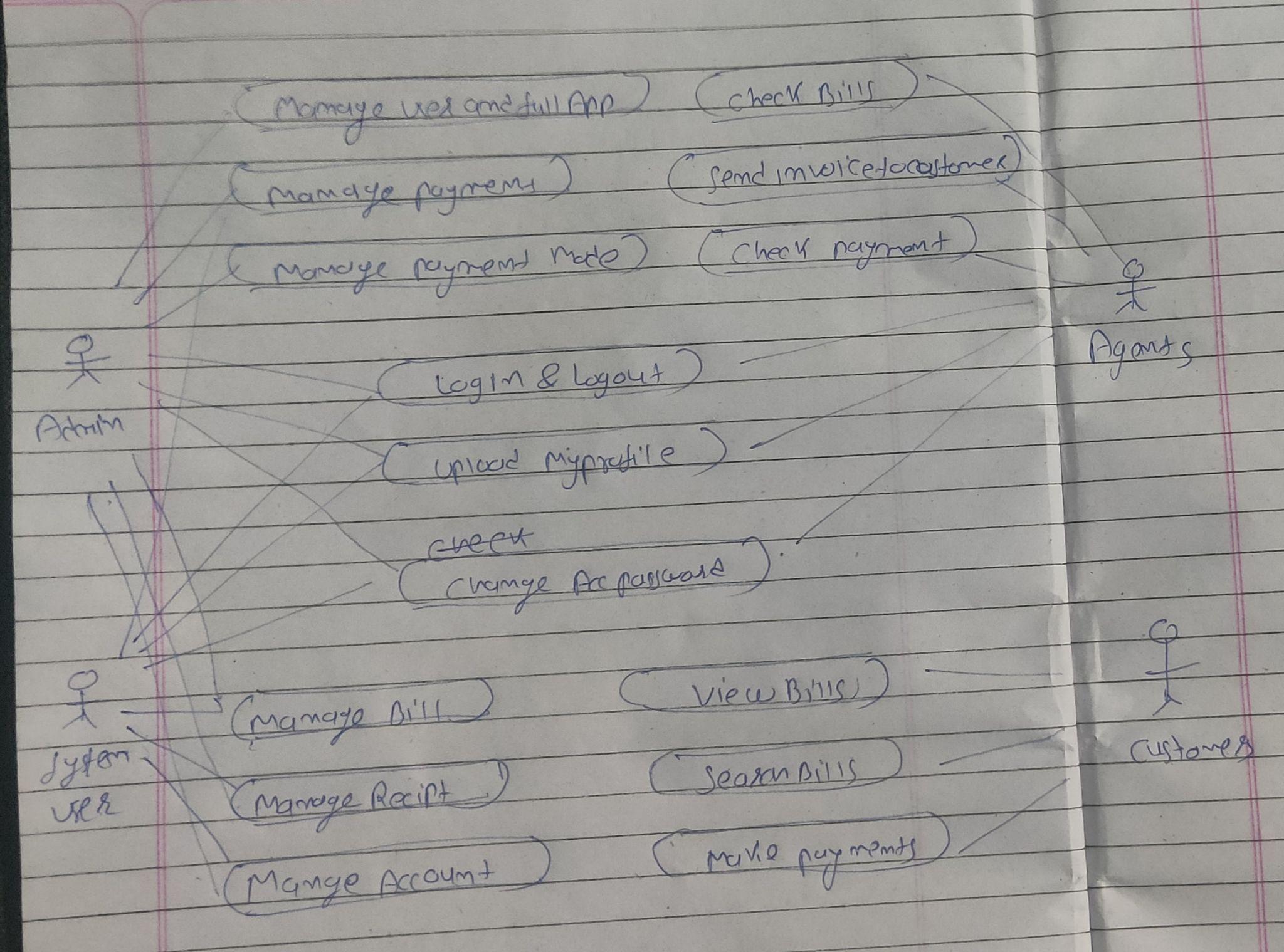
11:What is polymorphism

In programming language theory and type theory, polymorphism is the provision of a single interface to entities of different types or the use of a single symbol to represent multiple different types.

