# **Chapter 3: Requirement Analysis**

In this chapter we will define all the requirements of proposed system that include functional and non-functional requirements. We will also discuss about use cases of the system and see how our system will respond to various use cases.

# **3.1.** **Functional Requirements**

For our system to work and facilitate the user number of functional requirements have been are needed. These functional requirements also be presumed as interface requirements as they are all but interface.

* The system will authenticate the user through login functionality.
* The system will display the previous projects.
* The system will display the member list to project admin.
* The system will allow project admin to create new projects .
* The system will allow project admin to perform edit delete operations on created projects.
* The system will allow to perform edit delete operation on member.
* System allow admin to assign project to different members.
* The system will allow member to test the feasibility study.
* The system will allow member to add cost.
* The system will allow member to add risk.
* The system will allow member to add Integration.
* The system will allow member to add stakeholder.
* The system will allow member to add module.
* The system will allow member to add procurement.
* The system will allow member to add Resources.
* The system will allow member to view proposal.
* The system will allow member to chat with its teammate and project admin.
* Member recieve notification if teammate or project admin send message.

# **3.1.** **Non-Functional Requirements**

* The system should be able to handle the concurrent requests from different users.
* The system should provide confidentiality for user data.
* The system should be stable and reliable enough to handle the exceptions.
* The system should be available for 24/7 of the time to handle the concurrent request of the users.
* The system should permit only authorize users to ensure its security.
* The system should be efficient enough to handle the concurrent request to the user.
* Interface and the system itself should be user friendly so that the customer will feel it easy to use.
* The system will authenticate the user by verifying the credentials to database.

# **Use Case Model**

In the Unified Modeling Language (UML), a use case diagram can summarize the details of your system's users (also known as actors) and their interactions with the system. Following are the use cases of the Product Lifecycle Management