Non-Functional Requirement Documentation

These comprise non-functional requirements along with brief descriptions outlining the expected criteria for the software to meet each requirement.

1. Security
   Security as a non-functional requirement refers to the set of measures and features designed to safeguard a system's data, functionality, and infrastructure from unauthorized access, breaches, and malicious activities. It encompasses aspects such as authentication, authorization, encryption, and auditing to ensure the confidentiality, integrity, and availability of the software. Meeting security requirements is essential for protecting sensitive information and maintaining the trust of users and stakeholders in the software system.

2. Privacy
   Privacy involves ensuring the protection and responsible handling of users' personal information. It includes features like data anonymization, consent management, and secure data storage to comply with privacy regulations and uphold user confidentiality. Meeting privacy requirements is crucial for establishing trust, legal compliance, and ethical use of data within a software system.

3. Reliability
   Reliability as a non-functional requirement in software development refers to the capability of a system to consistently perform its intended functions without failures or disruptions. It involves minimizing the occurrence of errors, ensuring fault tolerance, and providing effective error recovery mechanisms. Reliability is crucial for maintaining system stability, user confidence, and meeting service level agreements, contributing to a dependable and trustworthy software product.

4. Maintainability
   Maintainability relates to how easily a software system can be updated, enhanced, or fixed. It involves designing code and architecture in a way that facilitates efficient and cost-effective maintenance by developers. A maintainable system should have clear documentation, modular structures, and well-defined interfaces, allowing for straightforward troubleshooting and modifications. Prioritizing maintainability ensures long-term sustainability and reduces the overall cost of software ownership.

5. Portability
   Portability in software development refers to the ability of a program to run on different systems and platforms without requiring major modifications. A portable software application should be easily transferred or adapted to various environments, such as different operating systems or hardware configurations. This characteristic allows for greater flexibility, as the software can be deployed across diverse settings while maintaining

consistent performance and functionality. Portability is a key non-functional requirement that contributes to the adaptability and accessibility of a software product
.

6. Scalability

   Scalability is a non-functional requirement that pertains to a system's ability to handle increasing amounts of workload or user demand. A scalable software application should be able to efficiently accommodate growth without a proportional increase in resources or a decline in performance. This is achieved through well-designed architectures, distributed systems, and the ability to scale both vertically (adding more resources to a single machine) and horizontally (adding more machines to a network). Scalability is crucial for adapting to changing user loads and ensuring optimal performance as a software system expands.

7. Ease of Use

   Ease of use focuses on ensuring that the software is user-friendly and intuitive. It involves designing interfaces and interactions that are easily understandable and navigable, reducing the learning curve for users. Prioritizing ease of use enhances user satisfaction, minimizes errors, and promotes efficient interaction with the software, contributing to a positive user experience.