Analysis Specification On

HOSTEL MANAGEMENT SYSTEM

SARANCE SHRESTHA

00162792

Computing Project

Level 5 Diploma in Computing

Softwarica College of IT & E-Commerce

Kathmandu, Nepal

23rd , April 2017

Table of Contents

[Introduction: 3](#_Toc480727980)

[Analysis Specification: 3](#_Toc480727981)

[Why we need to perform analysis: 3](#_Toc480727982)

[Rich Picture: 4](#_Toc480727983)

[2.1 Requirements: 5](#_Toc480727984)

[2.1.1 Functional requirements: 5](#_Toc480727985)

[2.1.2 Non-Functional requirements: 7](#_Toc480727986)

[2.1.3 Prioritization: 8](#_Toc480727987)

[2.1.4 MoSCoW prioritization: 8](#_Toc480727988)

[2.2 Use Cases: 9](#_Toc480727989)

[2.2.1 BENEFITS OF USE CASES: 9](#_Toc480727990)

[2.2.2 Scenario Description: 11](#_Toc480727991)

[2.3 Architecture: 13](#_Toc480727992)

[2.3.1 System Architecture: 13](#_Toc480727993)

[2.3.2 Initial Class Diagram: 13](#_Toc480727994)

[2.3.3 Natural Language Analysis (NLA): 13](#_Toc480727995)

[Conclusion: 15](#_Toc480727996)

# Introduction:

## Analysis Specification:

Analysis is of the crucial phase of the Software Development Life Cycle, for gathering the requirements needed for the development of any system. There are some of the stages needed to be done for the analysis specification: They are:

* Information gathering (Interviews, observations, Focus groups)
* Provide description of current system
* Provide description to solve problem
* Provide recommendation on solving the problem
* Prioritizing requirements
* Alternatives evaluation
* Provide the detailed specification

### Why we need to perform analysis:

Generally, we perform the analysis part in order to extract if any bug or error exist in the existing system or what sorts of the requirements may be necessary of essential for developing the new system. After the proper analysis is performed then the analyzer should have the full understand over an organization for which analysis is performed. In case of the existing system, if any fault occurs during the analysis then a complete set of solution should be provided to the existing problem along with an appropriate requirement specification.

The conceptual diagram for the analysis of the system in represented in the form of rich picture:

The requirements specification along with the use case and system architecture are describe below:

## Rich Picture:

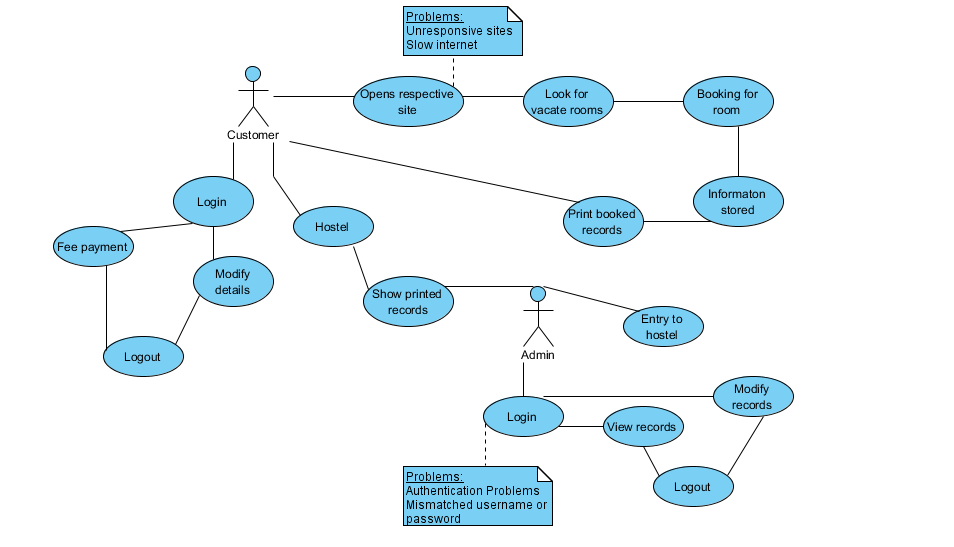


Fig: Rich picture (conceptual model)

