****

**COMSATS University Islamabad (CUI)**

Software Requirement Specification  
(SRS DOCUMENT)

for  
**< آسان کسان >**  
Version 1.0

***By***

**Muhammad Suleman CIIT/SP17-BCS-011/ISB**

**Hina Tajamal CIIT/SP17-BCS-038/ISB**

**Hamza Iftikhar CIIT/SP17-BCS-006/ISB**

***Supervisor  
M. Rashid Mukhtar***

*Bachelor of Science in Computer Science (2017-2021)*

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason for Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Application Evaluation History

|  |  |
| --- | --- |
| **Comments (by committee)**  **\*include the ones given at scope time both in doc and presentation** | **Action Taken** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Supervised by

Mr. M.Rashid Mukhtar

Signature\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Table of Contents**

Revision History i

Application Evaluation History ii

1. Introduction i

1.1 Purpose i

1.2 Scope i

7 Crop Remainder/Alert iv

8 Helpline iv

9 User management iv

3 Overall Description v

3.1 Product Perspective v

3.2 Operating Environment vi

3.3 Design and Implementation Constraints vi

4 Requirement Identifying Technique vi

4.1 Use Case Diagram vi

4.2 Use Case Description vii

5 Functional Requirements ix

5.1 Functional Requirement X ix

6 Non-Functional Requirements x

6.1 Usability x

6.2 Performance x

7 External Interface Requirements xi

7.1 User Interfaces xi

7.2 Software Interfaces xii

7.3 Hardware Interfaces xii

7.4 Communication Interfaces xii

8 Project Gantt Chart xii

9 References xiii

10 Plagiarism Report xiv

Attach the Plagiarism report of the targeted project scope document from the library staff of Turnitin tool (http://turnitin.com). xiv

# 

# Introduction

In the previous couple of decades, technology has made a profound impact on the lives of people. People prefer to use technology in every part of their life. So this document allows us to provide an application which will be helpful for farmers. There are millions of apps providing different solutions but the applications working on the farmer betterment are less in counts. In old times farming was very difficult as farmer have to relay on the old farming techniques for crop prediction and Crop production. We all know that farmers work very hard for the cultivation of their crops but unfortunately due to change in weather or natural disasters their crops destroy. Moreover farmers do not get paid according to how much they work on their fields to grow such beautiful crops. So this software will provide farmers to stand in front and communicate directly to the customers without the involvement of any third-party. Moreover our software provides all the liabilities that a farmer may need for better crop prediction and production. The crops are predicted based on temperature, humidity, UV index and soil moisture values of that area and a mechanism for predicting the disease of the crops and how to overcome it. Water alert system is also being introduced with the help of sensors to facilitate the farmers.

Our software will help the farmers by providing them a platform to sell, buy and even anticipate diseases in the crops along with the solutions to prevent it with sensible costs. They would be able to sell the crops without any interference of a third party. The farmers will also get information if the soil of an area is suitable for the crop production, by temperature, humidity, UV index and soil moisture values of that area. There are many software’s that provide these facilities but they are either too expensive or are based on a single purpose. As by developing this software we would be able to have abilities in Web, Android and Machine Learning.

## Purpose

Purpose of this SRS is to provide a full view and description about the functionalities and working of the system, and descriptions of all the use cases. Use case diagrams provides the view of user and system interaction. This SRS will provide the functional requirements i.e. response and processing of the system. Farmer will provide their location as input, and most suitable crops for a particular area will be shown as output. The farmer will give image of crops as input to detect disease in them as a output. Non-functional requirements are also the part of this SRS that tells about the usability, performance, security, availabity, scalability, reliability , compactibility and hence the quality of the system.

## Scope

This project permits the farmers to have a direct conversation with the customer and make a deal that comes up with good interest. Farmers are the stakeholders of this project .As they can get information about crops. Crop will be predicted based on temperature, humidity, UV index and soil moisture values of that area. They will get weather updates on daily basis in the form of notification. They will get water reminders for the crop based on sensors and moisture level. The nursery farmer will buy their crop seeds, fertilizers on an online agricultural cloud in less prices with no third-party interference. Farmer will get disease detection on their crop leaf by using image processing and they will be able to control these diseases.

