K. Gnaneswar
2200090109
CS-IT/24

ASE HOME ASSIGNMENT 2

1. What is known as SRS review? How is it conducted?

SRS Review (Software Requirements specification Review): An SRS review is a formal process in software engineering where the software requirements specification document, which outlines the functional and non-functional requirements of a software system, is evaluated and reviewed by relevant stakeholders. The goal of an SRS review is to ensure that the documented requirements are clear, complete, consistent, and feasible before proceding to the next stages of the software development process. This review typically involves developers, testers, business analysts, and other project stakeholders.

2. Describe reviews, walkthroughs, and inspections briefly.

Reviews :- A review is a collaborative process where the team members or stakeholders examine a software artifact, such as a design document or a piece of code, to identify defects, inconsistencies and areas for improvement. Reviews can be informal or formal, depending on the context and the level of rigor required.

Walk throughs :- A walkthrough is a more structured form of review where the author of the artifact guides others through it. The participants ask questions, provide feedback, and discuss potential issues. This helps to identify problems early and improve the quality of the artifact.

Inspections :- Inspections are a formal and rigorous type of review that involves a detailed examination of the software artifact with the goal of finding defects. Inspections follow a well-defined process and involve a checklist to ensure thorough examination.

3. Describe how software requirements are documented? State the importance of documentation.

Software Requirements Documentation and Importance :

Software requirements are documented in a Software Requirements Specification (SRS) document. This document outlines the functional and non-functional requirements of the software system, including its features, user interactions, system behavior, performance and constraints. proper documentation of requirements is crucial because it :

* provides a clear understanding of what the software should do and how it should behave.
* Acts as a foundation for design, development, testing and project management.
* Helps in communication between stakeholders, including developers, testers, clients and business analysts.
* Serves as a basis for validating whether the final product meets the desired functionally.

4. Differentiate functional and non-functional requirements.

Functional Requirements :- These specify what the software system should do. They describe the specific functions, features, and interactions the system must provide to its users. Examples include user authentication, data processing, and report generation.

Non-functional Requirements :- These specify how the system should perform its functions. They include qualities such as performance, security, scalability, usuability and reliability. Examples include response time, data encryption and user interface design.

5. What is extreme programming ? Describe the extreme programming process

Extreme Programming (XP): Extreme programming is an agile software development methodology that emphasizes close collaboration between developers, frequent releases, and a strong focus on customer satisfaction. It involves practices such as continues integration, test-driven development, pair programming and frequent communication with customers.

XP Process : The XP process involves several key practices including:

planning :- creating user stories and estimating their effort.

continuous Integration:- Integration code frequently to ensure early reliable functionality.

Pair programming :- Two programmers working together at one computer to improve code quality & knowledge sharing.

Small Releases : Releasing small, functional increments of the software frequently.

customer feedback : Regularly involving customers to provide feedback and guide development priorities.

Agility :- Agility in software development refers to the ability to respond quickly and effectively to changes in requirements, technology, or market conditions.

Agile Team :- An agile team is a cross-functional group of individuals working collaboratively on software development.

It typically includes developers, testers, designers and other roles necessary to deliver software.

Principles for Agile Software development (Agile Manifesto) :-
The Agile Manifesto outlines four core values and twelve principles that guide agile development. The values prioritize individuals and interactions, working software, customer collaboration and responding to change. The principles emphasize customer satisfaction, incremental development, continuous delivery and maintaining a sustainable pace.

7. Distinction between user stories and requirements.

User stories :- User stories are short, simple descriptions of a feature from the user's perspective. They are often written in a "As a [user], I want [feature] so that [benefit]" format. User stories are used in agile methodologies to capture user needs and requirements in a concise manner.

Requirements :- Requirements encompass both functional & non-functional aspects of the software. Requirements can be more formal and structured than user stories.

8. Describe the types of traceability in the software development.

Forward Traceability : This involves linking higher-level requirements to lower-level artifacts, ensuring that each req is addressed in the design, code & testing stages.

Backward traceability: This involves linking lower-level artifacts back to higher level req, demonstrating that each aspect of the software is derived from a specific req.

Bi directional traceability, this combines both forward & backward traceability, ensuring a complete link b/n req, design, implementation & testing stages.