# **2 Analysis**

## **2**.**1 Introduction to analysis**

Analysis is the first stage of software development process. During analysis, requirement analysis like functional, non-functional and Moscow prioritization are explored. For further ease of analysis diagrams like use case and Class diagram are designed.

It is the process of production of software. This is the part when programmers understand what the customer wants. It allows teams to transform the needs specified in earlier proposed proposal. Requirements and diagrams help team and user interact each properly where both parties understand how the system will be. Requirements are gathered through the help of processes like interviews, questionnaires, observations, focus-groups etc. it defines what user clearly wants.

## **2.2 Feasibility study**

It is an analysis of how successfully a project can be completed. It is an analysis of proposed project to determine if it is feasible economically, technically, operationally, schedulable and legally. An important purpose of feasibility study is that it helps planners focus on project. It also outlines alternatives. It addresses the probable issues that could affect the project.

**Types of feasibility study:**

1. Economic feasibility

It determines the efficiency of a new project also known as cost/benefit analysis. It helps in finding profit from and against investment expected from a project. Its two most essential factors are cost and time.

1. Technical feasibility

It defines the needs of current hardware and software for the proposed project. If any modification is necessary what kind of technology would be required.

1. Operational feasibility

It refers to usability of system with the user. Will there be any training required for user to operate some tasks? It also tells if any materials are required for the training.

1. Schedule feasibility

It relates the allocation of time to complete the development of the system. it also tells about the impact of the product is not delivered in time.

1. Legal feasibility

It states about the legal requirement of the project. It is important that the project is following the requirements needed to start a project including licenses, copyrights and other safety measures.

## **2.3Requirement analysis**

1. **Functional Requirements:**

The most important phase in SDLC is requirement analysis phase because this the part when programmers understands what the customer wants. Functional requirements are those tasks which describes what a system should do and are designed for users and their convenience.

E.g. a system should have a login page where people can login in order to enter their desired sites.

Functional requirements are given below:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Fid** | **Functions** | **Data** | **Rational** | **Dependency** | **Remarks** |
| F01 | Registration | Personal details | User information | F01 | Creates user account |
| F02 | Login | Username password | security | F01 | Opens user’s dashboard |
| F03 | Reset password | email | Update password | F02 | Gets new password |
| F04 | Search movies | \_ | Searching movies | F02 | User can search movies |
| F05 | Rent movie | \_ | Renting movies | F02 | User can rent a movie |
| F06 | Admins customization | Movie details | Add, Update, Delete, Cancel | F06 | User can add, update, delete movies |
| F07 | Admin login | Admin details | Admin | F09 | Admin can update profile |

1. **Non-Functional**

Non-functional requirements are those tasks that describes how the system will do so on. Non-functional aspects involve performance, usability, responsiveness etc. they are designed which are not addressed by functional testing. Usually they are in the form of how system shall do or part of the system.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nid** | **Functions** | **Data** | **Rational** | **Dependency** | **Remarks** |
| N01 | Responsive | - | Support different resolutions | N04 | Fits in different screen sizes |
| N02 | Usability | - | Easy and user friendly | N04 | User friendly |
| N03 | Maintainability | - | Easy to change | N04 | Easy maintenance and optimization |
| N04 | Reliable | - | Accurate output | N04 | Gives accurate output as per the input |
| N05 | Multi-Browser support | - | Supports different browsers | N04 | Runs in different browsers |
| N06 | Performance | - | Maintains system’s productivity | N04 | Regular tests are focused |

1. **Moscow Prioritization**

Moscow prioritization also known as Moscow method is a technique that manages both the functional and non-functional requirements. It has four prioritization categories must have, should have, could have and won’t have. It sets priorities between functionalities about how much importance they meet.

|  |  |  |
| --- | --- | --- |
| **ID** | **Functionalities** | **Priority** |
| F01 | Registration | Must have |
| F02 | Login | Must have |
| F03 | Reset password | Must have |
| F04 | Search movies | Must have |
| F05 | Rent movies | Must have |
| F06 | Admins customization | Must have |
| F07 | Admin login | Must have |
| N01 | Responsive | Must have |
| N02 | Usability | Must have |
| N03 | Maintainability | Must have |
| N04 | Reliable | Should have |
| N05 | Multi-browser support | Must have |
| N06 | Performance | Should have |

**Use case**

use case is a methodology used in system analysis to identify, clarify and organize system requirements. There is an actor and use cases in the diagram. Actor represents the roles played by users of the system.

Admin can register to the system

Admin can login to the system

Admin can add movies

Admin can add customers

Admin can update customers info

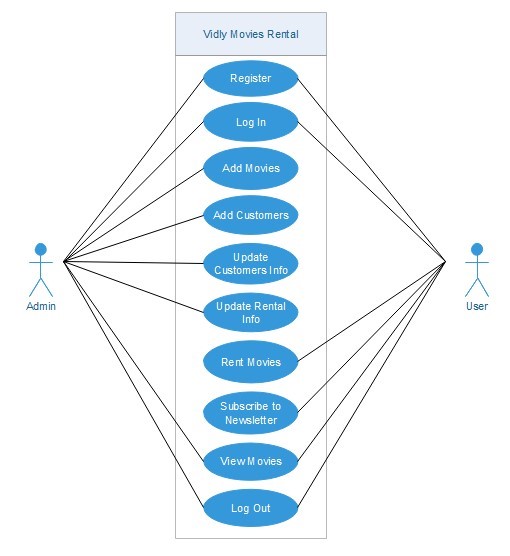
Admin can update rental info

Admin can view movies

Admin can logout of the system

User can register, log in and logout

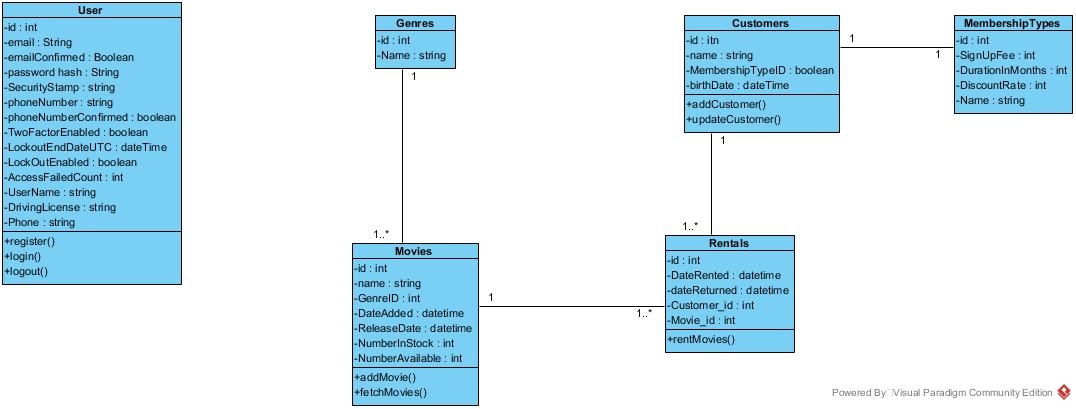
User can view and rent movies

****

**Use case diagram**

**Initial class diagram (NLA)**

They are used to show the different objects in a system, their attributes, operations and relationships among them. Customers have one to one relationship with membership types, customers have one to many relationships with rentals. Movies have one to many relationships with rentals. Genres have one to many relationships with movies.



Initial class diagram