### Crop Prediction:

There is a requirement for economical framework to anticipate and enhance the yield of crops. To take the full advantage of the soil kind, moisture, humidity, climate etc. Machine Learning algorithms will be used, which is becoming the hip of various agriculture areas, where straight applied math is not applicable. By using these algorithms, we can predict the soils which is best suited for a particular seed. In that way we can best utilize the data in order to get the most suitable seed that can grow in the particular area.

### Disease Detection:

Disease detection is used to detect the disease in the leaf of the crop. For this we use image processing algorithm that helps to predict the disease in the leaf of crop. Image processing algorithms require the picture of the leaf crop affected. Infection of the leaf could be of 3 types:

* Fungus infection (when the color is white)
* Bacterial infection(when the color is brown)
* Virus infection(when the color is yellow)

### Online agriculture cloud (آسان کسان)

Online agricultural cloud is a website that registers vendors and farmers and customers and allow them to communicate with each other. The website contains different variety of crops, pesticides and fertilizers that are managed by the vendor. The vendor is able to add and update the crops and fertilizers and make a deal with the farmer. Framer is also able to add and update his products and standing crops and make a deal with the customer. The website also helps the farmers to predict the crop disease.

#### Fertilizers:

Online agriculture Cloud contains different types of fertilizers that are updated and managed by the vendors.

#### Crops (grains and Nuts):

Online agriculture Cloud comprises of different types of grains that will be updated and managed by the vendors.

#### Pesticides Spray

Anti-Disease sprays will be available to protect and avoid the crops from the external disease.

#### Pay management

There will be two types of methods available to pay:

* Cash on delivery
* Credit card

### Vendor Management:

Vendor is a stake holder who oversees products of the customer. Vendor updates and sell products (fertilizers and grains of different types) on the website. There are more than one vendor, each with a unique login and are managed by the user management.

The sub modules of vendor are:

* Update products
* Insert products
* Delete products

### Customer Management

The customer here depicts the farmer. Farmer will buy the fertilizers and crops and user management manage the farmers. The farmers login with their own unique id, choose the payment method suitable and buy the product.as well as the crop buyers that could be of two types listed below:

### Hold crop buyers:

Hold crop purchasers purchase the yields in the wake of collecting on online agriculture cloud. They see the yields in the image and pick the ideal one. The nature of the yields is guaranteed.

### Standing crop buyer:

Standing crop purchasers will have the option to purchase the yield before reaping, on agriculture cloud. The maker will transfer its standing yields in the agriculture cloud so the purchaser can get it before collect. The maker will convey a few, or all, of the creation from a predetermined number of sections of land according to purchaser request. It don't guarantees the nature of the yields.

#### Live Chat

Farmers and Customers will be given a live chat facility so they can interact with each other, farmer will share product’s photos with the customers, so the customer will be satisfied by the product. In case of any issues they will resolve it online or in person if its possible.

#### Live Video

Farmers and customers will be given video call facility that will ease the customer to see the condition of standing crop from different side of the crop so they will purchase it without any issue or misconception.

### Farmer Management

There are two types of farmers who will sell their products to the customer and buy their product from the vendor. Farmer can add their crop product and standing crop details on the online agriculture market place. In this way farmer will deal directly with the market place and it will remove third party involvement. Each farmer will have unique login and are managed by user management.

The sub module of farmer is:

* + Update products
  + Upload products
  + Delete Products

### Crop Remainder/Alert

Crop remainder is a software that will alert the farmer when the water level of the crop will go below the required level. To detect the level of water nutrients that are required are: Phosphorus, Nitrogen and Moisture. The datasets that are required for the testing will be crop seed data, sunlit temperature, current year temperature.

### Helpline

A help line will be provided to the customers and farmer to see procedure about using the online agriculture market place. It will contains details about vendor products buying procedure, Farmers products procedure and contact detail in case of any problem, so they will contact with vendor online. The helpline will be managed by the vendors who will upload the procedure. It will allow the farmer and customer different features like

#### Report a problem

It will provide farmers and customers to report their problems regarding the application whether it is not working properly or server is down etc.

#### Give feedback

Farmers and customers will give their feedback about the application and suggestion regarding the betterment of the application.

