**10 key principles of user centered design**

**What is User-centered Design?**

Systems are designed to meet business goals through the fancy features, and technological capabilities of software or hardware tools. However, these system design approaches omit the end user, which is an essential part of the process.

User-centered Design (UCD) is the process of developing a tool, for instance, the user interface of a website or application, from the perspective of how it will be understood and used by a human user.

A system can be tailored to support its aimed users' existing beliefs, attitudes, and behaviors as they connect to the tasks that the system is intended to perform.

Employing UCD to a system design results in a product that delivers more efficiently, satisfyingly, and user-friendly experience for the user; thus, leading to increased sales and customer loyalty.

**Usability and its Importance**

Usability is a measure to which a product can help a user achieve the set goals effectively, efficiently, and satisfactorily in a specified environment.

This can be attained through a user-friendly interface design. It is easy for you to learn, and supports your tasks and goals in a more efficient and effective manner. Devoid of usability can cost you time, effort and can significantly influence a system's success or failure.

**Key principles of user-centered design**

User centered design is usually based on principles for it to remain focused on usability throughout the entire development process and further throughout the system life cycle. Here is a summary of the 10 key principles for the design of usable systems they can be customized to meet your unique interaction needs.

**1. Design for the users and their tasks**

Isolation does not work for interactive computer systems since they have to support those using them to perform what is required. Centered for support to the users which is key in making them user-centered and task-oriented. During development and this includes the whole period, it is important for the developer to consider the characteristics of the user population, the tasks involved in the real world and the specified environment.

**2. Maintain consistency**

The users need a system that is easy to learn with minimal and understandable requirements. The behavior of interface elements should be consistent. In fact, consistency will start at the designing phase so as to integrate with the existing components in a computer system. You can have a new design approach to counter interaction but what is most important is for to look at how well it contributes to consistency issue. This will determine how users will view your approach and the time taken to learn.

**3. Use simple and natural dialogue**

The core application of a system should be incorporated with proper interaction to enable a dialogue with the user. The user should see only the relevant information that is essential for task completion because each time irrelevant information is added, it puts the user in a more complicated situation. It is advisable for the developer to use plain English and use vocabulary that is relevant to the targeted audience. Define the terminology so that it carries the same meaning.

**4. Reduce unnecessary mental effort by the user**

Users like to concentrate on the task at hand and worry less or not at all of the tool in use and its interaction with the designed application. They are more frustrated with complicated interaction with the computer or any other mobile device compatible with the application. Why? They are distracted from the main work.

Too much effort invested in learning on the operation part makes them less efficient and prone to errors. This can certainly cost a business that heavily relies on the outcome of a task. The frequency of tasks is necessary since users don't have to memorize information from a previous part of the system that is to be used in the next part. Instructions on how to use should be clearly defined and can be retrieved when needed.

**5. Provide adequate feedback**

Users need assurance that their actions have been successfully executed. This can be made evident by a change in the appearance when completion is achieved successfully. If it takes longer, an indicator is useful to show that processing is still in progress, and this keeps the confidence of a user in shape. What is kept away is the information providing status about the internal affairs of the system.

Various levels of interaction should be backed up by feedback. At a lower level, confirmation can be received when a control operation is successful. A good example is a button appearing somehow pressed in to indicate that the user has already pressed it. Long operations can be verified by the system upon completion.

**6. Provide adequate navigation mechanisms**

Users being able to navigate with ease are another vital principle for them to know their position. This is made possible by application of an efficient and consistent mechanism that assigns titles to the current windows and use of indicators like page numbers and bars for scrolling. Other things that can be included are an overview, history of visited areas and a navigation map.

You need to provide clear routes between the different windows that the user is engaged. The form of provision should be appropriate for the user while at each stage of the intended task.

Sometimes, users can find themselves in areas that they do not intend to be in. There should be a clear emergency exit that can be used to leave an unwanted state without having to go through it like a Cancel button.

**7. Let the user take charge**

The user knows what he or she needs, and the developed system provides the solution. For the user to do what is required, they should be able only to take what is required and leave the rest to support an individual request. Constraints evoked by the system should as minimal as possible which prompts the developers to provide easy ways to achieved what is frequently needed.

**8. Present information clearly**

The arrangement of information is essential to the user while on-screen which enables the user to single out the different elements and data groups. This can be achieved by using boxes, spaces, and visual coding proficiency. Again, developers should not provide more than the necessary information to process a task.

**9. Offer Assistance**

A user should get all the help needed from a system with minimal use of the document provided. In other words, they should be self-explanatory. Information provided on the window should be in line with the user's tasks. It is important for you to provide tool tips for icon-labeled buttons.

Online help should be related to whatever interaction is provided on the window. Task-oriented should be the tone with a list of steps to be followed.

**10. Error-free**

Minimize errors by directing the users towards the right way to achieve their goals. Feedback from users should be constrained to prevent error, where necessary to the task. However, this shouldn't apply in situations that will end up limiting your users' the choice in how to accomplish their tasks. Ensure that the system validates data entry as close as possible to the point of input.

Consider expressing error messages in plain language to avoid the use of codes, point out the exact problem, and offer a solution suggestion to the problem.

**How to Maximize Usability**

To maximize usability, you can employ iterative design, which gradually enhances the design through assessment from the early design stages. These steps enable you to incorporate feedback from the user and client until the system attains an ideal usability level.

The preferred approach for you to ensure usability is by testing actual users on an operating system. For you to achieve significant levels of usability, necessitates you to focus on the design efforts for the specified system end-user.

Some of the methods to determine the primary users, how they work, and what duties they should accomplish, include user testing on system prototypes, a usability audit performed by specialists, and cognitive modeling.