Contents

[1. Introduction 2](#_Toc521047251)

[2. Analysis 3](#_Toc521047252)

[2.1 Introduction to Analysis 3](#_Toc521047253)

[2.2 Requirement Gathering Techniques 3](#_Toc521047254)

[Feasibility study 6](#_Toc521047255)

[2.4 Analysis methodology: Soft systems Methodology SSM): 7](#_Toc521047256)

[Rich picture 8](#_Toc521047257)

[Root definition 8](#_Toc521047258)

[Conceptual model 8](#_Toc521047259)

[2.5 System Requirement Specification 9](#_Toc521047260)

[2.5.1 What is SRS? 9](#_Toc521047261)

[2.5.2 Types of SRS 9](#_Toc521047262)

[2.5.3 Functional requirement specification 9](#_Toc521047263)

[2.5.4 Non-functional Requirements 11](#_Toc521047264)

[2.4 Use case diagram 12](#_Toc521047265)

[2.7 NLA 15](#_Toc521047266)

[2.8 Initial class diagram 15](#_Toc521047267)

# Introduction

# Analysis

## 2.1 Introduction to Analysis

**What is Analysis?**

Analysis is the first phase of the system development lifecycle. Analysis is an interpretive process, breakdown of a system into its organized components to gain an insight about what works with what, what causes success or failure of something. This process focuses on critically viewing the system, defining the methodology used to build a system, taking feasibility study.

**Why analysis?**

System analysis is important in that it provides an avenue for solutions in the system through the various tasks involved in the analysis process.

* Better management and controls
* Risk Management
* Minimizing errors
* (http://www.ftias.com/benefits-system-analysis/#.W2KewigzbIU)

## 2.2 Requirement Gathering Techniques

These are the techniques to do market research to understand needs of customers and weakness of competitors.

* Questionnaires
* Survey
* Focus Group
* Brainstorming
* Observation
* Reverse Engineering
* Interview
* (<https://www.brighthubpm.com/project-planning/60264-techniques-used-in-business-requirements-gathering/>)

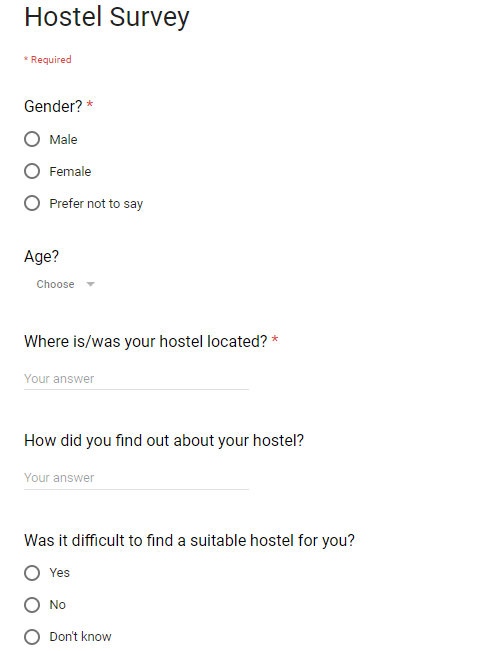
These are the techniques I focused on for requirement gathering.

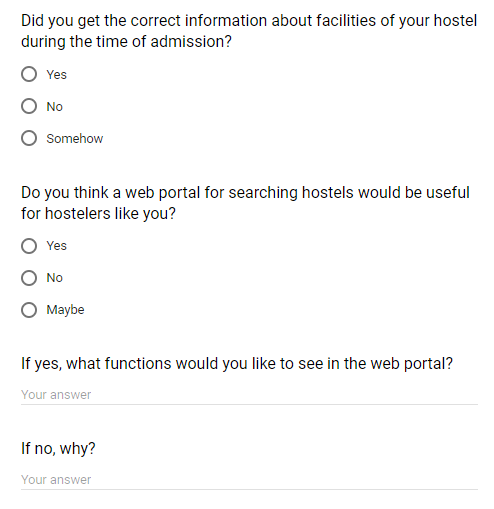
**Brainstorming**:

