1. 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. There are a number of different development models.

A Software Development Life Cycle 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.

->The methodology within the SDLC process can vary across industries and organizations but standards such as ISO/IEC 12207 represent processes that establish a lifecycle for software, and provide a mode for the development, acquisition, and configuration of software systems.

2.what is testing?

->Testing is the process of evaluating a system or its component(s) with the intent to find that whether it satisfies the specified requirements or not.

->testing is executing a system in order to identify any gaps, errors or missing requirements in contrary to the actual desire or requirements

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

.

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.

-> 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.

4.what is SRS?

-> A software requirements specification (SRS) is a complete description of the behavior 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.

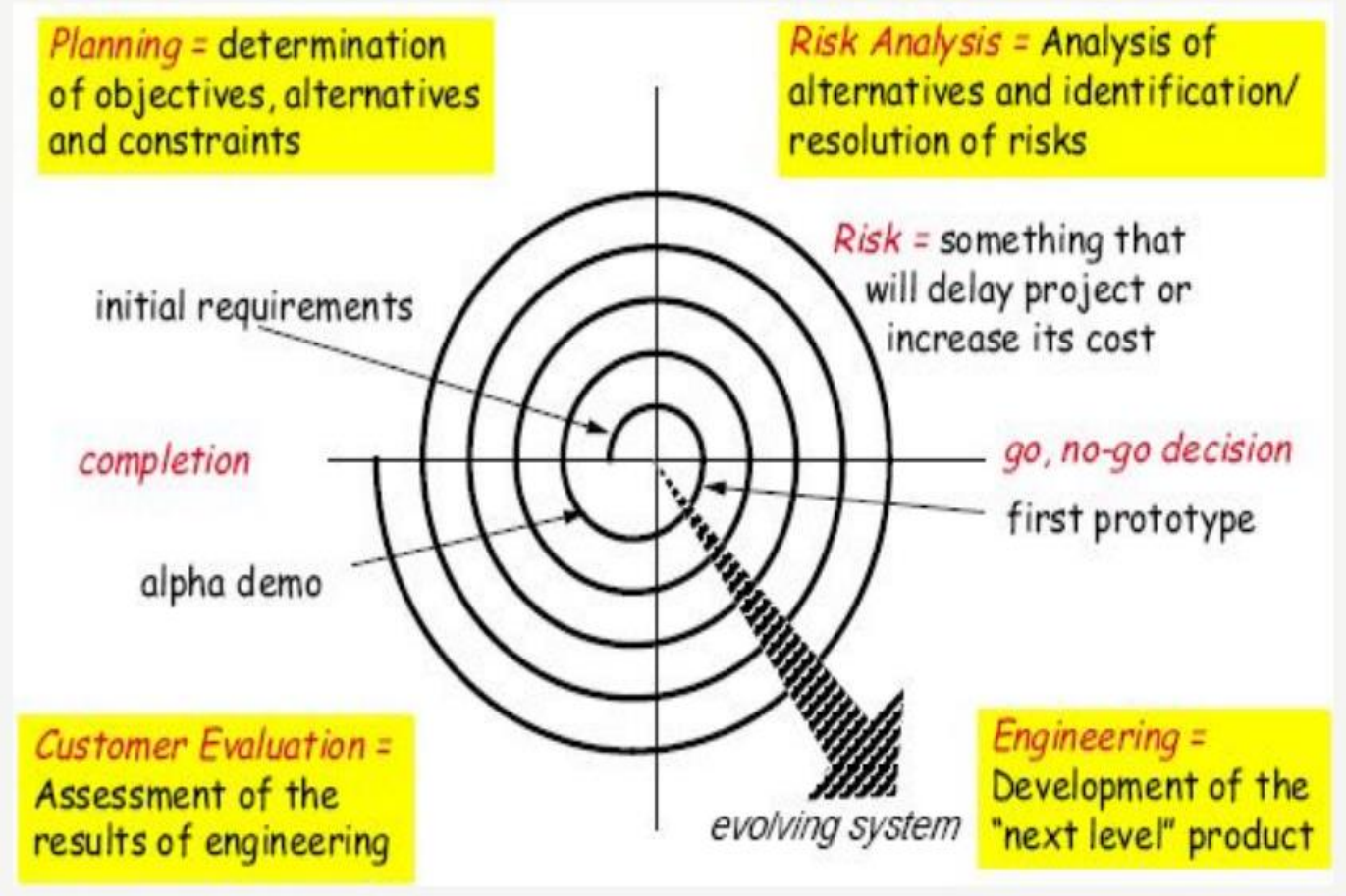
5. Write SDLC phases with basic introduction.

Requirements Collection/Gathering	Establish Customer Needs
Analysis	Model And Specify the requirements- “What”
Design	Model And Specify a Solution – “Whsoftwarey”
Implementation	software construct a solution in software
Testing	Validate the solution against the requirements
deployment	
Maintenance	Repair defects and adapt the solution to the new requirements

6. 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

The Classical Software lifecycle models the software development as a step-by-step waterfall between the various development phases.



7. 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.

->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. 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 documentation.

->prose

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.

➞Diagram:-use case