#### Application rating

Customers and farmers will rate the application.

#### FAQs

It will contain question and their answers that will frequently arise in the customers and farmers mind regarding the basic information for users of a website.

#### Contact Details.

It will show the contact details and a message form so customers and farmers will interact with vendor online.

### User management

User management is used to manage the user access to the agricultural market. User can sign up as a vendor if he/she want to update products such as grains, fertilizers on the site or sign up as a farmer if he/she wants to learn from this site and want to buy grains fertilizers or needs information about the crop disease. They have to fulfill certain requirements such as filling registration form which will ask your details before signup. Signing up as a vendor provides 3 options which user can choose: update the current products, Add new products and Delete products. Farmers are also provided with 2 options i.e. buy products from the vendor online, and sell their products to the external market.

#### Sign up

#### Log in

#### Vendor sign up

* Register online
* Update products
* Add new products
* Delete product

#### Farmer sign up

* Register online
* Buy products from vendor.
* Upload agree business (his own products).
* Crop prediction about disease.

#### Customer Signup

* Register online.
* Buy Products.
* View Products.

# Overall Description

Our project is a web and mobile-based application that will provide the farmers a platform where they can buy and sell crops online based on their requirements. Our application will also provide crop prediction, disease detection, weather alerts and water alarms to farmers so that they will be able to manage their crops accordingly. Farmers can upload their products online so the customer will purchase them directly from the farmers.

Meanwhile our application shall also provide a platform for customers to buy standing crop or harvested crop and a platform for vendor to manage fertilizers, pesticides and help line for the farmers and customers.

## Product Perspective

Online agricultural cloud is a website that registers vendors and farmers and customers and allow them to communicate with each other. The website contains different variety of crops, pesticides and fertilizers that are managed by the vendor. The vendor is able to add and update the crops and fertilizers and make a deal with the farmer. Framer is also able to add and update his products and standing crops and make a deal with the customer. The website also helps the farmers to predict the crop disease.

The farmers will also get information if the soil of an area is suitable for the crop production, by temperature, humidity, UV index and soil moisture values of that area. The farmer will also be able to get information about the diseases in the crops and also get solutions of how to resolve them.

## Operating Environment

The application shall work correctly with the system having the following specifications:

* Windows 7+
* Intel core i3/i5.i7
* Core 2 duo
* Browser i.e Google, internet Explorer 8+
* Android version 5.0 plus
* IOS Version 4.0 plus

## Design and Implementation Constraints

* The system shall work on all systems.
* Internet should be available all the time to the user. Without internet user will not be able to access application.
* Person using the system must be literate.