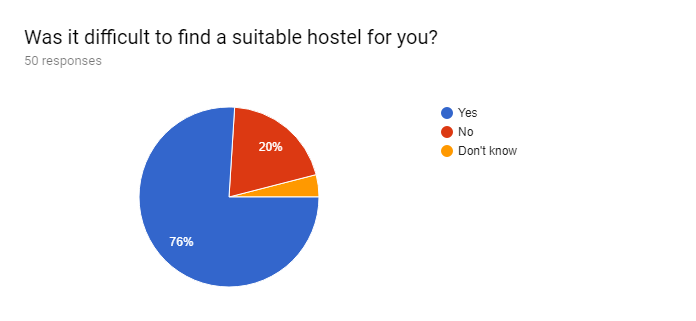
Brainstorming combines a relaxed, informal approach to problem solving with lateral thinking. Since this is an individual project, I started requirement gathering process by brainstorming, doing subtle research via internet and books. Individual brainstorming makes us think out of the box and discovers various discreet possibilities, generates creative solutions to the problem.

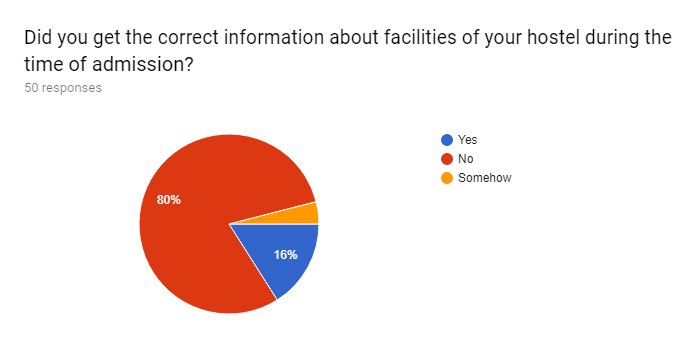
**Questionnaires:**

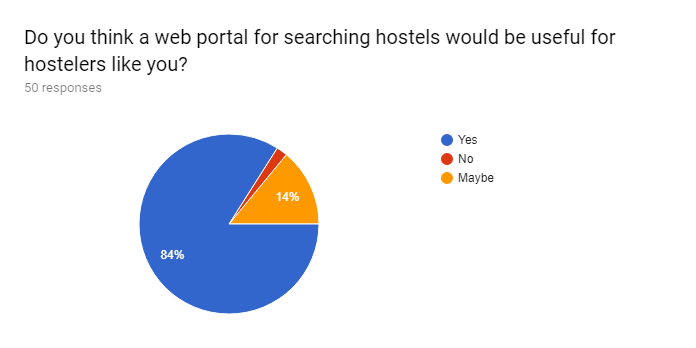
It is difficult to reach among individual audience and have one-to-one interviews with them. Questionnaires manage to gather answers from a larger audience at once and the responses can be sorted out to draw a simple conclusion from there.

I conducted a survey among 50 hostelers where they were asked some questions via google forms. I analyzed the feedbacks I got which helped me to make some important decisions. I have all the questions asked along with an overview of the analysis.









The charts prove that finding a hostel is indeed a tedious work to do and most of them agree that a web portal would be very easy for people like them.

## Feasibility study

Feasibility study is carried out to know weather or not a project is possible under certain circumstances. I performed the following studies to determine weather or not my project is feasible to be carried out.

* + 1. **Technical feasibility**: Technical feasibility is the procedure of determining whether the system can be developed using the available technical resources. Experiences available to use the technology, difficulty level is also determined. All these topics are covered under technical feasibility study.

This project is technically feasible as all the hardware and software necessities is already available to me. Also, a computer system with internet connection, which is vital for all the users is generally available very easily to most of the people now a days.

* + 1. **Schedule feasibility:** It is the likelihood of a project being completed on a provided time frame or within a deadline.

My project passes this feasibility test as it is bound to complete on time which is depicted via Gantt chart and WBS. This is an academic project which has to be completed within 3 months and there’s no possibility of exceptions. So, every steps of the system development will complete within provided time frame.

* + 1. **Economic feasibility:** Includes study of investments and revenues of a project to determine whether or not it is economically profiting project.

My project is made for academic purposes which doesn’t require a lot of financing and investing to be done at the moment. So, it is safe to say that it is economically feasible.

* + 1. **Cultural feasibility:** It includes validation of the system against the people and their cultures. If the system has negative impact on local and general culture, it is very likely to fail.

The project “Hostel finder” respects the sentiments of students and their parents and tries to solve the problems they face. It is not culturally offensive or questionable in any way.

