ACTIVITY - 11

Study and prepare report on testing tools.

1) Automated Testing Tools:
* selenium: A widely-used tool for automating web browser interactions. Supports multiple programming languages and browsers.
* Appium: An open-source tool for automating mobile applications across different platforms (iOS, Android).
* TestComplete: Offers automated UI testing for web, desktop, and mobile applications with support for scripting languages like JavaScript, python, and VBScript.

2) Load Testing Tools:
* JMeter: Apache JMeter is a Java based tool for load testing web applications, REST APIs, and more. It supports various protocols and can simulate high loads.
* Load Runner: A performance testing tool by Micro Focus, capable of testing applications under various loads and conditions
* Gatling: An open-source load testing tool based on Scala. it's known for its high performance and flexibility.

3) Security Testing Tools:
* OWASP ZAP: An open-source web application security scanner designed to find security vulnerabilities in web applications.

* Burp Suite: A comprehensive platform for web security testing, including scanning, crawling, and exploiting vulnerabilities.
* Acunetix: A web vulnerability scanner that automatically checks web applications for common security flaws.

4) Code Analysis Tools:
* Sonar Qube: An open-source platform for continuous inspection of code quality, security, and maintainability.
* Checkmarx: A static application security testing (SAST) tool that identifies and mitigates security vulnerabilities in source code.
* ESLint: A pluggable JavaScript linter that helps identify and fix coding errors, maintain code quality, and enforce coding standards.

5) Test Management Tools:
* HP ALM/Micro Focus ALM: A comprehensive test management solution for managing test cases, requirements, defects, and test execution.
* TestRail: A test case management tool that helps teams manage, organize, and track their testing activities.
* Zephyr: provides test management solutions for agile teams, integrating with popular agile project management tools like Jira.

ACTIVITY-12

study and prepare report on widely used software metrics.

1) Lines of code (LOC):
* Measures the size of the codebase by counting the number of lines of code.
* useful for estimating project effort and complexity.
* Can be misleading if used as the sole measure of software complexity or quality.

2) Cyclomatic Complexity:
* Measures the complexity of a program's control flow by counting the number of linearly independent paths through the code.
* Higher cyclomatic complexity indicates higher code complexity and potential difficulty in understanding and testing the code.
* Helps identify areas of code that may require refactoring or additional testing.

3) Code Coverage:-
* Measures the proportion of code that is executed by automated tests.
* Helps assess the effectiveness of the test suite in exercising different parts of the codebase.

* High code coverage does not guarantee absence of bugs, but low code coverage indicates areas of the code that are not adequately tested.

4) Defect Density:
* Measures the number of defects per unit of code size (e.g., defects per KLOC).
* provides insights into the quality of the codebase and the effectiveness of the development process.
* High defect density may indicate poor code quality or inadequate testing practices.

5) Mean Time to Failure (MTTF) and mean Time to Repair (MTTR)**:
* MTTF measures the average time between failures in a software system.
* MTTR measures the average time taken to repair a failed system.
* used in reliability engineering to assess system reliability and availability.

6) Code Churn:
* Measures the rate of code changes, including additions, deletions, and modifications over time.
* High code churn may indicate instability or frequent changes in requirements.
* can help identify areas of the codebase that require more attention during testing or refactoring.

ACTIVITY - 13

Identify different quality tools and report their features and usage

1) check sheets:-
*Features:- simple data collection tools used to track the frequency of events or defects.
usage: Helps in identifying patterns and trends in data, useful for identifying areas for improvement.

2) Pareto chart:-
Features:- Combines bar and line graphs to prioritize issues based on their frequency or impact.
usage:- Identifies the most significant factors contributing to a problem, allowing teams to focus on the vital few rather than the trivial many.

3) Cause-and-Effect Diagram (Fishbone Diagram):-
Features:- A visual tool for identifying potential causes of a problem or effect.
usage:- Helps in brainstorming into categories, facilitating root cause analysis.

4) Histogram:-
Features:- Displays the distribution of data over a continuous interval.

usage :- useful for understanding variation within a process and identifying potential outliers or trends.

5) Control charts :-
Features :- Graphical representations of process data over time, with control limits to identify variation.
usage :- Monitors process stability and detects any changes or trends, helping in maintaining consistency and quality.

6) Scatter Diagram :-
Features :- shows the relationship between two variables.
usage :- Helps in identifying potential correlations between factors, aiding in decision-making and problem-solving.

7) Flowcharts :-
Features :- Diagrams that represent processes or workflows.
usage :- Helps in visualizing and analyzing processes, identifying inefficiencies, and standardizing procedures.

8) statistical Process control (SPC) :-
Features :- uses statistical techniques to monitor and control processes.
usage :- Helps in identifying variations in processes, ensuring they remain within acceptable limits and producing consistent output.