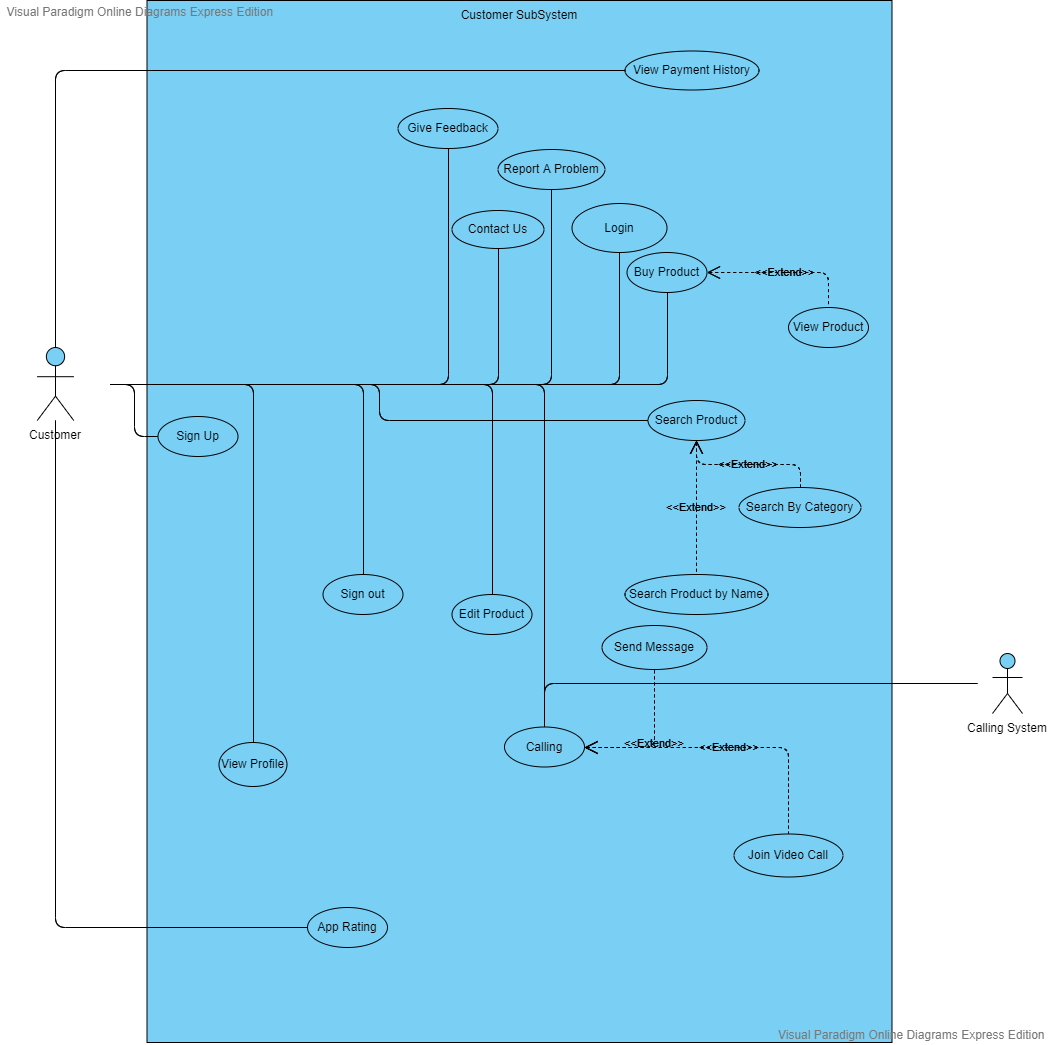
# Requirement Identifying Technique

This section describes the requirements identifying technique(s) which further help to derive functional requirements specification. The selection of the technique(s) will depend on the type of project. For instance,

* **Use case** is an effective technique for interactive end-user applications
* **Event- response tables** is for real-time system and
* **Storyboarding** for graphically intensive applications.

In addition to the above, the projects involving data warehouses, batch processes, hardware devices with embedded control software, and computationally intensive applications required to follow other suitable techniques. Such techniques are described further in Chapter 12, “A picture is worth 1024 words.” For documenting this section let consider identifying requirements through use case as an example.