* + 1. **Legal/ethical feasibility:** It defines legal implications and ethical considerations of the project. Determines whether the proposed system conflicts with legal requirements, e.g. a Data Processing system must comply with the local Data Protection Acts.

There is no such aspect in the system which conflicts with legal considerations. Similarly, the project is in compliance with ethical boundaries of the society and users. No such harm is intended.

(<https://www.brighthubpm.com/project-planning/56372-types-of-feasibility-studies/>)

## 2.4 Analysis methodology: Soft systems Methodology SSM):

SSM attempts to learn and appreciate the problem situations between the groups of stakeholders rather than set out to solve a problem that is pre-defined (Huaxia). SSM is a people focused approach to information system analysis that works in seven steps:

* Enter situation considered problematical
* Express the problem situation
* Formulate root definitions of relevant systems of purposeful activity
* Build conceptual models of the systems named in the root definitions
* Compare models with real world situations
* Define possible changes which are both possible and feasible
* Act to improve the problem situation

. **Features and advantages of SSM are:**

* Human activity is modelled.
* Enables input of user knowledge and skills
* Offers flexibility in the approach
* Searches for solution which is more than just technical
* Open discussions
* Doesn’t need professional practicing.

**Disadvantages of SSM:**

* May not be appropriate for complex systems and large organization
* Can be difficult to manage

**Why is it appropriate for this project?**

* We’re dealing with a situation that involves human problems.
* It is a small-scale project done individually.
* User, social, cultural issues need to be considered.

(<https://www.ifm.eng.cam.ac.uk/research/dstools/soft-systems-methodology/>)

### Rich picture

Rich Pictures are comprehensive, graphic (habitually hand-drawn) illustrations of what are stereotypically composite and ill-defined systems. It is an extremely simple way to conceptualize the scope/context of a system we are analyzing. A rich picture of how the system works originally is included below

