# Requirements Overview

Requirement analysis in [software engineering](http://en.wikipedia.org/wiki/Software_engineering), is a process of determining user expectations for a new or modified product, taking account of the possibly conflicting [requirements](http://en.wikipedia.org/wiki/Requirement) of the various [stakeholders](http://en.wikipedia.org/wiki/Stakeholder_(corporate)), analyzing, documenting, validating and managing  software or system requirements. These features, called requirements, must be quantifiable, relevant and detailed.

The requirements were distinguished into three parts: Functional requirements, Non-functional requirements and organizational and derived requirements.

**Functional requirements** – These are the activities that system must perform.

**Non-functional requirements** – These are the characteristics of the system other than activities that it perform or support.

**Organization or Derived requirements** – These are not directly provided by the customer but are generated through requirement analysis.

## Functional Requirements

In [software engineering](http://en.wikipedia.org/wiki/Software_engineering) (or [system engineering](http://en.wikipedia.org/wiki/System_Engineering)), a functional requirement defines a function of a [system](http://en.wikipedia.org/wiki/System) or its component.

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **DESCRIPTION** | **TYPE** | **PRIORITY** |
| FR1 | The prospect student should be able to continue as a guest and be able to make an appointment online with admissions office. | User Interface | M |
| FR2 | The prospect should also get an email notification of the appointment. | User Interface | M |
| FR3 | ITU’s students, staff and admin must be able to log into the system using their authorized credentials | User Interface | M |
| FR4 | Admin should be able to login to system and create user into the app and assign them a role. | User Interface | H |
| FR5 | Admin should be able to send the E-Appointment credentials to the users registered ITU email id |  | M |
| FR6 | Admin should be able to update the user information | User Interface | M |
| FR7 | Admin should be able to delete the user if they are no longer associated with ITU. | User Interface | M |
| FR8 | The staff should be able to login to the system after authentication and take action (approve, reject) on the appointment | User Interface | M |
| FR9 | The staff should be able to view the new appointments made by the students to them | User Interface | M |
| FR10 | The staff should provide the reason when rejecting any appointment. | User Interface | H |
| FR11 | Email notification must be sent to the student upon approval of rejection of the appointment |  | M |
| FR12 | The staff should be able to set up their appointment availability. | User Interface | H |
| FR13 | The staff should be able to edit their profile like mobile number and should be able to change password. | User Interface | M |
| FR14 | The student should be able to login to the system after authentication and make appointment. | User Interface | M |
| FR15 | The student should be able to view all the appointment made by them. | User Interface | M |
| FR16 | The student should be able to schedule an appointment after selecting the respective department or faculty. | User Interface | M |
| FR17 | The student should be able to edit their profile like mobile number and should be able to change password. | User Interface |  |

Table 1.1: Functional Requirements

## Non Functional Requirements

In [systems engineering](http://en.wikipedia.org/wiki/Systems_engineering) and [requirements engineering](http://en.wikipedia.org/wiki/Requirements_engineering), a non-functional requirement is a [requirement](http://en.wikipedia.org/wiki/Requirement) that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors.

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **DETAILS** | **TYPE** | **PRIORITY** |
| NF1 | The Online E-Appointment should provide database modification permission only to the admin and staff members after complete authentication and authorization procedures. The student should be able to view and update his profile and make appointment only after complete authentication and authorization. | Safety &  Security | M |
| NF2 | The system should provide an online web application that provides an interface, which is user friendly, simple and elegant. The interface should have a structured architecture, which should enable the user to easily navigate the system and quickly get the required information. | Usability | M |
| NF3 | The system should be available to all the users 24X7. The screening of the application should be done at a very fast pace of about 15-30 seconds. The response time of the E-Appointment should be very quick and fast enough to avoid data update collisions. | Performance | M |
| NF4 | The system should be able to provide real time information about the faculty availability. The system should be highly reliable to ensure first come first appointment of time, in the case of receiving multiple appointment for the same faculty. | Reliability | M |
| NF5 | The E-Appointment is a web based application and should be platform independent and support multiple devices. | Software and Hardware Constraints | M |

Table 1.2: Non Functional Requirements

## Organizational and Derived Requirements

In system engineering process, derived requirements are inferred or derived from user requirements. These requirements are developed through Requirement analysis.

