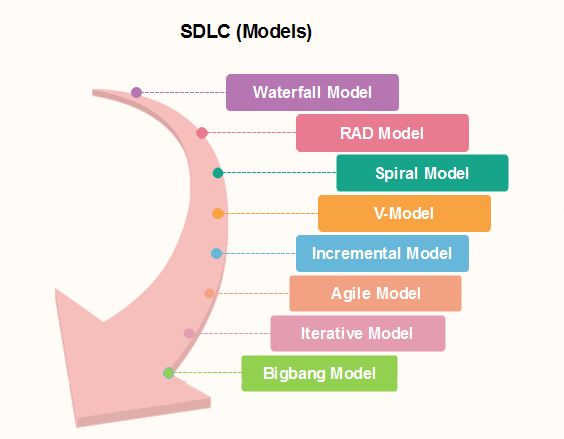
**SDLC Models**

**Software Development life cycle** (SDLC) is a spiritual model used in project management that defines the stages include in an information system development project, from an initial feasibility study to the maintenance of the completed application.

There are different software development life cycle models specify and design, which are followed during the software development phase. These models are also called "**Software Development Process Models**." Each process model follows a series of phase unique to its type to ensure success in the step of software development.

**Here, are some important phases of SDLC life cycle:**



### [**Agile Model**](https://www.javatpoint.com/software-engineering-agile-model)

Agile methodology is a practice which promotes continues interaction of development and testing during the SDLC process of any project. In the Agile method, the entire project is divided into small incremental builds. All of these builds are provided in iterations, and each iteration lasts from one to three weeks.

Any agile software phase is characterized in a manner that addresses several key assumptions about the bulk of software projects:

1. It is difficult to think in advance which software requirements will persist and which will change. It is equally difficult to predict how user priorities will change as the project proceeds.
2. For many types of software, design and development are interleaved. That is, both activities should be performed in tandem so that design models are proven as they are created. It is difficult to think about how much design is necessary before construction is used to test the configuration.
3. Analysis, design, development, and testing are not as predictable (from a planning point of view) as we might like.

### [**Iterative Model**](https://www.javatpoint.com/software-engineering-iterative-model)

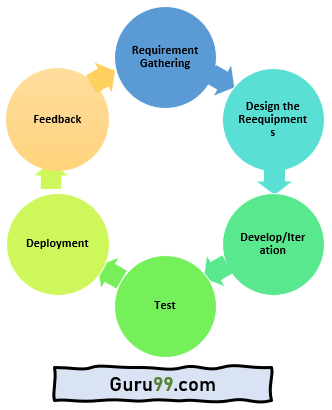
It is a particular implementation of a software development life cycle that focuses on an initial, simplified implementation, which then progressively gains more complexity and a broader feature set until the final system is complete. In short, iterative development is a way of breaking down the software development of a large application into smaller pieces.

The meaning of Agile is swift or versatile."**Agile process model**" refers to a software development approach based on iterative development. Agile methods break tasks into smaller iterations, or parts do not directly involve long term planning. The project scope and requirements are laid down at the beginning of the development process. Plans regarding the number of iterations, the duration and the scope of each iteration are clearly defined in advance.

Each iteration is considered as a short time "frame" in the Agile process model, which typically lasts from one to four weeks. The division of the entire project into smaller parts helps to minimize the project risk and to reduce the overall project delivery time requirements. Each iteration involves a team working through a full software development life cycle including planning, requirements analysis, design, coding, and testing before a working product is demonstrated to the client.







## **Phases of Agile Model:**

Following are the phases in the Agile model are as follows:

1. Requirements gathering
2. Design the requirements
3. Construction/ iteration
4. Testing/ Quality assurance
5. Deployment
6. Feedback

**1. Requirements gathering:** In this phase, you must define the requirements. You should explain business opportunities and plan the time and effort needed to build the project. Based on this information, you can evaluate technical and economic feasibility.

**2. Design the requirements:** When you have identified the project, work with stakeholders to define requirements. You can use the user flow diagram or the high-level UML diagram to show the work of new features and show how it will apply to your existing system.