12:Draw Usecase on Online book shopping



13:Draw Usecase on online bill payment system (paytm)



14:Write SDLC phases with basic introduction

1. Planning

In the Planning phase, project leaders evaluate the terms of the project. This includes calculating labor and material costs, creating a timetable with target goals, and creating the project’s teams and leadership structure.

Planning can also include feedback from stakeholders. Stakeholders are anyone who stands to benefit from the application. Try to get feedback from potential customers, developers, subject matter experts, and sales reps.

Planning should clearly define the scope and purpose of the application. It plots the course and provisions the team to effectively create the software. It also sets boundaries to help keep the project from expanding or shifting from its original purpose.

2. Define Requirements

Defining requirements is considered part of planning to determine what the application is supposed to do and its requirements. For example, a social media application would require the ability to connect with a friend. An inventory program might require a search feature.

Requirements also include defining the resources needed to build the project. For example, a team might develop software to control a custom manufacturing machine. The machine is a requirement in the process.

3. Design and Prototyping

The Design phase models the way a software application will work. Some aspects of the design include:

Architecture – Specifies programming language, industry practices, overall design, and use of any templates or boilerplate

User Interface – Defines the ways customers interact with the software, and how the software responds to input

Platforms – Defines the platforms on which the software will run, such as Apple, Android, Windows version, Linux, or even gaming consoles

Programming – Not just the programming language, but including methods of solving problems and performing tasks in the application

Communications – Defines the methods that the application can communicate with other assets, such as a central server or other instances of the application

Security – Defines the measures taken to secure the application, and may include SSL traffic encryption, password protection, and secure storage of user credentials

Prototyping can be a part of the Design phase. A prototype is like one of the early versions of software in the Iterative software development model. It demonstrates a basic idea of how the application looks and works. This “hands-on” design can be shown to stakeholders. Use feedback to improve the application. It’s less expensive to change the Prototype phase than to rewrite code to make a change in the Development phase.

4. Software Development

This is the actual writing of the program. A small project might be written by a single developer, while a large project might be broken up and worked by several teams. Use an Access Control or Source Code Management application in this phase. These systems help developers track changes to the code. They also help ensure compatibility between different team projects and to make sure target goals are being met.

The coding process includes many other tasks. Many developers need to brush up on skills or work as a team. Finding and fixing errors and glitches is critical. Tasks often hold up the development process, such as waiting for test results or compiling code so an application can run. SDLC can anticipate these delays so that developers can be tasked with other duties.

Software developers appreciate instructions and explanations. Documentation can be a formal process, including wiring a user guide for the application. It can also be informal, like comments in the source code that explain why a developer used a certain procedure. Even companies that strive to create software that’s easy and intuitive benefit from the documentation.

Documentation can be a quick guided tour of the application’s basic features that display on the first launch. It can be video tutorials for complex tasks. Written documentation like user guides, troubleshooting guides, and FAQ’s help users solve problems or technical questions.

5. Testing

It’s critical to test an application before making it available to users. Much of the testing can be automated, like security testing[.](https://phoenixnap.com/blog/automated-security-testing-best-practices) Other testing can only be done in a specific environment – consider creating a simulated production environment for complex deployments. Testing should ensure that each function works correctly. Different parts of the application should also be tested to work seamlessly together—performance test, to reduce any hangs or lags in processing. The testing phase helps reduce the number of bugs and glitches that users encounter. This leads to a higher user satisfaction and a better usage rate.

6. Deployment

In the deployment phase, the application is made available to users. Many companies prefer to automate the deployment phase. This can be as simple as a payment portal and download link on the company website. It could also be downloading an application on a smartphone.