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **DETAILS** | **TYPE** | **PRIORITY** |
| D1 | Appointment availability should be updated. | Usability | M |
| D2 | Appointment with the ISO should be made 24 hours in advance. | Business | M |
| D3 | The system should be able to operate in extreme temperatures and climatic conditions. | Performance | M |
| D4 | The system should allow the staff to accept or reject the appointment. | Usability | M |
| D5 | The web interface should adapt to screens of all sizes. | User Interface | M |

Table 1.3: Organizational and derived requirements

# Design

## System Design Overview

The E-Appointment system is a web-based widespread user-friendly system. The user can use web browser to access the system for activities like scheduling and viewing appointment. Another advantage is that E-Appointment system is platform independent; it works on all kinds of popular operating systems such as Unix, Linux, Windows or Mac. Front end UI will be developed using JSF and Java script and the server side application will be developed using JSP and JAVA. Platform independent feature increases the flexibility of the system, so that there is no need of new hardware that uses the E-Appointment system. The source code reusability will reduce the cost of application.

The UI data related to appointment availability is automatically refreshed and loaded from database.

## System Architecture

The architecture of the E-Appointment System is based on the three-tier architecture. This three-tier architecture mainly consists of three layers namely:

1. Presentation Tier
2. Business Tier
3. Data Access Tier

The Presentation Tier converts and displays information into a human legible form. This tier displays information related to services such as scheduling and viewing appointment. It communicates with the other tiers by outputting results to the browser/client tier and all the other tiers. The Business Logic tier is mainly responsible for information exchange between the user interface and the database of the project. The final layer of the three-tier architecture is the Data Access tier, which mainly consists of the Database servers. The information related to the users and appointment is stored and retrieved from here.

A simple representation of the three-tier architecture would be as follows:

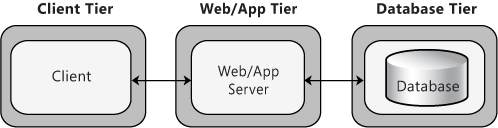


Figure 2.1: Three Tier Architecture

The three-tier architecture would be discussed in detail in the following sections:

**1. Presentation Tier**

Occupies the top level and displays information related to services available on a website. This tier communicates with other tiers by sending results to the browser and other tiers in the network. The presentation tier is mainly responsible for the user interface of the application, which deals with the presentation of data to the user.

**2. Business Logic Tier**

Also called the middle tier, logic tier, business logic or logic tier, this tier is pulled from the presentation tier. It controls application functionality by performing detailed processing. The business logic for the E-Appointment System would be present here. In this project, the Java classes would be performing the duty of the business logic. This is the layer, which is responsible for the information exchange between the user interface and the database. The E-Appointment system mainly consists of the Users, which can be further classified into the Admin, staff, Prospect Student and Current student of the System website.

**3. Database Tier**

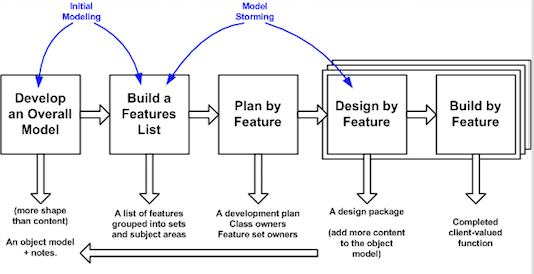
The database tier is the final and last tier of the three-tier architecture. All the data related to the E-Appointment System project is stored and retrieved from here. For this project we have used the MySQL Enterprise Edition Server to create the database. It is very easy to work with and makes creation and maintaining of tables very easy.

## Software Model

1. **Feature-Driven Development (FDD):**

1. Agile iterative development methodology

2. Based on breaking down the requirements into small client-valued pieces of functionality



# UML Diagrams

## Use Case Diagrams

Use case diagrams are behavior diagrams, which are used to describe a set of actions also referred to as use cases, which some system or systems (subject) will be able to perform in collaboration with many external users of the system actors. All the system actors or the stakeholders are provided with some observable and valuable result through every use case.

