Q1) What is Agility? Explain Kanban Model.

A) **Agility:**

Agility, in a general sense, refers to the ability to move quickly and easily, to be nimble and flexible. In the context of business and project management, agility is often associated with the capacity of an organization to adapt and respond rapidly to changes in the internal or external environment. It involves being able to adjust strategies, processes, and resources efficiently to meet evolving needs and uncertainties.

In the realm of software development and project management, agility is frequently linked with Agile methodologies, which prioritize iterative and incremental development, collaboration, and customer feedback. Agile methodologies, such as Scrum and Kanban, aim to enhance responsiveness to changing requirements and improve overall project outcomes.

Kanban Model:

Kanban is a visual management method that originated in manufacturing and was later applied to software development and project management. It emphasizes continuous delivery and flow, with the primary goal of minimizing lead time and optimizing efficiency. The word "Kanban" itself is Japanese and means "visual signal" or "card."

Key principles of the Kanban model include:

1. **Visualization:** The work process is visualized on a Kanban board, which typically consists of columns representing different stages of the workflow (e.g., "To Do," "In Progress," "Done"). Cards or tasks represent work items and move through these columns.

2. **Limiting Work in Progress (WIP):** To prevent overloading teams and ensure a smooth flow, limits are set on the number of tasks allowed in each stage of the workflow. This helps balance workloads and promotes a steady pace of work.

3. **Flow:** Kanban emphasizes the continuous and smooth flow of work through the system. Teams focus on completing tasks one at a time, and new work is pulled into the system only when there is capacity.

4. **Feedback Loops:** The Kanban model encourages frequent feedback and continuous improvement. Teams regularly review their processes and performance to identify areas for optimization.

5. **Pull System:** Work is pulled into the system based on actual capacity and demand rather than being pushed onto the team. This helps prevent bottlenecks and ensures that work is taken on when there is available capacity.

Kanban is known for its adaptability and can be applied in various contexts, making it suitable for both software development and non-software projects. It provides a visual way to manage work, optimize flow, and respond to changing priorities, aligning with the broader principles of agility in project management.

Q2) What is Feasibility study? Discuss the different types of feasibility studies.

A) **### Feasibility Studies (5 marks):**

A feasibility study is a comprehensive analysis conducted to assess the practicality and viability of a proposed project or business venture. It provides decision-makers with the necessary information to make informed choices about whether to proceed with the project. The primary objectives of a feasibility study include:

1. **Assessment of Viability (1 mark):**
- The study aims to determine whether the proposed project is feasible and realistic, considering various aspects such as technical, economic, legal, and operational factors.

2. **Risk Identification (1 mark):**
- Feasibility studies help identify potential risks and challenges associated with the project. This allows decision-makers to understand and mitigate risks, enhancing the chances of project success.

3. **Cost-Benefit Analysis (1 mark):**
- An essential aspect of feasibility studies is the evaluation of costs and benefits associated with the project. This includes estimating the financial implications and potential returns on investment.

4. **Informed Decision-Making (1 mark):**
- The ultimate goal of a feasibility study is to provide decision-makers with the necessary insights to make well-informed choices about whether to proceed with the project, modify its scope, or abandon it.

5. **Project Planning Foundation (1 mark):**
- Feasibility studies lay the foundation for project planning by providing a clear understanding of the project's requirements, potential obstacles, and the overall environment in which the project will be executed.

Types of Feasibility Studies (5 marks):

1. **Technical Feasibility (1 mark):**
- **Definition:** Assesses whether the proposed project can be implemented from a technological perspective.
- **Considerations:** Evaluates technology availability, expertise, infrastructure, and potential technical challenges.

2. **Economic Feasibility (1 mark):**
- **Definition:** Evaluates the financial aspects of the proposed project.
- **Considerations:** Involves cost estimation, benefits analysis, and determining the overall financial viability of the project.

3. **Legal Feasibility (1 mark):**
- **Definition:** Examines the project's compliance with laws and regulations.
- **Considerations:** Involves assessing zoning laws, environmental regulations, intellectual property rights, and other legal constraints.

4. **Operational Feasibility (1 mark):**
- **Definition:** Assesses the practicality and effectiveness of the proposed system or project.
- **Considerations:** Examines alignment with existing processes, integration ease, and potential impact on day-to-day operations.

