1. **Planning and Requirement Analysis:**
   * ***Explanation:*** In this phase, project goals, scope, timelines, and resources are defined. Requirements are gathered from stakeholders to understand what the software needs to achieve.
   * ***Example (E-commerce):*** Planning involves deciding the features of the e-commerce platform, such as user registration, product catalog, and shopping cart functionality. Requirements include user stories like "As a customer, I want to browse products."
2. **Feasibility Study:**
   * ***Explanation:*** This phase assesses the project's technical, operational, and economic feasibility. It helps determine whether the project is viable and worth pursuing.
   * ***Example (E-commerce):*** The feasibility study examines if the e-commerce project is financially viable, considering costs of development, potential revenue, and market demand.
3. **Design:**
   * ***Explanation:*** The design phase involves creating the architecture and technical specifications based on the requirements. It outlines how the system will be implemented.
   * ***Example (E-commerce):*** Designing the e-commerce application involves creating wireframes and deciding on the database structure, ensuring the system can handle product catalogs and user data efficiently.
4. **Coding:**
   * ***Explanation:*** In this phase, actual coding of the software takes place. Developers write code based on the design specifications.
   * ***Example (E-commerce):*** Developers write the code for the e-commerce platform, implementing features like user authentication, product search, and order processing.
5. **Testing:**
   * ***Explanation:*** Testing involves verifying that the software functions as intended and identifying and fixing defects.
   * ***Example (E-commerce):*** QA engineers perform testing on the e-commerce site, checking that users can register, browse products, add items to the cart, and complete the checkout process without errors.
6. **Deployment:**
   * ***Explanation:*** Deployment is the process of making the software available for users. It includes installation, configuration, and ensuring the system is ready for use.
   * ***Example (E-commerce):*** The e-commerce platform is deployed to a web server, and the website becomes accessible to customers for online shopping.
7. **Maintenance:**
   * ***Explanation:*** Maintenance involves ongoing support, bug fixes, and updates to keep the software operational and up-to-date.
   * ***Example (E-commerce):*** Maintenance for the e-commerce application includes fixing any issues that arise, updating product information, and adding new features based on user feedback.

**Agile and Waterfall Comparison Chart**

|  | **Waterfall** | **Agile** |
| --- | --- | --- |
| Timeline | Waterfall has a fixed timeline. The idea is that the start and finish of the project are already mapped out from the beginning. | Agile is a lot more flexible and accounts for experimenting with different directions. Rather than a fixed timeline, the schedule adapts as the project progresses. The [Agile Manifesto](http://agilemanifesto.org/principles.html), an online document released in 2001 by a group of software developers, says team members are expected to, “​​Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.” |
| Client Involvement | Once the end goal is established, Waterfall does not involve the client or project owner during the process, apart from specific check-ins or for deliverables. The course of the project is outlined from the start, so incorporating client feedback is not an ongoing part of the process. | A fundamental part of Agile is including clients in the project development at every step. The Agile Manifesto states, “Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.” Therefore, business owners are expected to be involved and give feedback to the software development team as they progress through the different phases of the project. |
| Flexibility | Waterfall is not as flexible as Agile because each phase needs to be fully completed before moving on to the next phase. The project is also planned out ahead of time, making this management system ideal for teams with a clear vision of where they are headed from start to finish. | Flexibility is built into the Agile method. Agile values short bursts of work, which are called sprints. The method welcomes adapting to different directions, incorporating new information even at a later stage of the project. |
| Budget | Fixed.  The budget for projects using the Waterfall methodology is generally fixed. Because the project is determined from start to finish, there is less room to change the budget mid-project. | Flexible.  Agile is open to adaptation, encourages experimentation and welcomes changes of direction, even in later phases of the project. Because of this, the budget tends to be more flexible. |

**The following 12 Principles are based on the**[**Agile Manifesto**](https://www.agilealliance.org/agile101/the-agile-manifesto/)**.**

1. Our highest priority is to satisfy the customer through the early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer’s competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. Business people and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity–the art of maximizing the amount of work not done–is essential.
11. The best architectures, requirements, and designs emerge from self-organizing teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.