## Use Case Diagram

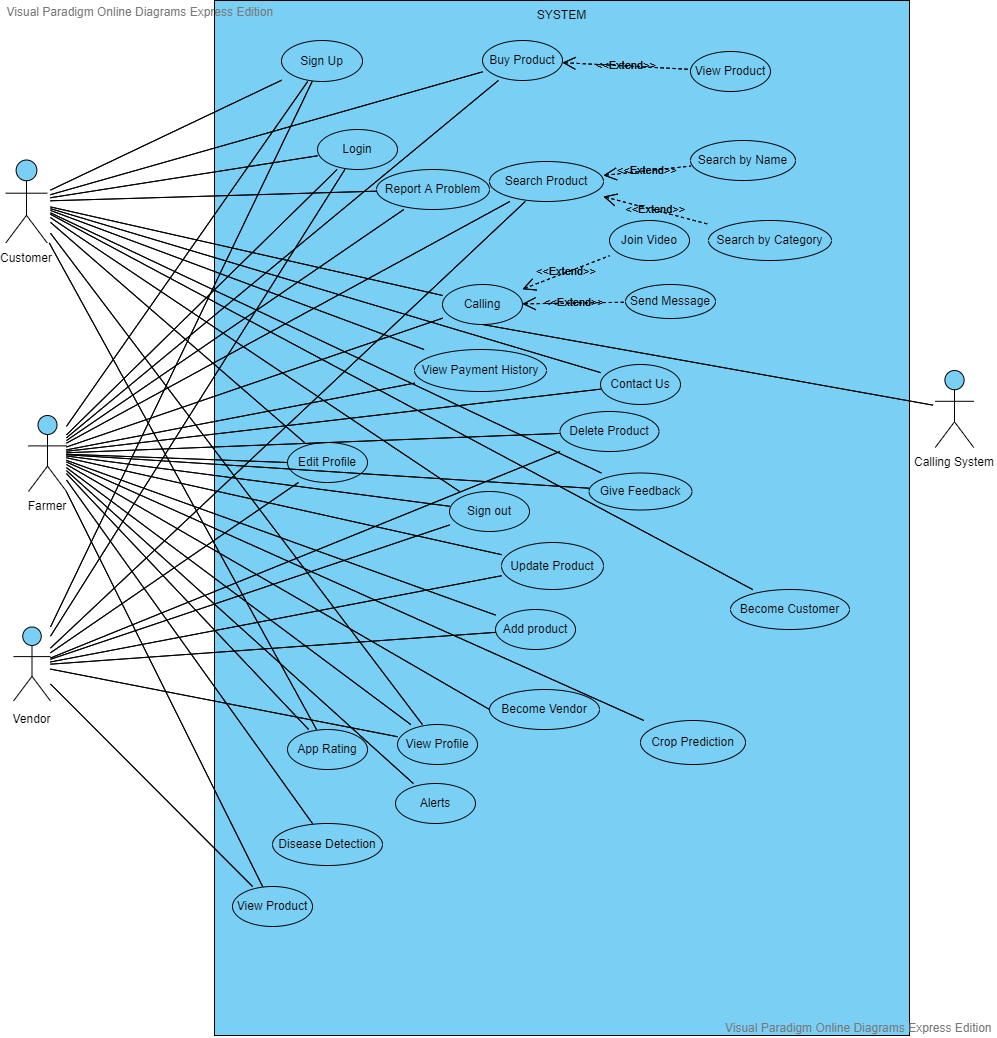
**3.1.1Customer**

**A close up of text on a white background

Description automatically generated3.1.2 Vendor**

A close up of text on a white background

Description automatically generated**3.1.3 Farmer**

**3.1.4 Complete UseCase Diagram**

## Use Case Description

The table below indicates a comprehensive use case template filled in with an example drawn from the Cafeteria ordering system (COS). (Appendix C) shows more sample use cases written according to this template. As with all templates, you don’t complete this from top to bottom, and you don’t necessarily need all the template information for every use case. The template is simply a structure in which to store the information you encounter during a use case discussion in an organized and consistent fashion. The template reminds you of all the information you should contemplate regarding each use case. For more detail see Chapter 8, “Understanding user requirements”

Table 1: Textual Description of <use case name>.