5. **Scheduling Feasibility (1 mark):**
- **Definition:** Evaluates whether the project can be completed within a reasonable timeframe.
- **Considerations:** Involves creating a project schedule, identifying critical milestones, and assessing the time required for each phase.

In summary, feasibility studies are crucial for decision-making by providing a comprehensive analysis of project viability, identifying risks, and offering insights into the financial and operational aspects. Different types of feasibility studies focus on specific dimensions, ensuring a holistic assessment of the project's feasibility.

Q3) Explain McCall's Quality factors?

A) The McCall's Quality Factors, developed by John McCall and his colleagues, are a set of software quality attributes that help in evaluating the quality of a software product. These factors provide a framework for assessing different aspects of software from various perspectives. Here's an explanation of McCall's Quality Factors:

1. **Correctness (1 mark):**
- Correctness measures how well the software meets its specified requirements and performs its intended functions without errors. A correct software product produces accurate and reliable results under various conditions.

2. **Reliability (1 mark):**
- Reliability refers to the ability of the software to perform consistently and predictably over time. A reliable software system minimizes the occurrence of failures or unexpected behavior and can recover gracefully from errors.

3. **Efficiency (1 mark):**
- Efficiency assesses how well the software utilizes system resources, such as processing power and memory. An efficient software system performs its functions in a timely manner and doesn't waste resources unnecessarily.

4. **Integrity (1 mark):**
- Integrity focuses on the security and robustness of the software. A software product with high integrity ensures that data is protected, transactions are secure, and the system can resist unauthorized access or malicious attacks.

5. **Usability (1 mark):**
- Usability evaluates how user-friendly the software is. It includes factors such as ease of learning, ease of use, and overall user satisfaction. A usable software product is designed with the user's experience in mind, making it intuitive and efficient to operate.

In summary, McCall's Quality Factors provide a comprehensive framework for assessing software quality from different dimensions. By considering correctness, reliability, efficiency, integrity, and usability, developers and evaluators can gain a well-rounded understanding of the overall quality of a software product. Each factor addresses a specific aspect, contributing to the creation of a software system that not only meets its functional requirements but also delivers a positive user experience while being secure and resource-efficient.

Q4) Explain Evolutionary Process Model?

A) The evolutionary process model is a software development approach that emphasizes continuous refinement and improvement of the software product through iterative development and feedback. Unlike traditional linear models, such as the Waterfall model, the evolutionary process model allows for flexibility, adaptation, and ongoing evolution of the software based on changing requirements and user feedback. Here's an explanation of the evolutionary process model:

1. **Iterative Development (1 mark):**
- The evolutionary process model involves the development of the software in multiple iterations or increments. Each iteration represents a version of the software that is built, tested, and then refined based on feedback and changing requirements.

2. **Incremental Growth (1 mark):**
- The software evolves incrementally with each iteration. New features, improvements, or changes are added in a step-by-step fashion, allowing for a gradual and manageable development process.

3. **Continuous Feedback (1 mark):**
- User feedback and evaluation are integral to the evolutionary process model. As each iteration is completed, it is evaluated by users or stakeholders. This feedback is then used to refine the software in subsequent iterations.

4. **Adaptability to Change (1 mark):**
- The model is highly adaptable to changing requirements. If there are modifications or new features identified during the development process, they can be easily incorporated into the next iteration, ensuring that the software aligns with evolving needs.

5. **Phases of Evolution (1 mark):**
- The evolutionary process typically involves several phases, including prototyping, development, testing, and feedback. The cycle repeats until the software reaches a satisfactory level of quality and functionality. This cyclic nature allows for continuous improvement.

In summary, the evolutionary process model is characterized by its iterative and incremental approach to software development. It allows for flexibility, responsiveness to changing requirements, and continuous refinement of the software through user feedback. This model is particularly suitable for projects where requirements are not well-defined initially or are expected to change over time. The ability to adapt to evolving needs makes it a valuable approach in dynamic and uncertain development environments.

Q5) Explain different types of coupling and cohesion?

A) Coupling and cohesion are concepts in software engineering that describe the relationships between different components or modules within a system. They are important for understanding the design and maintainability of a software system.