**3. Construction/ iteration:** When the team defines the requirements, the work begins. Designers and developers start working on their project, which aims to deploy a working product. The product will undergo various stages of improvement, so it includes simple, minimal functionality.

**4. Testing:** In this phase, the Quality Assurance team examines the product's performance and looks for the bug.

**5. Deployment:** In this phase, the team issues a product for the user's work environment.

**6. Feedback:** After releasing the product, the last step is feedback. In this, the team receives feedback about the product and works through the feedback.

## **Agile Testing Methods:**

* Scrum
* Crystal
* Dynamic Software Development Method(DSDM)
* Feature Driven Development(FDD)
* Lean Software Development
* eXtreme Programming(XP)

### **Scrum**

SCRUM is an agile development process focused primarily on ways to manage tasks in team-based development conditions.

There are three roles in it, and their responsibilities are:

* **Scrum Master:** The scrum can set up the master team, arrange the meeting and remove obstacles for the process
* **Product owner:** The product owner makes the product backlog, prioritizes the delay and is responsible for the distribution of functionality on each repetition.
* **Scrum Team:** The team manages its work and organizes the work to complete the sprint or cycle.

## **When to use the Agile Model?**

* When frequent changes are required.
* When a highly qualified and experienced team is available.
* When a customer is ready to have a meeting with a software team all the time.
* When project size is small.

## **Advantage(Pros) of Agile Method:**

1. Frequent Delivery
2. Face-to-Face Communication with clients.
3. Efficient design and fulfils the business requirement.
4. Anytime changes are acceptable.
5. It reduces total development time.

## **Disadvantages(Cons) of Agile Model:**

1. Due to the shortage of formal documents, it creates confusion and crucial decisions taken throughout various phases can be misinterpreted at any time by different team members.
2. Due to the lack of proper documentation, once the project completes and the developers allotted to another project, maintenance of the finished project can become a difficulty.

### Summary:

* The Agile Model is an incremental and iterative process of software development.
* It focuses on working software rather than comprehensive documentation.
* Agile model is divided into various stages like 1) Requirements Gathering, 2) Design the Requirements, 3) Develop/Iteration, 4) Test, 5) Deployment 6) Feedback.
* Various types of Agile types are: 1) Scrum, 2) Crystal, 3) Dynamic Software Development Method (DSDM): 4) Feature Driven Development (FDD), 5) Lean Software Development 6) Extreme Programming (XP).
* The agile model is used when frequent changes need to be implemented.
* It provides a very realistic approach to software development
* This model has a greater risk of sustainability, maintainability, and extensibility.
* Agile methodologies in Software Testing adopt incremental and iterative approaches to software design, whereas software development flows sequentially from the starting point to the endpoint.

## **What are sprints?**

With scrum, a product is built in a series of iterations called sprints that break down big, complex projects into bite-sized pieces.

["Sprints make projects more manageable, allow teams to ship high-quality work faster and more frequently, and gives them more flexibility to adapt to change."](https://twitter.com/intent/tweet?text=%20%20%22Sprints%20make%20projects%20more%20manageable,%20allow%20teams%20to%20ship%20high-quality%20work%20faster%20and%20more%20frequently,%20and%20gives%20them%20more%20flexibility%20to%20adapt%20to%20change.%22%20&url=https://www.atlassian.com/agile/scrum/sprints&via=Atlassian)

## **How to plan and execute scrum sprints**

### **What is a user story?**

A [user story](https://www.wrike.com/agile-guide/epics-stories-tasks/) is a small unit of work in an[Agile workflow](https://www.wrike.com/agile-guide/agile-development-life-cycle/). It is a short, written explanation of a particular user’s need and how it can be fulfilled. There is no room for jargon in a user story. It is written in easily accessible language to provide a clear picture of what the user requires. The technical details can be discussed at a later stage.

A collection of user stories is referred to as an [epic](https://www.wrike.com/agile-guide/agile-epics-guide/). A [product owner](https://www.wrike.com/blog/what-is-product-management/) will be responsible for managing these user stories.

### User stories: Examples

User stories follow a simple template. The chosen user story format will outline the “who,” “what,” and “why” of a particular requirement.

* **Who** wants something?
* **What** do they want?
* **Why** do they want it?