# 2.1 Requirements:

It is the process of describing, analyzing, finding and finally documenting the type of the requirements. It covers the external interface requirements, non-functional requirements, and functional requirements, technical and operational. Among them functional and non-functional requirements are described and documented below:

## 2.1.1 Functional requirements:

ID: s1

Title: User booking (registration form)

Description: The new users or customers who are intended to roll into the hostel are allowed to book for the room in the hostel through the web portal.

Rational: To take the credentials information of the customer for authentication and allotting rooms.

ID: s2

Title: Friendly user interface

Description: It helps to enhance the interaction between the system and the user and the customer. Friendly user interface makes the system more reliable, moderate and easy to use.

Rational: Increase the user satisfaction and interaction.

ID: s3

Title: User login

Description: Existing user of the admin should be suitably logged in into the system with the use of the username and the password existing in the database. It should also flash the appropriate message when logged in.

Rational: To authenticate the users and admin privacy and confidentiality.

ID: s4

Title: Authentication

Description: It is the process of authenticating the user or the admin or the warden to differentiate the facilities and the authorization that is provided to each of them.

Rational: To manage the privacy and security.

ID: s5

Title: Administrative functions

Description: Only the warden or the admin should be able to inspect over the activities and the element of the hostel.

Rational: To perform the authenticate task only by the admin or warden of the hostel.

ID: s6

Title: Store hostellers information

Description: Users/customers information should be stored without any difficulties and error. Database must be capable to store the giant and bulk amount of information of booked and the added hostellers.

Rational: To store the details of the customers.

ID: s7

Title: Modify hostellers information

Description: Warden as well as hostellers should be able to edit or delete the information of the hostellers.

Rational: To update or delete the information.

ID: s8

Title: Automate fee payment

Description: Hostellers should be able to pay the fee monthly with automate system.

Rational: To automate the fee payment.

ID: s9

Title: Features panel

Description: It should be able to visualize the features and elements of the hostel.

Rational: To display the features/element in separate panel.

ID: s10

Title: Notice board panel

Description: It should be able to view the different information related to the hostel and can see the student administration division.

Rational: Informs the hostellers to the upgraded information to the hostel.

## 2.1.2 Non-Functional requirements:

Reliability: It shows how much the system would be able to show the response time done by the work of the end user and customer. It should be able to store the bulk amount of the information at an instant time and should be able to restore the system when it crashed.

Security: The system should be able to be prevent from the malicious attacks and the other accident, modification and breaking the confidentiality and integrity of the information.

Performance: The system should have the better performance speed based on the software utilization and the human interaction with the whole system. It should provide the greater flexibility during its performance.

Speed: The speed for the performance of the system should also be acceptable and appropriate that the user may not find it slow and monotonous to perform the task of the system due to the low speed.

Availability: The system should be made so that it would get available to its optimum level. The system should not get down in its working time and should be made 99% available with some features such as recovery, restart.

Maintainability: The system should be documented well or appropriate order of designing model should be consist, which would made it easier for the further maintenance if any fault occurs in the system.

Data Integrity: The system should be secured and made much stable that the information of the system or the hosteller’s information may not be easily changed or modified. The modification in the system should be strictly banned using different method such as cryptographic keys, restricting data transferring.

Usability: It shows how much the system shows the ease of access, and how fast to the cognitive user while using the any task of the system. It generally shows the speed of the system.

Safety: The system should contain the backup facilities, if in case the database crashes then there should be back up of the data and information contain by the database.

## 2.1.3 Prioritization:

It is done based on how much the requirements are much necessary in the development of the system. It is done using the MoSCoW technology, which tallies all the necessity of the requirements.

The prioritization technique is highlighted in short view below:

Mo- Must have: The most crucial requirements should fall under this criterion, which defines that if the requirement of the criteria are not fulfilled then system will surely go failure.