Deployment can also be complex. Upgrading a company-wide database to a newly-developed application is one example. Because there are several other systems used by the database, integrating the upgrade can take more time and effort.

7. Operations and Maintenance

At this point, the development cycle is almost finished. The application is done and being used in the field. The Operation and Maintenance phase is still important, though. In this phase, users discover bugs that weren’t found during testing. These errors need to be resolved, which can spawn new development cycles.

In addition to bug fixes, models like Iterative development plan additional features in future releases. For each new release, a new Development Cycle can be launched.

15:Explain Phases of the waterfall model

* Requirement Gathering and analysis − All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
* System Design − The requirement specifications from the first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
* Implementation − With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
* Integration and Testing − All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
* Deployment of system − Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
* Maintenance − There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

16:Write phases of spiral model

Spiral Model Phases

It has four stages or phases: The planning of objectives, risk analysis, engineering or development, and finally review. A project passes through all these stages repeatedly and the phases are known as a Spiral in the model.

Determine objectives and find alternate solutions – This phase includes requirement gathering and analysis. Based on the requirements, objectives are defined and different alternate solutions are proposed.

Risk Analysis and resolving – In this quadrant, all the proposed solutions are analyzed and any potential risk is identified, analyzed, and resolved.

Develop and test: This phase includes the actual implementation of the different features. All the implemented features are then verified with thorough testing.

Review and planning of the next phase – In this phase, the software is evaluated by the customer. It also includes risk identification and monitoring like cost overrun or schedule slippage and after that planning of the next phase is started.

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

**Pros:** Is Very Realistic Approach To Software Development

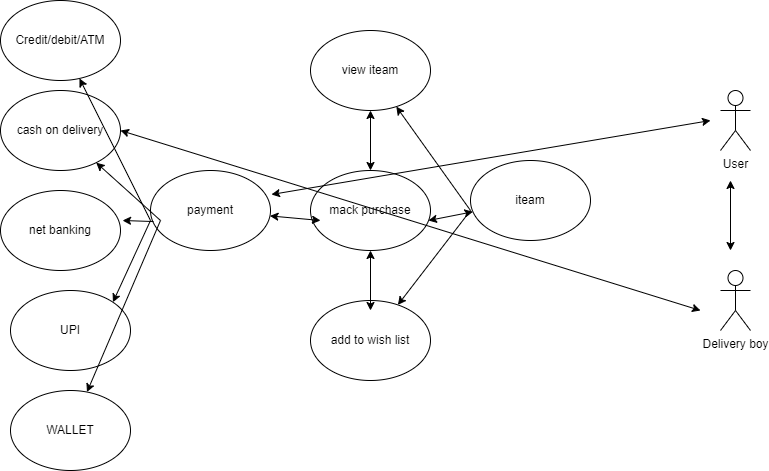
Promotesteamwork and cross training?

Functionality can be developed rapidly and demonstrated.

Resource Requirements Are Minimum. Suitable For Fixed Or changing requirements Deliver Early Partial Working Solutions. Good Model For Environments That Change Steadily. Minimal Rules,documentation easily employed. Enablesconcurrentdevelopmentanddeliverywithinanoverallplanned context LittleornoplanningrequiredEasytomanageGivesflexibilitytodevelopers

**Cons:** Not Suitable For Handling Complex Dependencies. More Risk Of Sustainability,maintainability and extensibility. Anoverallplan,anagileleaderandagilePMpracticeis amustwithout whichitwillnotwork. Strictdeliverymanagementdictatesthescope,functionalitytobe delivered,andadjustmentstomeetthedeadlines. Dependsheavilyoncustomerinteraction,soifcustomeris not clear,teamcanbedriveninthe wrong direction. Thereisveryhigh Individual Dependency,since thereisminimum documentation nerated. Transferoftechnologytonewteammembersmaybe quite challenging duetolack documentation use – case

18:Draw use case on Online shopping product using COD.



19:Draw use case on Online shopping product using payment gateway