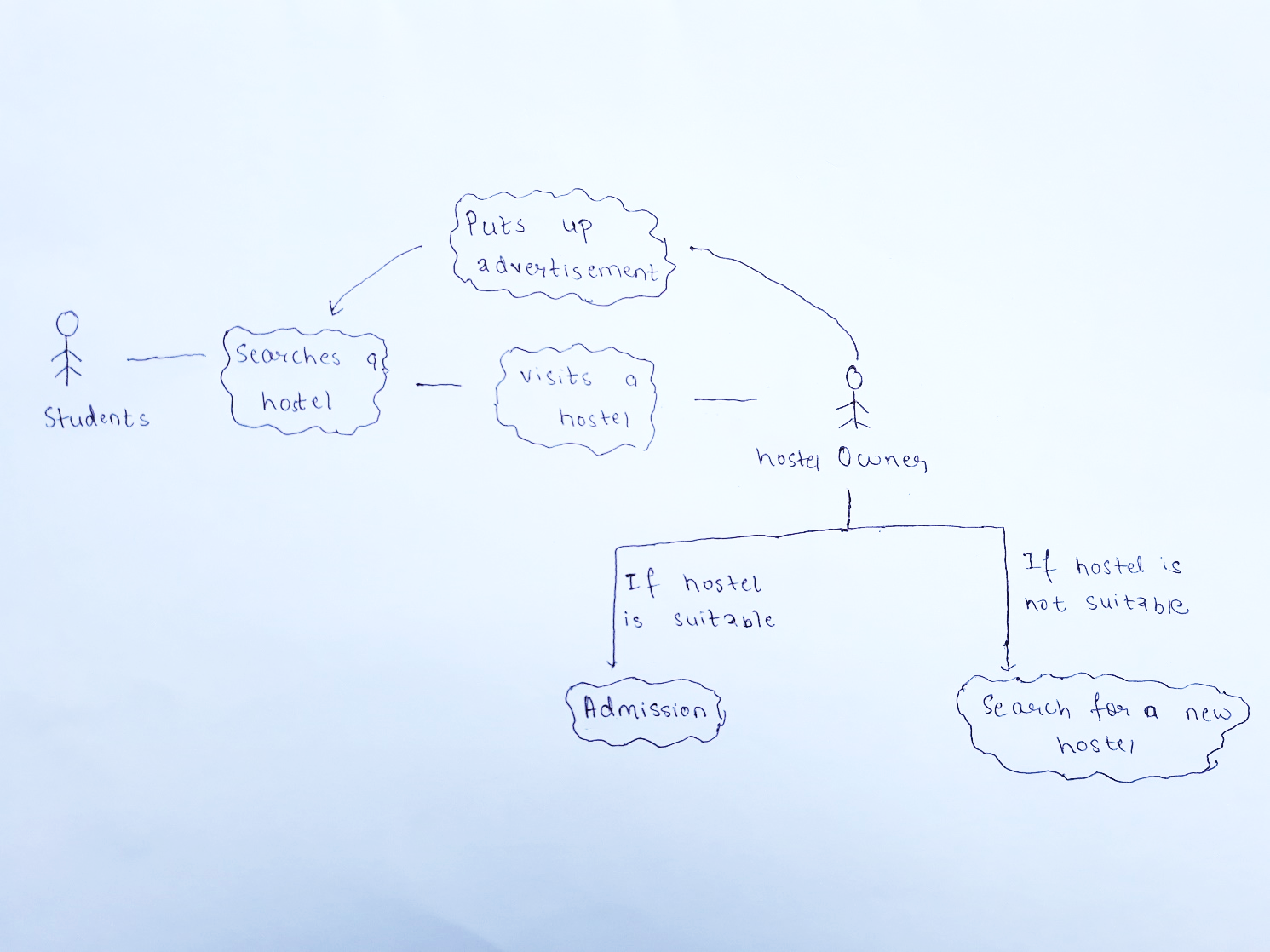


Figure : Rich picture diagram

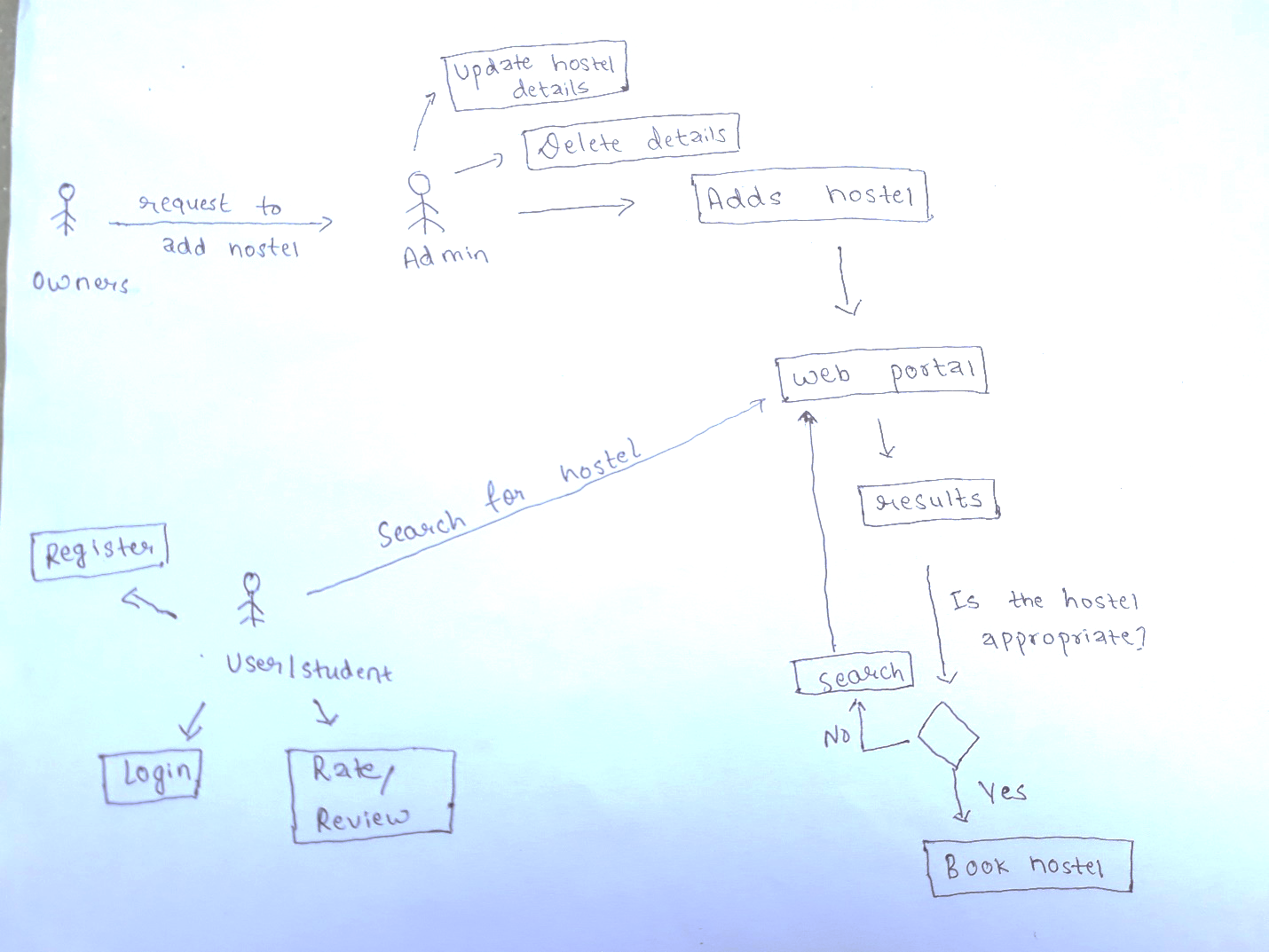


Figure : Conceptual Model

### Root definition

A Root Definition is a structured description of a system. It is a clear statement of activities which take place (or might take place) in the organization being studied.

Root definitions are powerful ways of building systems models which can help get to the heart of the problem issue. A powerful mnemonic – CATWOE – helps create root definitions which need to take account of and spell out: (https://businesschange.co.za/what-is-catwoe-analysis/)

|  |  |  |
| --- | --- | --- |
|  | Details | System overview |
| Clients | Who is the system operated for? | Hostelers/Students/Parents |
| Actors | Who is responsible for performing this transformation? | System admin |
| Transformation | What transformation will this system bring? | Making hostel searching process automatic and web based. |
| Worldview | What makes this transformation worthwhile? What is the bigger picture? | Reducing the mental turmoil migrant students face while searching for accommodation. |
| Owner | Who is authorized to make changes happen with the system? | System admin |
| Environment constraint | What constraints might have impact upon functioning of this system? | * No internet availability to many students * Less advertising * Financial constraints |

### Conceptual model

It is a composition of concepts involved in a system, used to represent a complex system in a simpler way.

Comparison between old and new system

|  |  |  |
| --- | --- | --- |
|  | Old system | New system |
| 1 | Students have to rely upon adverts or recommendation to search hostels | Students can browse the web portal to search hostels |
| 2 | Students have to visit hostel to find out about their features | Students will get information from the web portal |
| 3 | Student don’t get provision of advance booking or such. | There is a provision of advanced booking |
| 4 | It’s hectic to visit hostels one by one and choose the best one for you | It’s easier as there’s a ranking system. |

## 2.5 System Requirement Specification

### 2.5.1 What is SRS?

A software requirements specification (SRS) is a description of a software system to be developed. It labels functional and non-functional requirements, describes features and behaviors of the system. sSRS allows developers to be clear on the goals of the software and on what they should focus on. Furthermore, it allows them to:

* Save time on communication
* Minimize development efforts
* Gives the customer feedback
* Eliminate task duplication
* Facilitate the transfer to new users or to new machines
* Breaks problems down into parts
* Serves as the main document to verify the validation and testing processes

### 2.5.2 Types of SRS

Functional requirement specification:

**Functional requirement specification:** Describes a particular behavior of function of the system when certain conditions are met.

**Non-functional requirement specification:** Describes how a system should behave and what limits there are on its functionality. It lists its quality and attributes.

