Human-Computer interaction (HCI) is the study of how humans interact with computer systems. HCI is a broad term that covers all aspects of how people interact with computers.

Importance of HCI

1. Usability: Human-computer interaction is crucial for making computer systems easy to use. If users can't easily interact with a system, they won't be able to use it effectively.
2. Efficiency: Human-computer interaction helps to make computer systems more efficient. By designing systems that are easy to use, users can get more done in less time.
3. Productivity: The ease of use and efficiency gained from human-computer interaction leads to increased productivity.

User Interface (UI) can be defined as that part of the computer system with which users interact to undertake their tasks and achieve their goals.

What Is a Good User Interface Design?

A good user interface design is one that encourages an easy, natural, and engaging interaction between a user and a system, allowing users to carry out their required tasks.

Importance of a good UI design

- A good user interface encourages user retention. When users have a positive experience with a product, they are more likely to return and continue using it
- A good UI will make a product more usable and intuitive, resulting in a positive ui experience
- A good UI gives a product a significant competitive advantage

What Is Usability?

Usability is defined as "the extent to which a product can be used by specified users to achieve specified goals with **effectiveness**, **efficiency** and **satisfaction** in a specified context of use." **Effectiveness** is the accuracy and completeness with which specified users can achieve specified goals in particular environments. **Efficiency** is defined as the resources expended in relation to the accuracy and completeness of the goals achieved. **Satisfaction** is the comfort and acceptability of the work system to its users and other people affected by its use.

The Problems of Poor or Bad User Interfaces

- User Frustration and Dissatisfaction
- Loss of Productivity, Efficiency, and Money
- Small Irritations Are Also a Problem

User-centered design (UCD) is an approach to user interface design and development that involves users throughout the design and development process.

Principles of Human-centered design

1. The active involvement of users
2. An appropriate allocation of function between user and system

3. The iteration of design solutions
4. Multidisciplinary design teams

Essential activities for human-centered design

1. Understand and specify the context of use
2. Specify the user and organizational requirements
3. Produce design solutions (prototypes)
4. Evaluate designs with users against requirements

Ways to be user-centered

1. **Involving Users**

The way to be user-centered is to involve users and pay attention to their views. This can include a variety of approaches, from simply observing users' working practices as part of collecting system requirements to using psychologically based user modelling techniques, to including user representatives on the design team.

Who Are the Users?

- Customers, who pay for and perhaps specify the computer system under development
- Other people within the users' organizations who have an interest in the development of the system
- Users or end users — the people who use the system directly to undertake tasks and achieve goals

2. **Making the Design Process Iterative**

Making the user interface design and development process iterative is a way of ensuring that users can get involved in design and that different kinds of knowledge and expertise related to the computer system can be brought into play as needed.

When and How to Involve Users

- Early in the design process when the requirements are being specified. Users could help in defining the requirements for the system by contributing a specification or by testing early mockups. Users can get involved by allowing themselves to be observed and giving feedback about the problems of the current system.
- During prototyping, to test designs and options. Users could test versions of the interface to provide feedback and make suggestions to the designers.
- Just before delivery of the product. Users again could test the product by using it or completing surveys about the various features. At this point, however, only minimal changes would be allowable.
- During training/after delivery of the system. Again, users would use the product and give their opinions and detail any problems. Revisions at this stage would be included in the next version.

When and How Do You Evaluate?

Evaluation is a way of finding out whether your systems work, and it is an ongoing activity throughout the life cycle. You can look for as many problems as possible (diagnostic evaluation) or you can try to measure the performance of the system (measurement evaluation).

Stages of Evaluation

- Evaluation Early in the Life Cycle: It is undertaken to validate the users' requirements, predict the usability of the product or the usability of an aspect of the product, and assess how well the interface meets the users' needs. The earliest user evaluations may be best done using paper-based prototypes and mockups. These are very low cost but can yield a great amount of information and feedback about a design. Findings from early evaluations can be taken on board and fed back into the design process before the design is set.

- Evaluation Later in the Life Cycle: It is also undertaken to assess how well the user interface meets the users' needs. It is carried out to check the usability of the nearly completed system and to ensure that the system meets the specified usability requirements. At this point, findings are unlikely to be fed into the UI design and development process, as generally by this point the system is more or less ready. These findings might be used for the next version or release of a system rather than for changing the near-finished product.

How Do You Evaluate?