### Admin Use Case

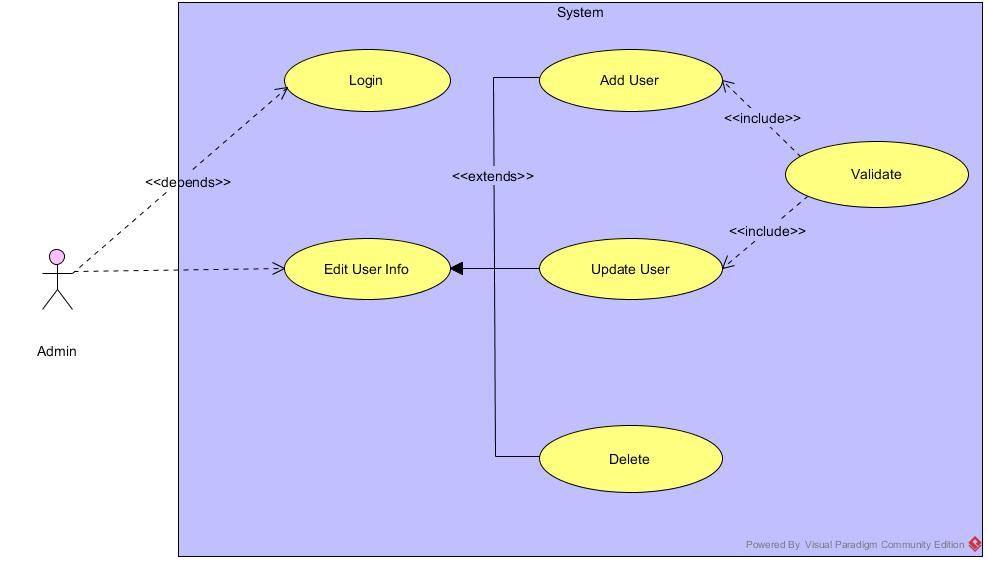


Figure 3.1.1: Admin Use Case

#### Admin Use Case Description

|  |  |
| --- | --- |
| **Use Case ID** | UC\_1 |
| **Use Case Name** | Admin Module |
| **Actor(s)** | Admin |
| **Goal in context** | The main purpose of this use case is to allow admin to add, update and delete user. |
| **Pre-conditions** | 1 .The system is running properly.  2. Admin logs into the system successfully |
| **Post-conditions** | 1. Different types id users will be created with a specific role such as student, staff  2. User credentials should be sent to the users ITU authorized email id. |
| **Basic Flow of Actions** | 1. The actor will open the E-Appointment user interface.  2. Then admin will login to the system user admin credentials.  3. Admin will have 3 sub tabs in the UI to add, update and delete user.  4. Admin can create users using the add user interface and assign them a role.  5. Admin can search a user by id or name in order to update the user info  6. Admin can delete the user by searching them by id or name. If they are no longer associated with ITU.  7. Can generate the report of the prospect student showing interest in the university programs. (Nice to have) |

