|  |  |  |  |
| --- | --- | --- | --- |
| **SDLC Model** | **Advantages** | **Disadvantages** | **Best Use Case** |
| **Waterfall Model** | Easy to understand and use. Clear structure with defined stages. | Cannot move back to a previous stage once it’s done. Doesn’t handle changes well. | Small projects where requirements are very well understood. |
| **V-Shaped Model** | Emphasizes rigorous testing and validation. Clear and simple structure. | Like Waterfall, it’s inflexible to changes. | Projects with clear and fixed requirements, where high reliability is important. |
| **Prototype Model** | Reduces risk of failure, as a working model is seen early. Helps in getting user feedback and refining requirements. | May lead to too much focus on a limited prototype, not the full system. | Projects where user requirements are unclear or complex. |
| **Spiral Model** | High degree of risk management and flexibility. Allows for repeated iterations. | Can be complex to follow/understand. Needs careful management. | Large, complex, and high-risk projects. |
| **Iterative Incremental Model** | Progressive elaboration of the product. Allows for refinement with each increment. | Requires careful planning to make sure increments are meaningful. | Projects where it’s beneficial to get basic functionality out quickly and refine over time. |
| **Big Bang Model** | Minimal planning is required. Can potentially deliver quick results. | High risk, as most work is done simultaneously with minimal to no requirements identified upfront. | Small projects or simple internal tools with one or two developers. |
| **Agile Model** | High flexibility and adaptability. Emphasizes customer satisfaction and team collaboration. | Can be difficult to estimate time and cost. Requires customer and team engagement. | Projects where requirements can change and quick, incremental delivery is desired. |

**STLC and QA Testing**

**List down all the Models of SDLC**