- Observing the organization and how people work. Several kinds of evaluation depend on some form of observation or monitoring of how users interact with a product or prototype. The observation may take place informally in the field or a laboratory as part of more formal usability testing.
- Interviewing, talking, and asking questions. As well as examining users' performance, it is important to find out what they think about using the technology. No matter how good users' performance scores are when using technology, if for some reason they do not like using it, then it will not be used. Surveys using questionnaires and interviews provide ways of collecting users' attitudes to the system.
- Making predictions. This kind of evaluation aims to predict the types of problems that users will encounter without actually testing the system with them.

How to gather requirements

- Interviews and focus groups: Interviewing involves talking to or questioning users. It enables the gathering of information in a fast and friendly way. You will need to plan interviews carefully, deciding in advance who to interview, what questions to ask to gather the relevant information, and how long the interviews should be. There are two main kinds of interview: **structured and unstructured (flexible)**. A **structured interview** has predetermined questions that are asked in a set way; there is little, if any, scope for exploring additional topics that might arise during the interview. In contrast, a **flexible interview** generally has some set topics for discussion and exploration, but no set sequence: the interviewer is free to follow up the interviewees' replies, and to find out more about anything that is said.

- **Question-based surveys:** refer to the preparation of unambiguous questions and statements for the gathering of more precise information. Broadly speaking, there are two question structures for questionnaires: **closed questions and open questions**. **A closed question** asks the respondent to select an answer from a choice of alternative replies. Closed questions may require just “yes” or “no” responses, or they may have some form of rating scale associated with them. Which type you use will depend on whether you need simple or detailed information. **An open question** allows respondents to say whatever they like in response, and they are used where there are no predetermined answers. Open questions typically start with phrases such as “What do you . . . ,” “How do you . . . ,” or “What ways . . . .” Limiting the amount of space on the form for the answer can encourage respondents to prioritize their points
- Contextual analysis

Undertake the following activities in gathering requirements:
- Observe users — real users — doing real work, where the application is to be used.
- Observe and talk to real users. Many people will have information to offer or will have something to say about the system. But you must also remember to observe and talk to real users — the people who will use the system
- Observe, talk to, and involve real users throughout the design process and its activities.

Users: Finding Out Who They Are
primary users: People or groups of people in an organization who use the application directly Secondary users are people or groups of people who are not primary users but who are affected or influenced in some way by the computer system or who affect or influence its development.

User Profiling: Describing Your Users and Their Characteristics
There are two main ways to find out about your users so that you can create a user profile
- Questionnaire
- Interview

NOTE: Check out the user profile of ATM customers on page 44, chapter 2 of the textbook

A user persona is a precise description of a user and what he or she wishes to do when using a system. Personas are not real; rather, they are imaginary examples of the real users they represent.

Goals, Tasks, and Actions

A goal is an end result to be achieved. It must be described at a high level of abstraction, indicating what is to be achieved. The specific details of how the goal is to be achieved are not stated.
A task is a structured set of related activities that are undertaken in some sequence. Tasks are what a person has to do (or thinks she or he has to do) in order to accomplish a goal.
An action is defined as an individual operation or step that needs to be undertaken as part of the task.

Mental/User's models: are the models people have of themselves, others, the environment and the things with which they interact. People form mental models through experience, training and instruction. Mental models enable users to reason about a system, to apply already held experience and knowledge about the world to system use.

Characteristics of Mental Models

- Incomplete
- Partial
- Subject to change
- Possibly inconsistent
- Based on imperfect observations and inference

Types of Mental Models

- Structural Models: assume that the user has internalised, in memory, the structure of how a particular device or system works.

- Functional Models: assume that the user has internalized procedural knowledge about how to use the device or system.

Four Psychological Principles of UI Design

1. **Users see what they expect to see**. If the OK button is usually on the left of the Delete button and a new screen swaps these around, users may think they see the OK button on the left and erroneously press the Delete button.

2. **Users have difficulty focusing on more than one activity at a time**. A user who is working in a busy environment may be distracted, so the UI should remind the user what he or she needs to do next.

3. **It is easier to perceive a structured layout**. It is difficult to perceive a button if the screen does not give you any clues, such as lines or shadows encompassing a clickable area.

4. **It is easier to recognize something than to recall it**. Some UIs expect the user to recall esoteric information such as command names. This can be difficult, particularly for a novice user. It is often better to make the information visible in the form of a menu or a button

Visibility: In considering visibility, it is often useful to start from the goal you wish to achieve.

Affordance: To have the property of affordance, the design of the control should suggest (that is, afford) how it is to be operated. It is the strong clues to the operation of things.

Feedback: is information that is sent back to the user about what action has been accomplished upon the use of a control.