The following template is one of the most common:

**“As [persona], I want to [action], so that I can [benefit].”**

For each story, the writer will include a user persona, the action they wish to take or the ability they wish to have, and the benefit they hope to achieve as a result. Here are some examples:

##### Example 1: An online gamer

“As an online gamer, I want to have a multiplayer option so that I can play online with friends.”

##### Example 2: A design team lead

“As a design team lead, I want to organize assets, so I can keep track of multiple creative projects.”

##### Example 3: An e-commerce shopper

“As an e-commerce shopper, I want to filter my searches so I can find products quickly.”

Now that you know what a user story looks like, you can get to work creating one.

### Five steps for writing user stories

Want some practical advice on how to write user stories? Use these five steps as a guide:

##### Step 1: Outline acceptance criteria

The [definition of done](https://www.wrike.com/project-management-guide/faq/what-is-definition-of-done-agile/) is the set of criteria that needs to be fulfilled for your user story to be considered complete. Define the specific acceptance criteria for each user story and use it as a checklist.

##### Step 2: Decide on user personas

Conduct extensive user research by creating surveys, hosting focus groups, and reading user forums. Analyze your data and search for patterns to identify your key [personas](https://www.wrike.com/blog/how-create-marketing-persona/amp/).

##### Step 3: Create tasks

Break your user story down into numerous [tasks](https://www.wrike.com/agile-guide/epics-stories-tasks/) to make it more manageable. If it is a complex requirement, you can also add subtasks. Include detailed descriptions, so your team is aligned on what each task requires.

##### Step 4: Map stories

Use [story mapping](https://www.wrike.com/blog/story-mapping-tools-guide/) to structure work in a large process. In this case, your user stories will take the form of ordered steps.

##### Step 5: Request feedback

Speak to users and potential customers to find out what they want. Ask them for their opinions on existing products or if they have suggestions for new features. Incorporate this feedback into your user story.

### What makes a good user story?

So, you’ve written your user story. But how do you know if it’s any good?

Agile teams assess the quality of stories by using the [INVENT acronym](https://www.agilealliance.org/glossary/invest/). This stands for:

* **I**ndependent: The user story should be independent of all others. Because they are not connected, they can be worked on in any order.
* **N**egotiable: A user story should be flexible enough to allow for negotiation between the customer and product owner.
* **V**aluable: What value does the user story bring? If you cannot find any value, the story should not be completed.
* **E**stimable: You should be able to estimate how long a user story will take so that you can effectively manage your time.
* **S**mall: The user story must be small enough to be completed within a single sprint.
* **T**estable: You must be able to test your user story in line with quality assurance standards.

If a user story does not meet the INVENT criteria, it should be rewritten or removed from the [epic](https://www.wrike.com/agile-guide/agile-epics-guide/). However, if it does, your team members can get to work. Schedule daily [Agile meetings](https://www.wrike.com/blog/agile-meetings-guide/) to check on their progress and ensure they are on track to complete the user story within the [sprint timeframe](https://www.wrike.com/project-management-guide/faq/what-is-a-sprint-in-agile/).

### Benefits of user stories

Why write user stories in the first place? Because they offer numerous benefits for an Agile project. Here are a few examples:

* **Simplified format:**User stories are written in easy-to-understand language. This eliminates confusion and makes it easier to grasp what the customer is looking for.
* **Increased flexibility:**Because user stories don’t go into technical detail, they can be molded to fit changing situations.
* **Improved collaboration:**When team members are aligned on one goal, they can work better together and collaborate easily with other project [stakeholders](https://www.wrike.com/project-management-guide/faq/what-is-a-stakeholder-in-project-management/).

Though the benefits of writing user stories are significant, a product owner must also consider the potential disadvantages.

## **What is a product backlog?**A product backlog is a prioritized list of work for the development team that is derived from the roadmap and its requirements. The most important items are shown at the top of the product backlog so the team knows what to deliver first.

**What is a product backlog in scrum?**

The agile product backlog in Scrum is a prioritized features list, containing short descriptions of all functionality desired in the product. When applying Scrum, it's not necessary to start a project with a lengthy, upfront effort to document all requirements.

