Software Testing Assignment

Module–1

1. What is SDLC ?

**Ans.** : 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.

1. What is software testing ?

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

1. What is agile methodology ?

**Ans.** :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.

1. What is SRS ?

**Ans.** : 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. Use cases are also known as functional requirements. In addition to use cases, the SRS also contains non-functional (or supplementary) requirements.

1. What is oops ?

**Ans.** : 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. Object-oriented programming has a web of interacting objects, each house-keeping its own state.

1. Write Basic Concepts of oops ?

**Ans.** : Object, Class, Encapsulation, Inheritance, Polymorphism(i. Overriding, ii. Overloading), Abstraction.

1. What is object ?

**Ans.** : An object represents an individual, identifiable item, unit, or entity, either real or abstract, with a well-defined role in the problem domain. An "object" is anything to which a concept applies. This is the basic unit of object oriented programming(OOP).

1. What is class ?

**Ans.** : A class represents an abstraction of the object and abstracts the properties and behavior of that object. Class can be considered as the blueprint or definition or a template for an object and describes the properties and behavior of that object, but without any actual existence.

1. What is encapsulation ?

**Ans.** : Encapsulation is placing the data and the functions that work on that data in the same place. Encapsulation is the practice of including in an object everything it needs hidden from other objects. Encapsulate in plain English means to enclose or be enclosed in or as if in a capsule. In Java, a class is the capsule (or unit).

1. What is inheritance ?

**Ans.** : Inheritance means that one class inherits the characteristics of another class. This is a very important concept of object-oriented programming since this feature helps to reduce the code size. Inheritance describes the relationship between two classes. A class can get some of its characteristics from a parent class and then add unique features of its own.

1. What is polymorphism ?

**Ans.** : 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 use an operator or function in different ways in other words giving different meaning or functions to the operators or functions is called polymorphism.

1. Draw Usecase on Online book shopping

**Ans.** : <https://drive.google.com/file/d/1qJp9hTOVa2V0OXM3EIbIxi9UzuzE_cCB/view?usp=sharing>

1. Draw Usecase on online bill payment system (paytm).

**Ans.** : <https://drive.google.com/file/d/1vne9v1WctgL0G0nF_COwIPqEHiMFTgOw/view?usp=sharing>

1. Write SDLC phases with basic introduction.

**Ans.** **: (1)** Requirements Collection/Gathering

Requirements definitions usually consist of natural language, supplemented by (e.g., UML) diagrams and tables. Although requirements may be documented in written form, they may be incomplete, unambiguous, or even incorrect. Requirements will Change! Inadequately captured or expressed in the first place User and business needs change during the project Validation is needed throughout the software lifecycle, not only when the “final system” is delivered.

**(2)** Analysis

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. Ideally, this document states in a clear and precise fashion what is to be built.

**(3)** Design

The Design team can now expand upon the information established in the requirement document. The requirement document must guide this decision process. Analyzing the trade-offs of necessary complexity allows for many things to remain simple which, in turn, will eventually lead to a higher quality product. Design Architecture Document, Implementation Plan.

**(4)** Implementation

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.

**(5)** Testing

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. The testing phase is a separate phase which is performed by a different team after the implementation is completed.

**(6)** Maintenance

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.

1. Explain Phases of the waterfall model.

**Ans.** : Requirements Analysis,System Design**,** Implementation**,** Integration**,** Testing

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.

1. Write phases of spiral model.

**Ans.** : (1) Planning = determination of objectives, alternatives and constraints.

(2) Risk Analysis = Analysis of alternatives and identifaction/resolution of risks.

(3) Customer Evaluation = Assessment of the results of engineering.

(4) Engineering = Development of the “next level” product.

**17.** Write agile manifesto principles.

**Ans.** : Satisfy the customer through early and continuous delivery of valuable software. Welcome changing requirements, even late in development. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale. Business people and developers must work together daily throughout the project.

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

**Ans. :** 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 planned context.
* 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.

1. Draw usecase on Online shopping product using COD.

**Ans.** <https://drive.google.com/file/d/1qvhJHM-KWhW4C6SCti04wRF8p78RKuGb/view?usp=sharing>

1. Draw usecase on Online shopping product using payment gateway.

**Ans.** : <https://drive.google.com/file/d/1Slg8hx0cbNFOo5pmu8NrPxybKt313J54/view?usp=sharing>