### Prospect Student Use Case

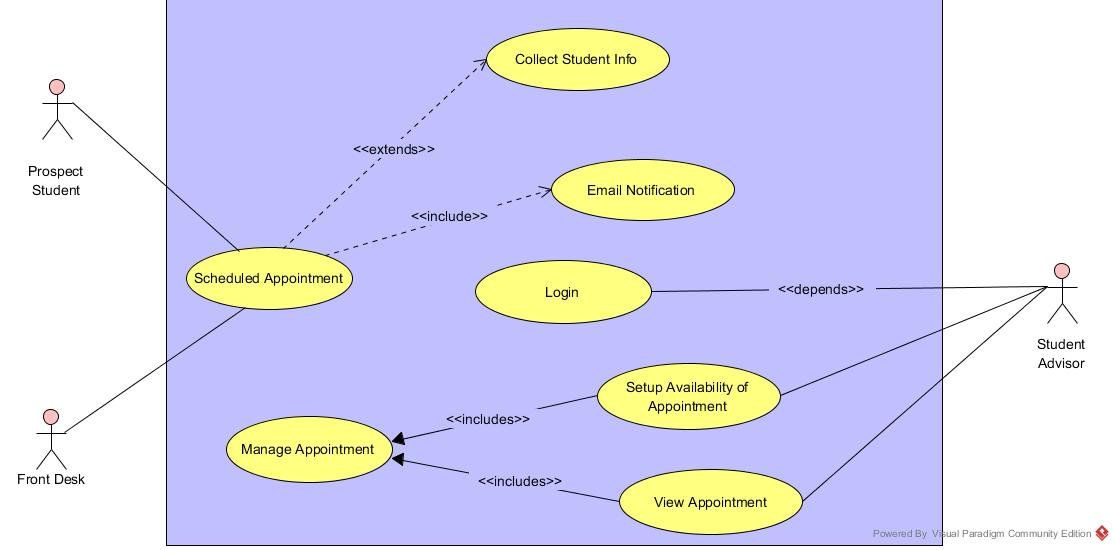


Figure 3.1.2: Prospect Student Use Case

#### Prospect Student Use Case Description

|  |  |
| --- | --- |
| **Use Case ID** | UC\_2 |
| **Use Case Name** | Prospect Student Use Case |
| **Actor(s):** | Prospect Student, Admission advisors, Front Desk |
| **Goal in context:** | The main purpose is to enable the prospect student make appointment online for their admission related queries. |
| **Pre-conditions:** | 1. The system is running properly.  2. The prospect student should be able to access the system as guest to make appointment. |
| **Post-conditions:** | 1. Appointment will be scheduled with the admission advisor.  2. An email notification will be sent to the prospect student for appointment confirmation. |
| **Basic Flow of Actions:** | 1.  The actor will open the E-Appointment system.  2. Prospect student can access the required part of the app by continuing as guest.  3. Then prospect student can schedule an appointment with admission advisors.  4. Prospect student can also call the front desk to make appointment on their behalf.  5. Front desk then uses the same system to schedule an appointment for prospect student.  6. Email notification is sent to the student for their appointment confirmation.  7. Admission advisors can login to the system using their authorized credentials.  8. Admission advisors can set their appointment availability by accessing the set up appointment page.  9. Admission advisors can view all the appointment scheduled. |

### Current Student/Staff Use Case

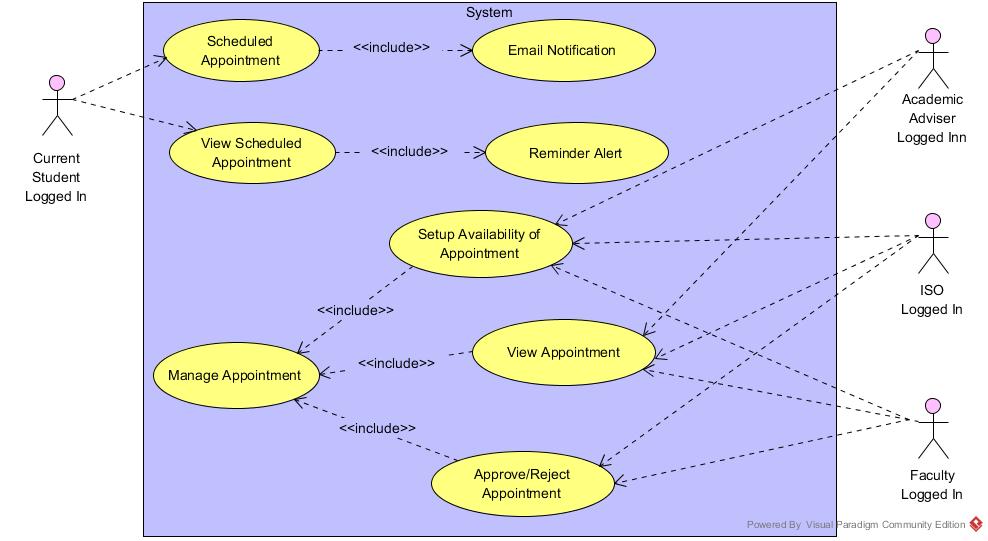


Figure 3.1.3: Current Student/Staff Use Case

#### Current Student Use Case Descriptions

|  |  |
| --- | --- |
| **Use Case ID** | UC\_3 |
| **Use Case Name:** | Current Student Use Case |
| **Actor(s):** | Current Student |
| **Goal in context:** | The main purpose of this use case is to allow student to make appointment online with their faculty, ISO and academic advisors. |
| **Pre-conditions:** | 1.The student should be logged into the system  2. The staff should be able to access the staff data from the database. |
| **Post-conditions:** | 1. Appointment should be scheduled with the respective department or team.  2. Email notification must be sent. |
| **Basic Flow of Actions:** | 1. The actor will log in to the E-Appointment system.  2. Then student can view all the appointment already scheduled.  3. Student can search for the respective department or faculty in order to schedule appointment.  4. Student can choose the staff and schedule an appointment.  5. Email notification should be sent on appointment scheduling.  6. Student can edit their info using Edit profile. |

#### Staff Use Case Description

|  |  |
| --- | --- |
| **Use Case ID** | UC\_3 |
| **Use Case Name:** | Staff Use Case |
| **Actor(s):** | Staff (ISO, Faculty, Academic advisors) |
| **Goal in context:** | The main purpose of this use case is to allow resident to view or update personal details and view payment history. Resident can also pay rent, reserve amenities or choose lease option through their profile. |
| **Pre-conditions:** | 1. The Staff should be logged into the system. |
| **Post-conditions:** | 1. Student will be notified by email if their appointment is accepted or rejected. 2. Student can see the current availability of the staff for appointment. |
| **Basic Flow of Actions:** | 1. The actor will log in to the E-Appointment system.  2. Then staff will view appointments and screen appointments.  3. Staff can accept or reject the appointment.  4. In case of the rejection, staff must provide a reason.  5. Email notification must be sent on approval or rejection of the appointment.  6. Staff can set their appointment availability by accessing the set up appointment page.  7. Staff can edit their info using Edit profile. |

### 

## Activity Diagram

An activity diagram is a graphical representation of business process or software algorithm as a series of actions. Activity diagram is basically a flow chart representing the flow form one activity to another activity. The activity can be described as an operation of the system. Activity diagrams are not only used for visualizing dynamic nature of a system but using forward and reverse engineering techniques also uses them to construct the executable system.