## 2.5.3 Functional requirement specification

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Functions | Data | Descriptions | Purpose |
| FR1 | Admin Register | Email, password | Admin should register first | To have an admin |
| FR2 | Admin Login | Email, password | Admin should log in to have full access to the system | To provide access to the system |
| FR3 | Register as a user | First Name, last Name,  Email,  hostel name | Users can create their accounts | To get hostel owners connected in the app |
| FR4 | Login as a user | Username/email, password | Registered users can login to the system | To fully access the app |
| FR5 | Add hostel | Hostel name, hostel location, prices | Owners can add details to their hostels | To add new details to the hostel |
| FR6 | Update hostel details | Hostel details | Details of the hostel can be updated by the owner. | To cope with the changes made in the hostel |
| FR7 | Add pictures | Pictures of the hostel | Pictures can be added | To see how hostel looks like |
| FR8 | Delete hostel details |  | Details can be deleted. | To delete the features of a hostel |
| FR9 | Rate a hostel |  | Students can rate a hostel they’ve lived in. | To compare hostels |
| FR10 | Leave a review on a hostel | Feedbacks and reviews of hostelers | Students can leave comments about the hostels they’ve lived in. | To get vies of hostelers on how good or bad is a hostel |
| FR11 | Search hostels by location | Location | Hostels can be searched based on the location. | To find hostels when you’re completely unknown about hostels in your area |
| FR12 | Search hostels by name | Hostel name | Hostel can be searched by typing their name | To find the hostels whose name you already know |
| FR13 | Notify about vacancies via SMS |  | Students seeking for vacancies will be notified via SMS when vacancy is available. | To be informed when the hostel you want to live in has vacancies |
| FR14 | Notification about vacancies in the app |  | Students seeking for vacancies will be notified in their account when vacancy is available. | To be informed when the hostel you want to live in has vacancies |
| FR15 | Display latest vacancies | Hostel details, vacancy amount | New vacancies will be displayed in the front page. | Provide information about vacancies |
| FR16 | Book hostels | Details of hostelers | Students willing to join a hostel can book when there is a vacancy. | Make it easier to students |
| FR17 | Remove booked vacancies |  | When vacancies are occupied, they are removed | To make vacancies information managed |
| FR18 | Hostel ranking |  | Hostels with highest ratings will be ranked as top 10 | To encourage hostels to be the best. |
| FR19 | Visitor count |  | Number of visitors who have visited the site | To review the popularity of the app |
| FR20 | Alert notifications |  | Inform hostel owners when their hostel if getting poor ratings | Encourage owners to constantly improve |
| FR21 | Delete user |  | Remove users when they are no longer part of the group | To remove users who are no longer part of the group |
| FR22 | User manual | FAQ, Help, Glossary | Guide to the user on how to use the system. | To make sure users know how to use the system. |

## 2.5.4 Non-functional Requirements

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Requirement | Details | Purpose |
| NF1 | Security | Protection of the system and user data | To secure data |
| NF2 | Reliability | Users should have the trust in the system even after using it for a long time. | To maintain user trust |
| NF3 | Availability | App is available and running every time | For easy access to the system every time. |
| NF4 | Usability | Navigation is easy to use for everyone | To make is easy to use for everyone |
| NF5 | Scalability |  |  |
| NF6 | Recoverability | Data can be recovered if ever erased | To recover valuable data |
| NF7 | Manageability |  |  |
| NF8 | Portability | Ability to be moved from one platform to another | To be able to use it in different devices |
| NF9 | Appearance | Aesthetics of the application | To make it visually pleasing |
| NF10 | Speed | Speed of loading and running | To save time |
| NF11 | Accuracy | Data should be accurate | To provide the right information |

**2.5.4 MoSCoW Prioritization**

Moscow prioritization is a technique that helps to understand and manage the priority of each task within the provided deadline. Here MoSCoW stands for:

|  |  |
| --- | --- |
| M-Must have | Minimum Usable Subset of requirement which project guarantees to deliver. |
| S-Should have | Important but not vital. Workaround is possible. |
| C-Could have | Valuable but no problem if dropped. |
| W-Won’t have | Won’t have this time |

(<https://www.agilebusiness.org/content/moscow-prioritisation-0>)

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Functions | Priority | Dependency |
| FR1 | Admin Register | M |  |
| FR2 | Admin Login | M |  |
| FR3 | Register as a user | M |  |
| FR4 | Login as a user | M | FR3 |
| FR5 | Add hostel | M | FR2 |
| FR6 | Update hostel details | M | FR2, FR5 |
| FR7 | Add pictures | M | FR2, FR5 |
| FR8 | Delete hostel details | M | FR2, FR5 |
| FR9 | Rate a hostel | M | FR3, FR4, FR5 |
| FR10 | Leave a review on a hostel | C | FR3, FR4, FR5, FR9 |
| FR11 | Search hostels by location | S | FR3 |
| FR12 | Search hostels by name | S | FR3 |
| FR13 | Notify about vacancies via SMS | M | FR3, FR4 |
| FR14 | Notification about vacancies in the app | M | FR3, FR4, |
| FR15 | Display latest vacancies | M |  |
| FR16 | Book hostels | M |  |
| FR17 | Remove booked vacancies | S |  |
| FR18 | Hostel ranking | S |  |
| FR19 | Visitor count | W |  |
| FR20 | Alert notifications | S |  |
| FR21 | Delete user | S | FR1, FR2, FR3, FR4 |
| FR22 | User Manual | M |  |
|  |  |  |  |
| NF1 | Security | M | FR1, FR2, FR3, FR4 |
| NF2 | Reliability | M | FR1, FR2, FR3, FR4, FR11, FR12 |
| NF3 | Availability | M | FR22 |
| NF4 | Usability | S | ALL |
| NF5 | Scalability | W | ALL |
| NF6 | Recoverability | W | FR1, FR3 |
| NF7 | Manageability | C | ALL |
| NF8 | Portability | W | ALL |
| NF9 | Appearance | C | ALL |
| NF10 | Speed | S | ALL |
| NF11 | Accuracy | S | ALL |

