System Requirements in Software Engineering

What are System Requirements?

- **System requirements** are the specific needs or conditions that must be met for software to work properly.
- They describe what the software needs in terms of hardware, software, and functionality.
- These requirements are important because they guide the development process and ensure that the final system meets user expectations.

Types of System Requirements

1. **Functional Requirements:**
   - **What they describe:** What the system should do or the specific functions it must perform.
   - **Example:** A banking app must allow users to check their account balance, transfer money, and pay bills.
   - **Importance:** Functional requirements ensure that the system does what users expect.
2. **Non-Functional Requirements:**
   - **What they describe:** How the system performs its tasks, focusing on quality and performance.
   - **Example:** The system should load in under 2 seconds, or it should be secure to protect user data.
   - **Importance:** Non-functional requirements focus on user experience, such as speed, security, and reliability.
3. **Technical Requirements:**
   - **What they describe:** The hardware and software needed to run the system.
   - **Example:** The software might require a certain operating system (like Windows or Linux), a specific processor speed, or a minimum amount of memory (RAM).
   - **Importance:** These requirements ensure that the software is compatible with the user's system.

4. **User Requirements:**

- **What they describe:** The needs and expectations of the end-users who will be using the system.
- **Example:** The system should be easy to use and accessible to all users, including those with disabilities.
- **Importance:** User requirements ensure the system is user-friendly and meets the needs of those who will use it.

Why Are System Requirements Important?

1. **Clear Direction:**

- They give developers a clear understanding of what needs to be built, avoiding confusion or misunderstanding during development.

2. **Project Planning:**

- Helps plan resources, budget, and time needed for the project by clearly outlining what is required.

3. **Avoiding Miscommunication:**

- Ensures that everyone involved (developers, managers, and clients) is on the same page about what the system will do.

4. **Testing and Quality Assurance:**

- System requirements provide a checklist that testers can use to verify that the system meets all expectations.

5. **User Satisfaction:**

- Ensures the final product meets user needs, works properly, and provides a good experience for the users.

Conclusion

- **System requirements** define what a software system needs to do, how it should perform, and what technology is required.
- By clearly defining these requirements, developers can create a software system that meets expectations, works well, and satisfies user needs.