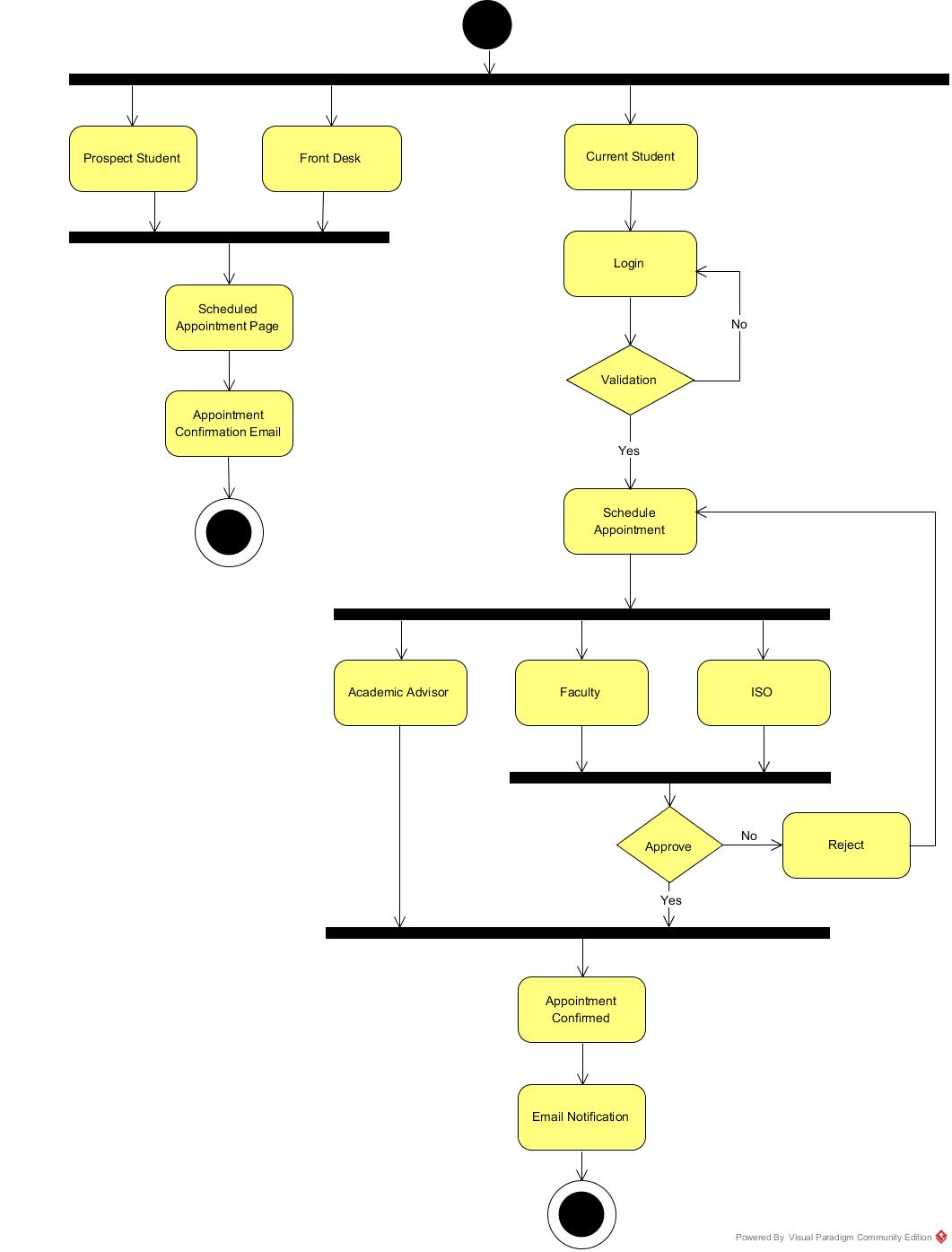


Figure 3.2: Activity Diagram