## Use case diagram

Use case is a UML diagram whose purpose is to demonstrate the different ways that a user might interact with the system. Use case diagrams are ideal for:

* Representing the goals of system-user interactions
* Defining and organizing functional requirements in a system
* Specifying the context and requirements of a system
* Modeling the basic flow of events in a use case
* Demonstrating the scope of your system

(https://www.tutorialspoint.com/uml/uml\_use\_case\_diagram.htm)

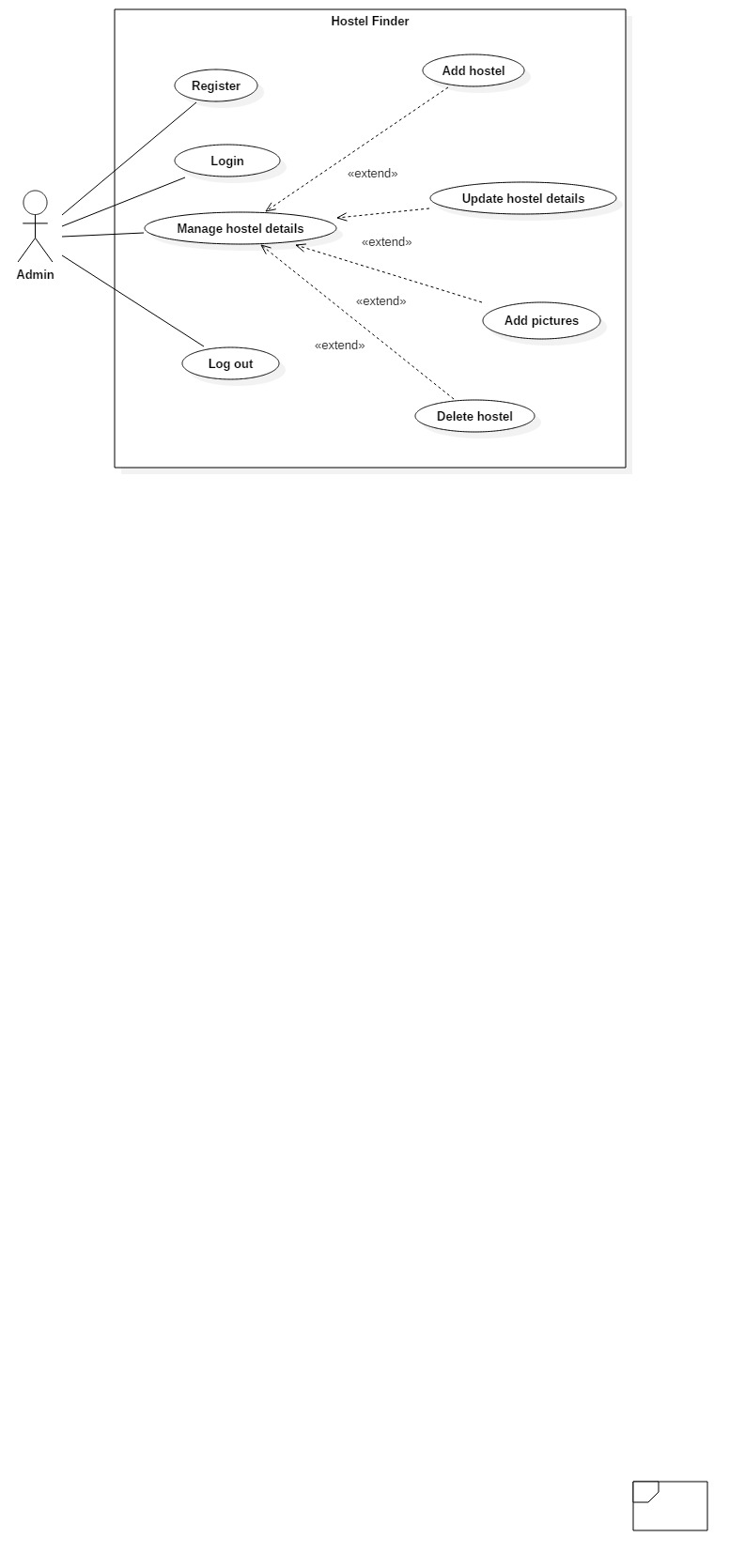


Figure : Use case of Admin

**Admin Login**

|  |  |
| --- | --- |
| Actors | Admin |
| Flow of events | Provides email and password |
| Alternative flow | A dialog box is displayed, if password and email don’t match  User will enter correct data. |
| Entry condition | Data entered must be valid |
| Exit condition | User will be displayed a success message. |
| Relationships | Includes: Email and password |

**Manage hostel details**

|  |  |
| --- | --- |
| Actors | Admin |
| Flow of events | Admin logs in to the system  Adds a new hostel/ Updates existing details/ deletes hostel  The data is stored in the database |
| Alternative flow |  |
| Entry condition | Data entered must be valid |
| Exit condition | Admin will be provided success message |
| Relationships | * Extends: Add hostel * Extends: Add pictures * Extends: Update details * Extends: Delete hostel |

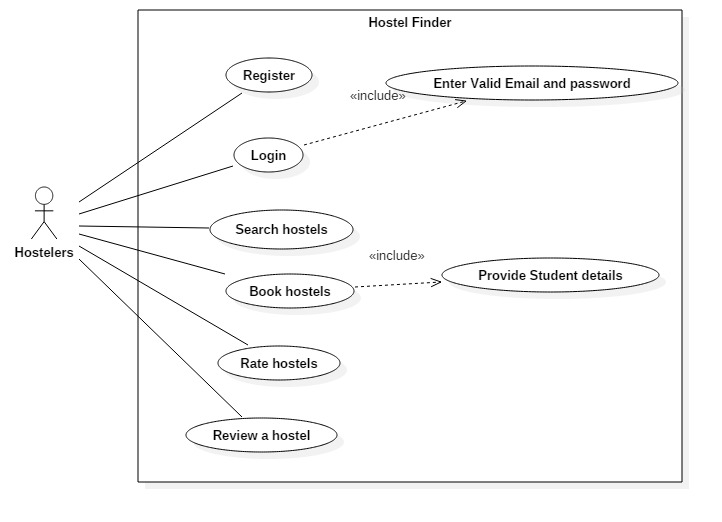


Figure : Use case of Hostelers

**User register**

|  |  |
| --- | --- |
| Actors | User |
