**SE-ASSIGNMENT-2(BRIAN MUMBO)**

1. **Define Software Engineering**: It refers to implementing, testing, managing, and maintaining software systems.
2. **What is software engineering, and how does it differ from traditional programming? Software Development life cycle(SDLC)**

Software engineering is an engineering discipline that designs, develops, tests, and maintains software systems.  Traditional programming refers to writing code to solve specific problems without necessarily following a formalized process. software engineering encompasses a broader approach, emphasizing quality and novel elements, while traditional programming focuses on solving specific problems using established methods.

1. **Explain the various phases of the Software Development Life cycle. Provide a brief description of each phase. Agile vs. Waterfall Models:**

Planning & Analysis-Gather of business requirements from stakeholders. Evaluating feasibility, revenue potential, and end-user needs.

Design-Create architectural and system design.

Development-Write code based on design specifications.

Testing- Verify functionality, performance, and security.

Deployment-Release the software to production.

1. **Compare and contrast the Agile and Waterfall models of software development. What are the key differences, and in what scenarios might each be preferred? Requirements Engineering:**

Agile

Focus: Agile emphasizes flexibility, rapid iteration, and customer feedback. Work is divided into time-based Sprints (typically 1-4 weeks). Agile teams allocate resources to meet requirements and it Prioritizes delivering value quickly. Agile is ideal for smaller, dynamic projects.

Waterfall

It has Linear, well-defined stages with formal hand-offs where all requirements are defined upfront. Waterfall suits projects with stable, specific requirements.

Requirements engineering defines, documents, and maintains requirements in the engineering design process.

Activities involved

* Inception/Elicitation: Gather stakeholder needs and wants.
* Analysis/Negotiation: Identify requirements and resolve conflicts.
* System Modeling: Design and model the system.
* Specification: Document requirements formally.
* Validation: Ensure consistency and stakeholder alignment.

1. **Explain the concept of modularity in software design. How does it improve the maintainability and scalability of software systems? Testing in Software Engineering:**

* Modularity refers to breaking down a complex software system into smaller, loosely coupled modules. Modules handle specific functions or features and interact through clear interfaces. This is quite important since it makes it easier to understand and navigate which Simplifies updates and bug fixing.
* Modular systems can scale by adding or modifying modules. Maintainability is affected in the sense that Changes in one module don’t affect others.
* Software Testing-The Purpose is to Verify software functionality, identify defects, and ensure it meets requirements.

1. **Describe the different levels of software testing (unit testing, integration testing, system testing, acceptance testing). Why is testing crucial in software development? version control systems:**

* UNIT TESTING- Unit testing examines individual components (such as methods and functions) in isolation to identify bugs and errors.

The purpose is to Verify correctness at the smallest level, ensure individual units function as expected, and Detect issues early in the development process.

The techniques used are Statement Coverage Which ensures all code statements are tested, and Condition Coverage Which tests each conditional statement.

* INTEGRATION TESTING- integration testing checks if combined software components work together seamlessly.

The purpose is to Validate interactions between modules and Detect issues arising from component integration.

The techniques involved are Big Bang which Combines all components and tests. And Bottom-Up which tests lower-level components first.

* SYSTEM TESTING- System testing evaluates the entire software system against functional and non-functional requirements.

It focuses on Functional correctness, Performance, security, and reliability. It ensures the system meets stakeholder expectations and all components work together seamlessly.

* ACCEPTANCE TESTING- Acceptance testing validates if the software meets specified criteria for acceptance by stakeholders.
* There are 3 types of acceptance testing: User Acceptance Testing (UAT): In end-users validate functionality, Business Acceptance Testing: Where business requirements are met, and Contract Acceptance Testing: Where compliance with contractual terms.

It is important in ensuring customer satisfaction and validating adherence to requirements

* SOFTWARE TESTING IS CRUCIAL TO SOFTWARE DEVELOPMENT in that it saves money, and early debugging reduces cost. It ensures data protection and reliability. It also provides high-quality software to the user as well as a better user experience.

1. **What are the version control systems, and why are they important in software development? Give examples of popular version control systems and their features. Software project management:**

Version control (also known as source control) is the practice of tracking and managing changes to software code.

Types of Version Control Systems

* Local Version Control:

Changes are stored locally before being pushed to a single code version.

Retrieving changes can be challenging if local versions or the single code version become corrupted.

* Central Version Control:

Hosts different code versions in a centralized repository.

Users can access and push/pull changes.

Retrieval can be difficult if the repository becomes corrupted.

Popular Version Control Systems

* Git:

Features: Speed, flexibility, robustness.

Usage: Widely adopted in modern software development.

* Subversion (SVN):

Features: Centralized, tracks changes over time.

Usage: Common in enterprise environments.

Importance of version control systems

* Streamlined Release Management:

Different software releases are maintained, aligning with the release roadmap.

* Conflict Prevention:

Separate branches for different releases minimize code conflicts.

1. **Discuss the role of a software project manager. What are some key responsibilities and challenges faced in managing software projects? Software maintenance**

Software project managers play a crucial role in ensuring the success of software development projects.

Responsibilities

* Project Planning: Prepare project proposals, define scope, and create project plans.
* Resource Management: Allocate resources (people, time, budget) effectively.
* Risk Management: Identify and manage project risks.
* Communication: Liaise with stakeholders (clients, team members, management).

Challenges

* Scope Creep: Managing changes to project scope.
* Time Constraints: Balancing deadlines and quality.
* Technical Complexity: Handling intricate software requirements.
* Team Dynamics: Coordinating diverse teams.

1. **Define software maintenance and explain the different types of maintenance activities. why is maintenance an essential part of the software lifecycle? Ethical considerations in software Engineering:**

SOFTWARE MAINTENANCE refers to the ongoing process of modifying and updating software after its initial delivery.

* Corrective Maintenance:

Purpose: Resolves defects (bugs) identified during software usage. The goal is to Enhance system reliability and user satisfaction.

* Adaptive Maintenance:

Purpose: Accommodates changes in the software environment (e.g., new hardware, OS). The goal is to ensure compatibility and adaptability.

Importance of Maintenance:

* Longevity: Extends the software’s useful life.
* Reliability: Addresses defects promptly.
* Adaptability: Keeps pace with evolving technology.
* User Satisfaction: Ensures a positive user experience.

1. **What are some ethical issues that software engineers might face? How can software engineers ensure they adhere to ethical standards in their work?**

* Algorithmic Bias:

Ensuring fairness in AI algorithms. Mitigating biases related to race, gender, and other factors.

* Personal Data Collection:

Handling user data responsibly and Transparency about data usage.

* Weak Security Protection:

Prioritizing robust security practices and Safeguarding sensitive information.

To adhere to ethical standards:

* Continuous Learning: Stay informed about ethical guidelines.
* Collaborate: Discuss ethical dilemmas with colleagues.
* Reflect: Consider the broader impact of your work.