S- Should have: This show the requirements, which may be get, need in the future and is not solely important in the critical time. Therefore, it may not be done recently.

Co- Could have: This shows the requirements those are considerable but are not much necessary and, which adds extra functionality to the system.

W-Would have: This show the requirements, which is not as important to be released, or are at the least criterion.

The MoSCoW prioritization table is shown below:

## 2.1.4 MoSCoW prioritization:

|  |  |  |  |
| --- | --- | --- | --- |
| Requirement(functional) | MoSCoW | Requirement (non-functional) | MoSCoW |
| S1. User booking | Must have | Reliable | Must have |
| S2. Friendly user interface | Must have | Security | Must have |
| S3. User login | Must have | Performance | Should have |
| S4. Authentication | Must have | Speed | Should have |
| S5. Administrative function | Must have | Availability | Must have |
| S6. Store hostellers information | Must have | Maintainability | Could have |
| S7. Modify hostellers information | Must have | Data Integrity | Must have |
| S8. Automate fee payment | Should have | Usability | Should have |
| S9. Features panel | Should have | Safety | Must have |
| S10. Notice board | Could have |  |  |

# 2.2 Use Cases:

Use case diagram is an important tool in representing the broad interactions between different parts of our system and managing our abstractions. It is used in describing the actions done by the actor to the system. The diagrams does not show the interactions between the actors but between the actor and the system.

## 2.2.1 BENEFITS OF USE CASES:

* It focuses on the users of system not only in system.
* It provides the extensive description of whole system in single illustration.
* It shows the interactions between actors and system.
* It represents the functionality that supports the actors.
* It makes the non-technical stakeholders more understandable to system.

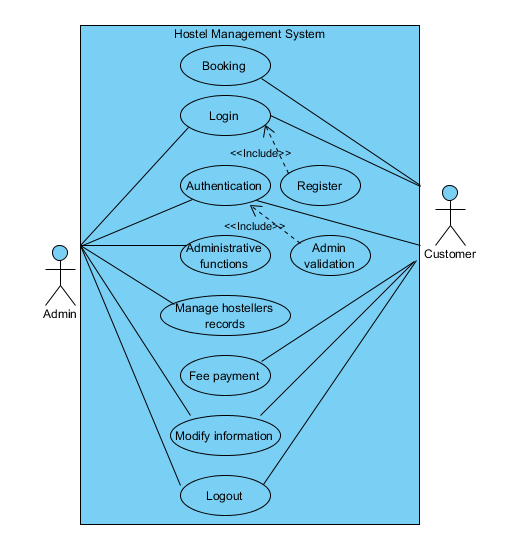


Fig: Use case

## 2.2.2 Scenario Description:

Scenario Description for Booking:

* Customer can book their entry for the hostel.
* Customer must fill necessary fields for their booking.
* Customer need to click the register button for their booking.
* Lastly, they will get register for the hostel environment.

Scenario Description for Login:

* Warden of the hostel can review the activities of the hostel.
* They need to provide username and password.
* They need to click the login button.

Scenario Description for Authentication:

* It provides the authentication to the system differentiating the admin and hostellers.
* Admin details is used for authentication.
* Login button is used for it.

Scenario Description for Administrative functions:

* Admins can only perform the core functions related to the system.
* They provide their username, password for authentication.
* They perform the functions after clicking login button.

Scenario Description for Managing hosteller’s records:

* Proper database is used for managing student’s records.
* Customer’s information during registration are stored in it.
* Database must be back up and maintained properly for managing all intellectual records.

Scenario Description for Fee payment:

* Hostellers are urged to pay the fee in time through automated system.
* All they have to check the box if they had paid the fee.
* They need to click the submit button to provide information to the admin of hostel.

Scenario Description for features frame:

* Customers and warden should provide their specific username and password.
* A single page opens showing the features of the hostel.

Scenario Description for Modifying information:

* Hostellers can view their information on records list.
* They can edit by clicking edit link.
* They need to provide new information and click the update button.
* Admin can simply delete the records.