| Flow of events | Clicks on register option  Provides Name, a unique email and password, Date of birth  Email is verified |
| Alternative flow | A dialog box is displayed, if data is not verified or authentic  User will enter correct data. |
| Entry condition | Data entered must be valid |
| Exit condition | User will be provided a success message |
| Relationships | Includes: Email and password |

**User login**

|  |  |
| --- | --- |
| Actors | User |
| Flow of events | Provides email and password |
| Alternative flow | A dialog box is displayed, if password and email don’t match  User will enter correct data. |
| Entry condition | Data entered must be valid |
| Exit condition | User will be provided a success message |
| Relationships | Includes: Email and password |

**Search hostel**

|  |  |
| --- | --- |
| Actors | User |
| Flow of events | User types in name of hostel, or preferred location  Search results are displayed |
| Alternative flow | If no result is found, a message is displayed |
| Entry condition | **-** |
| Exit condition | **-** |
| Relationships | **-** |

**Book hostel**

|  |  |
| --- | --- |
| Actors | Users |
| Flow of events | * User logs in to the system * Finds a suitable vacancy * Clicks on “book now” option * Fills in the form – Provides details |
| Alternative flow | * If any data is not invalid * A message box is displayed |
| Entry condition | Data entered must be valid |
| Exit condition | Admin will be provided success message |
| Relationships | * Extends: Add hostel * Extends: Add pictures * Extends: Update details * Extends: Delete hostel |

# 2.7 NLA

# 2.8 Initial class diagram