Non functional requirements

Team-01

October 2017

1 Performance Requirements

User Satisfaction: This software application has a basic benchmark which should be satisfied. Benchmark can be defined as "it should stand up to the user's expectations. Application should always be active.
Response Time: Time taken to respond to the user operations is minute. Response would be accurate.
Error Handling: User errors and undesired situations are taken care of in testing phase. Exceptions are already handles in coding phase. All the error and exceptions have been taken care of to ensure that the application operates without any uncertainty.
User Friendliness: The application is easy to understand, operating this application is quite simple. UI makes sure that this application is user friendly and attractive. A naive user can also use this application without any difficulties.

2 Safety Requirements

As such, there are no safety requirements which are to be satisfied apart from those which are to be followed while using a system. The only hazard is user using the device when they should not be.

3 Security Requirements

Software application should provide a secure login, registration for every user. It should also make sure that these details can be changed. Session management is to be established and should be ended while logging out (after the task is done). Making sure that the user doesn't give any false feedbacks/complaints which is a important factor to other user's integrity. The server side should also be immune to malicious attacks. The user should take care of their login details and make sure that there are no other fake login's.

4 Software quality attributes

4.1 Usability

Prioritize the important functions and features of the system based on the usage patterns. Frequently used functions should be tested for usability. Most important and used functions like login page, registration template should be tested for usability. The same goes with complex and critical functions. Be sure to create a requirement for this.

4.2 Reliability

Users have to trust the system, believe in the application and the developer even after using it for a long time. Create a requirement that makes sure that data created in the system will be retained in a number of years without the data being changed by the system/application. Things -will be good if it also include requirements that make it easier to monitor the system performance.

4.3 Supportability

The system needs to be cost effective to maintain. things will be easy to manage if the system is cost effective. Maintainability requirements may cover various levels of documentation such as system documentation and documentation related to testing and test cases.