***Software Testing Assignment Module–1(Fundamental)***

1. **What is SDLC?**

* SDLC (Software development life cycle) 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?**

* SoftwareTesting is a process used to identify the correctness, Completeness, and quality of developed computer software.

1. **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.

1. **What is SRS?**

* A software requirements specification (SRS) is a complete description of the behaviour of the system to be developed.
* Non-functional requirements are requirements which impose constraints on the design or implementation (such as performance requirements, quality standards, or design constraints).

1. **What is oops?**

* identifying object and assigning responsibility to object.
* 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 is derived from abstract data type.
* Object-oriented programming has a web of interacting objects, each house- keeping its own state.
* Objects of a program interact by sending messages to each other.

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

* Object
* Class
* Encapsulation
* Inheritance
* Polymorphism
* Abstraction

1. **What is object?**

* An object represents an individual, identifiable item, unit, or entity, either real or abstract, with a well-defined role in the problem domain.
* Instances of an class.

: to create an memory of an class

: to access the whole properties of an class except private.

1. **What is class?**

* A class represents an abstraction of the object and abstracts the properties and behaviour of that object.
* is an collection of data member(variable) and member function (process

Method) with its behaviour.

1. **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.
* Data hiding: Wrapping up of data into single unit private your data member or member function.

1. **What is inheritance?**

* Inheritance - Properties of parent class extends into child class.

: Main purpose is responsibility, extensibility

There are mainly 5 types

1. Single
2. Multilevel
3. Hierarchical
4. Multiple: java does not support
5. Hybrid: java does not support
6. **What is polymorphism?**

* Polymorphism - Ability to take one name having different forms (Many forms or multiple forms)

: There are mainly 2 types

1. Compile Time (Method over loading)
2. Run Time (Method Overriding)
3. **Draw Use case on Online book shopping?**

* GO TO THE APPLICATION
* ENTER LOGIN AND PASSWORD
* SELECT AREA PIN CODE
* SELECT BOOK
* CLICK ON BUY NOW
* ADDING ADDRESS DETAILS
* SELECT PAYMENT OPTION (NET BANKING)
* SELECT BANK
* REDIRECT TO BANK OFFICIAL LOGIN PAGE
* INPUT BANK LOGIN DETAIL
* PAYMENT PAGE
* OTP
* CONFIRMATION
* PAYMENT DONE PAGE

1. **Draw Use case on online bill payment system (Paytm)**

* GO TO THE PAYTM
* ENTER LOGIN AND PASSWORD
* SELECT STATE & ELECTRICITY BOARD
* ENTER COSTOMER SEVICE NUMBER
* CLICK ON QUICKPAY (BILL PAYMENT OPTION)
* SELECT PAYMENT OPTION (NET BANKING)
* SELECT BANK
* REDIRECT TO BANK OFFICIAL LOGIN PAGE
* INPUT BANK LOGIN DETAIL
* PAYMENT PAGE
* OTP
* CONFIRMATION
* PAYMENT DONE PAGE

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

* **Requirement Gathering –** Establish customer needs
* **Analysis –** model and specify the requirements (What)
* **Design-** model and specify the 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**

* Usage scenarios
* Although requirements may be documented in written form, they may Be incomplete, unambiguous, or even incorrect.
* Requirements will Change
* User and business needs change during the project
* Build constant feedback into the project plan.
* Plan for change
* Functional and Non-Functional

* **Three types of problems can arise:**
* **Lack of clarity:** It is hard to write documents that are both precise and easy to-read.
* **Requirements confusion:** Functional and Non-functional Requirements tend to be intertwined.
* **Requirements Amalgamation:** Several different requirements May be expressed together.

**Types of Requirements:**

* Functional Requirements: describe system services or Functions.
* Non-Functional Requirements: are constraints on the system or the development process.

# Analysis Phase

* The analysis phase defines the requirements of the system, independent of how these requirements will be accomplished.
* 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.
* The requirement documentaries to capture the requirements from the customer's perspective by defining goals.
* This phase starts with the requirement document delivered by the requirement phase and maps the requirements into architecture.
* The design may include the usage of existing components.

**Design Phase:**

* Implementation Plan
* Critical Priority Analysis
* Performance Analysis
* Test Plan
* The Design team can now expand upon the information established in the requirement document.
* The requirement document must guide this decision process.

**Implementation Phase:**

* In the implementation phase, the team builds the components eitherdocument 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.
* For example, a component may be narrowly designed for this particular system, or the component may be made more general to satisfy a reusability guideline.
* Critical Error Removal.
* The implementation phase deals with issues of quality, performance, Baselines, libraries, and debugging.

**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.
* A customer satisfied with the 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
* The developing organization or team will have some mechanism to document and track defects and deficiencies.
* Configuration and version management
* Reengineering (redesigning and refactoring)
* Updating all analysis, design and user documentation
* Repeatable, automated tests enable evolution and Refactoring.

**Maintenance** is the process of changing a system after it has been deployed.

* **Corrective maintenance:** identifying and repairing defects
* **Adaptive maintenance:** adapting the existing solution to the new platforms.
* **Perfective Maintenance:** implementing the new on decides the utility and value of the software at a particular level of quality outweighs the impact of the known defects and deficiencies.

1. **Explain Phases of the waterfall model?**

**Waterfall model –** The software development as a step by step “waterfall” between the various development phases. Waterfall model is one way process and requirement must be FROZEN.

* **Requirement Gathering –** Establish customer needs
* **Analysis –** model and specify the requirements (What)
* **Design-** model and specify the 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

1. **Write phases of spiral model?**

* **Planning –** determination of objectives, alternatives and constraints.
* **Risk analysis –** analysis of alternatives and identification/resolutions of risks
* **Engineering –** development of the next level product
* **Customer evolution –** assessment of the results of engineering

1. **Write agile manifesto principles?**

* Individual interaction
* Working software
* Customer collaboration
* Respond to change

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

* Agile SDLC model was 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.

**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
* Little or no planning required Easy to manage Gives flexibility to developers

**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 use – case.

1. **Draw use case on Online shopping product using COD?**

* GO TO THE APPLICATION
* ENTER LOGIN AND PASSWORD
* SELECT AREA PIN CODE
* SELECT PRODUCT AND SIZE
* CLICK ON BUY NOW
* ADDING ADDRESS DETAILS
* SELECT PAYMENT OPTION (COD)
* CAPTCHA OR OTP
* ORDER CONFIRMATION

**20) Draw use case on Online shopping product using payment gateway?**

* GO TO THE APPLICATION
* ENTER LOGIN AND PASSWORD
* SELECT AREA PIN CODE
* SELECT PRODUCT AND SIZE
* CLICK ON BUY NOW
* ADDING ADDRESS DETAILS
* SELECT PAYMENT OPTION (DEBIT AND CREDIT CARD)
* SELECT BANK

FILL THE CARD DETAIL (CARD NUMBER, EXPIRY DATE, CVV)

* OTP
* CONFIRMATION PAGE