The principles of visibility, affordance, and feedback are very important in user interface design because they form part of the knowledge needed and the knowledge that should be used to guide your UI designs.

Usability Requirements

Usability requirements are gathered using techniques such as interviewing, surveying, and observation.

Types of Usability Requirements

1. qualitative usability requirements
2. quantitative usability requirements.

Problems with Requirements Gathering

- Not enough user/stakeholder involvement in the requirements-elicitation process may cause the requirements to be incomplete.
- Lack of requirements management
- Communication problems related to different stakeholders with different levels of understanding can be problematic for sharing understanding of the requirements between all groups.
- Capturing the relevant application domain-related information can be difficult

Prototyping: Prototyping is an experimental process where design teams implement ideas into tangible forms from paper to digital. In prototyping, you craft a simple experimental model of your proposed product so you can check how well it matches what users want through the feedback they give.

Purposes of Prototyping

- To check the feasibility of ideas with users
- To check the usefulness of the application
- To allow users to contribute to the design of the application
- To allow users to test ideas
- To validate the requirements (i.e., to reveal inconsistent or incomplete requirements)
- To negotiate requirements

Low-Fidelity Prototypes

Low-fidelity prototypes are generally paper based and include sketches, screen mockups, and storyboards. They can be created by hand, but they can also be created using a drawing package like Paint or PowerPoint and then printed out for testing with users.

Low-fidelity prototypes are useful in the requirements-gathering phase of UI design. They can be used as a communication medium between you and the users and stakeholders. Low-fidelity prototypes also help users and stakeholders to articulate what they want and

need from a system, as they will find it easier to talk about something visual and concrete as opposed to conceptual (or abstract mental) ideas, which can be harder to share.

Advantages of Low-fidelity prototype

- They are cheap to produce
- They can evaluate design ideas and design alternatives
- They romote rapid, iterative development
- They are useful for facilitating communication between users and stakeholders and the UI designer
- They can show the look and feel and layout of the screens

Disadvantages of Low-fidelity prototype

- Their ability to check errors in design is limited
- THe specification is less detailed so it may be more difficult for programmers to code
- A human facilitator is needed to simulate how the UI will work
- They are useful for gathering requirements but are generally thrown away once the requirements have been established

High-Fidelity Prototypes

High-fidelity prototypes, which are based on software, provide a functional version of the system that users can interact with. As such, they show the UI layout and its navigation.
In essence, a high-fidelity prototype looks and behaves as if it is the final product and can be used as a tool for marketing the final product.
Ensuring that a proposed system has the necessary functionality for the tasks that users want to do is an important part of requirements gathering and task analysis. If feasible, high-fidelity prototyping can fulfill an important role in testing designs with users and in validating requirements.

Advantages of high fidelity

- They can show comlete functionality
- They can show the look and feel, layout, and behavior of the final product
- They are fully interactive, and can be useful as a marketing too;l

Disadvantages of high fidelity

- They are more time consuming to create than low-fidelity prototypes
- They are not as effective as low-fidelity prototypes for requirements gathering, because they cannot easily be changed during testing
- They can look so professional and finished that users are less willing to comment

Differences between hi-fi and lo-fi prototype

Design principles

- **Simplicity** is a design principle which emphasizes the importance of keeping the UI as simple as possible. The user interface should be communicated clearly and simply in the users' own language.

- **Structure** is a design principle that emphasizes the importance of organizing the UI in a meaningful and useful way. Features that users think of as related should appear together on the user interface, or at least they should be clearly and closely associated.
- **Consistency** is a design principle that emphasizes the importance of uniformity in appearance, placement, and behavior within the user interface to make a system easy to learn and remember. If something is done or presented in a certain way on the user interface, users expect it to be the same throughout.
- **Tolerance** is a design principle that emphasizes the importance of designing the user interface to prevent users from making errors. However, errors are not always due to poor user interface design. They could be due to poor task or domain knowledge, stress, or poor communication between colleagues.

Design rationale: A documented explanation (be it paper or electronic) of why an artifact has been designed the way that it has (that is, the reasoning behind a design).

Benefits of Design Rationale

- Design rationale facilitates the reuse of parts of the UI, thus possibly saving development effort.
- It will encourage you to approach the design process in a more systematic and principled way
- Various people — trainers, marketing personnel, or even, in case you leave the project, other UI designers — may need to understand your decisions. New staff joining the development team may also need to understand why certain design decisions have been made. The users, too, may benefit from understanding why you designed the system the way you did.