|  |  |
| --- | --- |
| **Use Case ID:** | Enter a unique numeric identifier for the Use Case. e.g. UC-1 |
| **Use Case Name:** | Enter a short name for the Use Case using an active verb phrase. e.g.  Order a Meal |
| **Actors:** | [An actor is a person or other entity external to the software system being specified who interacts with the system and performs use cases to accomplish tasks.] e.g.   |  |  |  |  | | --- | --- | --- | --- | | Primary Actor: | Patron | Secondary Actors: | Cafeteria Inventory System | |
| **Description:** | [Provide a brief description of the reason for and outcome of this use case.] e.g.  A Patron accesses the Cafeteria Ordering System from either the corporate intranet or external Internet, views the menu for a specific date, selects food items, and places an order for a meal to be picked up in the cafeteria or delivered to a specified location within a specified 15-minute time window. |
| **Trigger:** | [Identify the event that initiates the use case.]e.g.  A Patron indicates that he wants to order a meal. |
| **Preconditions:** | [List any activities that must take place, or any conditions that must be true, before the use case can be started.  PRE-1. Patron is logged into COS.  PRE-2. Patron is registered for meal payments by payroll deduction. |
| **Postconditions:** | [Describe the state of the system at the conclusion of the use case execution.  POST-1. Meal order is stored in COS with a status of “Accepted.”  POST-2. Inventory of available food items is updated to reflect items in this order.  POST-3. The remaining delivery capacity for the requested time window is updated. |
| **Normal Flow:** | [Provide a detailed description of the user actions and system responses that will take place during execution of the use case under normal, expected conditions.  1.0 Order a Single Meal  1. A patron asks to view the menu for a specific date. (see 1.0. E1, 1.0.E2)  2. COS displays the menu of available food items and the daily special.  3. Patron selects one or more food items from the menu. (see 1.1)  4. Patron indicates that the meal order is complete. (see 1.2)  5. COS displays ordered menu items, individual prices, and total price, including taxes and delivery charges.  6. Patron either confirms meal order (continue normal flow) or requests to modify meal order (return to step 2).  7. COS displays available delivery times for the delivery date.  8. Patron selects a delivery time and specifies the delivery location.  9. Patron specifies the payment method.  10. COS confirms acceptance of the order.  11. COS sends Patron an email message confirming order details, price, and delivery instructions.  12. COS stores order sends food item information to Cafeteria Inventory System, and updates available delivery times. |
| **Alternative Flows:**  **[Alternative Flow 1 – Not in Network]** | [Document legitimate branches from the main flow to handle special conditions (also known as extensions). For each alternative flow reference the branching step number of the normal flow and the condition that must be true for this extension to be executed. e.g.  1.1 Order multiple identical meals  1. Patron requests a specified number of identical meals. (see 1.1. E1)  2. Return to step 4 of normal flow.  1.2 Order multiple meals  1. A patron asks to order another meal.  2. Return to step 1 of normal flow.  Note: Insert a new row for each distinctive alternative flow. ] |
| **Exceptions:** | 1.0. E1 Requested date is today and the current time is after today’s order cutoff time  1. COS informs Patron that it’s too late to place an order for today.  2a. If Patron cancels the meal ordering process, then COS terminates use case.  2b. Else if Patron requests another date, then COS restarts use case.  1.0. E2 No delivery times left  1. COS informs Patron that no delivery times are available for the meal date.  2a. If Patron cancels the meal ordering process, then COS terminates use case.  2b. Else if Patron requests to pick the order up at the cafeteria, then continue with normal flow, but skip steps 7 and 8.  1.1. E1 Insufficient inventory to fulfill multiple meal order  1. COS informs Patron of the maximum number of identical meals he can order, based on current available inventory.  2a. If Patron modifies a number of meals ordered, then return to step 4 of normal flow.  2b. Else if Patron cancels the meal ordering process, then COS terminates use case. |
| **Business Rules** | Use cases and business rules are intertwined. Some business rules constrain which roles can perform all or parts of a use case. Perhaps only users who have certain privilege levels can perform specific alternative flows. That is, the rule might impose preconditions that the system must test before letting the user proceed. Business rules can influence specific steps in the normal flow by defining valid input values or dictating how computations are to be performed e.g.  BR-1 Delivery time windows are 15 minutes, beginning at each quarter-hour.  BR-2 Deliveries must be completed between 11:00 A.M. and 2:00 P.M. local time, inclusive.  Note: If you are maintaining the business rule in a separate table in SRS then only mention here their IDs. |
| **Assumptions:** | [List any assumptions.e.g. Assume that 15 percent of Patrons will order the daily special (Source: previous 6 months of cafeteria data). |

# Functional Requirements

This section describes the functional requirements of the system expressed in the natural language style. This section is typically organized by feature as a system feature name and specific functional requirements associated with this feature. It is just one possible way to arrange them. Other organizational options include arranging functional requirements by use case, process flow, mode of operation, user class, stimulus, and response depend on what kind of technique has been used to understand functional requirements. Hierarchical combinations of these elements are also possible, such as use cases within user classes. For further detail see Chapter 10 “Documenting the requirements”. Let consider the feature scheme as an example.

## Functional Requirement X

Itemize the specific functional requirements associated with each feature. These are the software capabilities that must be implemented for the user to carry out the feature’s services or to perform a use case. Describe how the product should respond to anticipated error conditions and to invalid inputs and actions. Uniquely label each functional requirement, as described earlier. You can create multiple attributes for each functional requirement, such as rationale, source, dependencies, etc. The following template is required to write functional requirements. For further detail see Chapter 11” Writing excellent requirements”.