Scenario Description to Logout:

* After performing all the functions users and warden, can logout from the functionality.
* They need to click the logout button or link.

# 2.3 Architecture:

It defines the whole structure of the system that how it will be developed, its architectural patterns and what sort of components will be in it during its development.

## 2.3.1 System Architecture:

The system architecture shows the skeleton of the system. It shows the interface of human with software or hardware, also overview the technical architectures that could be used in the development of the system and implementation. It defines the conceptual model of the system behavior and structure. Just like in the design pattern, we can also have the architectural patterns: MVC Model, View, and Controller.

## 2.3.2 Initial Class Diagram:

This diagram shows conceptual classes and the relationship between the classes. It is derived from the use case diagram of our system. It mainly contains classes and relationship but not methods and attributes.

## 2.3.3 Natural Language Analysis (NLA):

Here, different nouns, verbs and adjectives can be listed from our conceptual scenario, this process is called Natural Language Analysis (NLA).

In NLA, nouns reflects classes, verbs reflects operations or methods and adjectives reflects attributes.

Some of the class name and verb that might get helpful while making the class diagram are listed in tabular form.

|  |  |
| --- | --- |
| Classes (noun) | Verb (methods) |
| Booking, Customer, user, Hostel owner, Warden, Admin, Login | Monitor, book, progress, update, delete,  Generate, view, |

After listing all of the possible classes and verbs, candidates classes which doesn’t contain any relevant functionality, attributes and relationship were removed and a convenient candidate classes were extracted. Those classes were also removed which produced the same meaning sense and which were out of our project scope. The extracted and refined candidate classes are listed in tabular format below:

|  |  |
| --- | --- |
| Classes (noun) | Verb (methods) |
| Booking, customer, admin, login | Book, update, delete, view |

The proposed initial class diagram is shown below:

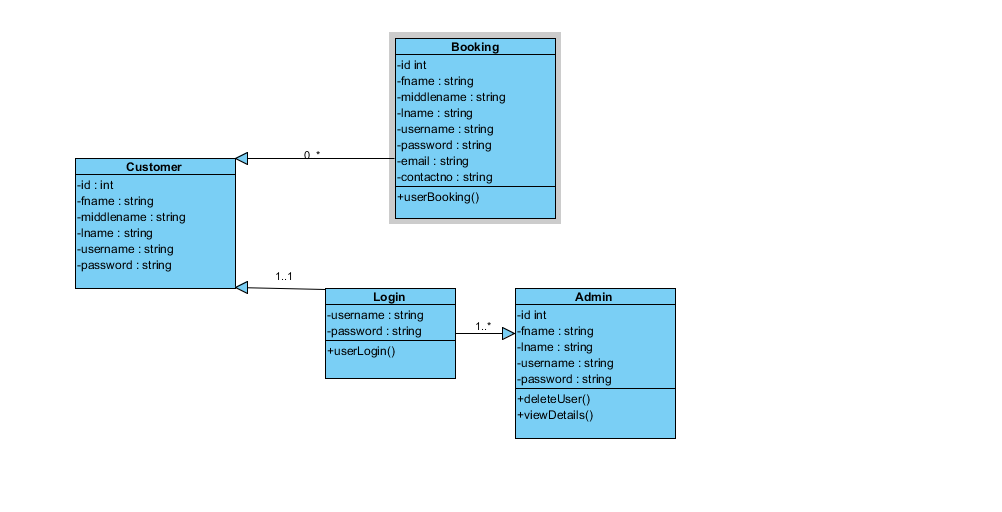


Fig: Class diagram

# Conclusion:

Ultimately, analysis specification was documented which might assists in planning and developing the new system. In this specification, we talked about the requirement of the analysis and its gathering methods. Then we listed all the functional and the non-functional requirements, which might need by the system during its development phase. We also listed the functionality components needed for the use case. Similarly, we choosed appropriate architecture for betterment of our system and hence created class diagram which came as a subsequent from our use case, which didn’t contain any suitable methods.