Types of Coupling:

1. **Data Coupling (1 mark):**
- Data coupling occurs when modules share data, but each module maintains its independence. The modules are connected by passing data parameters, and changes in one module do not affect the others as long as the data interface remains unchanged.

2. **Control Coupling (1 mark):**
- Control coupling exists when modules share control information, such as passing control flags or variables. One module influences the behavior of another by specifying its state. Ideally, modules should be as independent as possible, and control coupling should be minimized.

3. **Stamp Coupling (1 mark):**
- Stamp coupling occurs when modules share a composite data structure, such as a record or array. Modules are connected by passing the entire data structure. Changes to one part of the structure may impact other modules using it.

4. **Data-Content Coupling (1 mark):**
- Data-content coupling happens when one module relies on the internal details or data representation of another module. This form of coupling is generally considered undesirable, as it makes the modules highly dependent on each other's internal implementation.

5. **Common Coupling (1 mark):**
- Common coupling exists when modules share a global data area or variable. Any changes to this global data can affect multiple modules. Common coupling should be minimized to enhance module independence and maintainability.

Types of Cohesion:

1. **Functional Cohesion (1 mark):**
- Functional cohesion occurs when elements within a module are grouped because they all contribute to a single, well-defined task. This is the strongest form of cohesion, and modules with functional cohesion are easier to understand and maintain.

2. **Sequential Cohesion (1 mark):**
- Sequential cohesion involves elements that are processed sequentially within a module. The output of one part becomes the input for the next. While this is a common form of cohesion, it is weaker than functional cohesion.

3. **Communicational Cohesion (1 mark):**
- Communicational cohesion involves elements that operate on the same data within a module. While the elements may not contribute to a single, well-defined task, they work on the same set of data. This form is weaker than sequential cohesion.

4. **Procedural Cohesion (1 mark):**
- Procedural cohesion occurs when elements within a module are grouped because they all belong to a specific procedure or process. While this can lead to understandable code, it is weaker than functional cohesion.

5. **Temporal Cohesion (1 mark):**
- Temporal cohesion involves elements that are grouped together because they are processed at the same time. This is considered weaker than other forms of cohesion and may indicate a lack of a clear structure in the module.

In summary, coupling and cohesion are crucial concepts in software design. Low coupling and high cohesion are generally desirable as they contribute to more modular, maintainable, and understandable systems. The goal is to design systems where modules are independent, changes are localized, and the overall system is flexible and adaptable.

SE IMP Questions

Repeated Ques
1) Explain Evolutionary Process Model? 5marks DONE
2) What is Feasibility study? Discuss the different types of feasibility studies. 10marks DONE
3) Explain software testing strategy and its techniques? 10marks DONE
4) Explain different types of coupling and cohesion? 10marks DONE
5) Short note on Mc-Calls Quality factors? 5marks DONE
6) Explain the characteristics and nature of software. 5marks DONE
7) Discuss about the principles of user interface design steps? 5marks DONE

Slight change in questions based on topic
8) What is a risk? Explain different types of risk in details? 10marks DONE
9) Explain Risk Mitigation, Monitoring, and Management (RMMM) plan. 10marks DONE
10) Explain Risk assessment and Risk Projection. 10marks DONE
11) What is SCM? Explain SCM Repositories. 10marks DONE
12) Explain change control process in SCM in detail? 10marks DONE
13) Explain the Principles of Agile methodology? Discuss the difference between Agile and Evolutionary Process Model? 10marks DONE
14) What is agility? Explain Kanban model. 10marks DONE
15) Explain V-Model in details? 10marks DONE
16) Explain different architectural styles? 10marks DONE

Not Repeated yet
17) Explain 3 P's in software project spectrum? 5marks DONE
18) Explain the steps involved in SQA Plan? 5marks DONE

19) Describe the advantages and limitations for large sized software projects? 5marks DONE
20) Explain characteristics of SRS? Build an SRS Document for online student feedback System? 10marks DONE
21) Explain LOC and Function Point estimation technique in detail? 10marks DONE
22) Short note on Process Metrics and project metrics. 5marks DONE
23) Explain in details Reengineering and reverse engineering. 10marks DONE

https://www.geeksforgeeks.org/risk-mitigation-monitoring-and-management-rmmm-plan/