Table 1: Description of FR-1

|  |  |
| --- | --- |
| **Identifier** | FR-1 |
| **Title** | Title of requirement |
| **Requirement** | Description of requirement which may be written either from the user or system perspective e.g.  If written in a **user perspective**  The [user class or actor name] shall be able to [do something] [to some object] [qualifying conditions, response time, or quality statement].  If written in a **system perspective**  [optional precondition] [optional trigger event] the system shall [expected system response] |
| **Source** | Where this requirement comes from (who originate it) |
| **Rationale** | The motivation behind the requirement |
| **Business Rule (if required)** | Any restriction, policy, the rule that the particular requirement must be fulfilled through its functional behavior |
| **Dependencies** | Requirements ID that is dependent on this requirement |
| **Priority** | High/Medium/Low |

# Non-Functional Requirements

This section specifies nonfunctional requirements other than constraints, which are recorded in section 2.3, and external interface requirements, which will appear in section 7. These quality requirements should be specific, quantitative, and verifiable. Chapter 14 “beyond functionality” presents more information about these quality attribute requirements and many examples. The following are some examples of documenting guidelines.

## Usability

Usability requirements deal with ease of learning, ease of use, error avoidance and recovery, the efficiency of interactions, and accessibility. The usability requirements specified here will help the user interface designer create the optimum user experience.

Example:

*USE-1: The COS shall allow a user to retrieve the previous meal ordered with a single interaction.*

## Performance

State specific performance requirements for various system operations. If different functional requirements or features have different performance requirements, it’s appropriate to specify those performance goals right with the corresponding functional requirements, rather than collecting them in this section.

Example:

*PER-1: 95% of webpages generated by the COS shall download completely within 4 seconds from the time the user requests the page over a 20 Mbps or faster Internet connection.*

# External Interface Requirements

This section provides information to ensure that the system will communicate properly with external components. If different portions of the product have different external interfaces, incorporate an instance of this section within the detailed requirements for each such portion.  
  
Reaching agreement on external and internal system interfaces has been identified as a software industry best practice. Place detailed descriptions of the data and control components of the interfaces in the data dictionary. A complex system with multiple subcomponents should use a separate interface specification or system architecture specification. The interface documentation could incorporate material from other documents by reference. For instance, it could point to a separate application programming interface (API) specification or to a hardware device manual that lists the error codes that the device could send to the software.

## User Interfaces

Describe the logical characteristics of each user interface that the system needs. Some possible items to include are

\* References to GUI standards or product family style guides that are to be followed.

\* Standards for fonts, icons, button labels, images, color schemes, field tabbing sequences, commonly used controls, and the like.

\* Screen layout or resolution constraints.

\* Standard buttons, functions, or navigation links that will appear on every screen, such as a help button.

\* Shortcut keys.

\* Message display conventions.

\* Layout standards to facilitate software localization.

\* Accommodations for visually impaired users.

Document the user interface design details, such as specific dialog box layouts, in a separate user interface specification, not in the SRS. Including screen mock-ups in the SRS to communicate another view of the requirements is helpful, but make it clear that the mock-ups are not the committed screen designs. If the SRS is specifying an enhancement to an existing system, it sometimes makes sense to include screen displays exactly as they are to be implemented. The developers are already constrained by the current reality of the existing system, so it's possible to know up front just what the modified, and perhaps the new, screens should look like.

## Software Interfaces

Describe the connections between this product and other software components (identified by name and version), including databases, operating systems, tools, libraries, and integrated commercial components. State the purpose of the messages, data, and control items exchanged between the software components. Describe the services needed by external software components and the nature of the intercomponent communications. Identify data that will be shared across software components. If the data-sharing mechanism must be implemented in a specific way, such as a global data area, specify this as a constraint.

## Hardware Interfaces

Describe the characteristics of each interface between the software and hardware components of the system. This description might include the supported device types, the data and control interactions between the software and the hardware, and the communication protocols to be used.

## Communication Interfaces

State the requirements for any communication functions the product will use, including e-mail, Web browser, network communications protocols, and electronic forms. Define any pertinent message formatting. Specify communication security or encryption issues, data transfer rates, and synchronization mechanisms.

# Project Gantt Chart

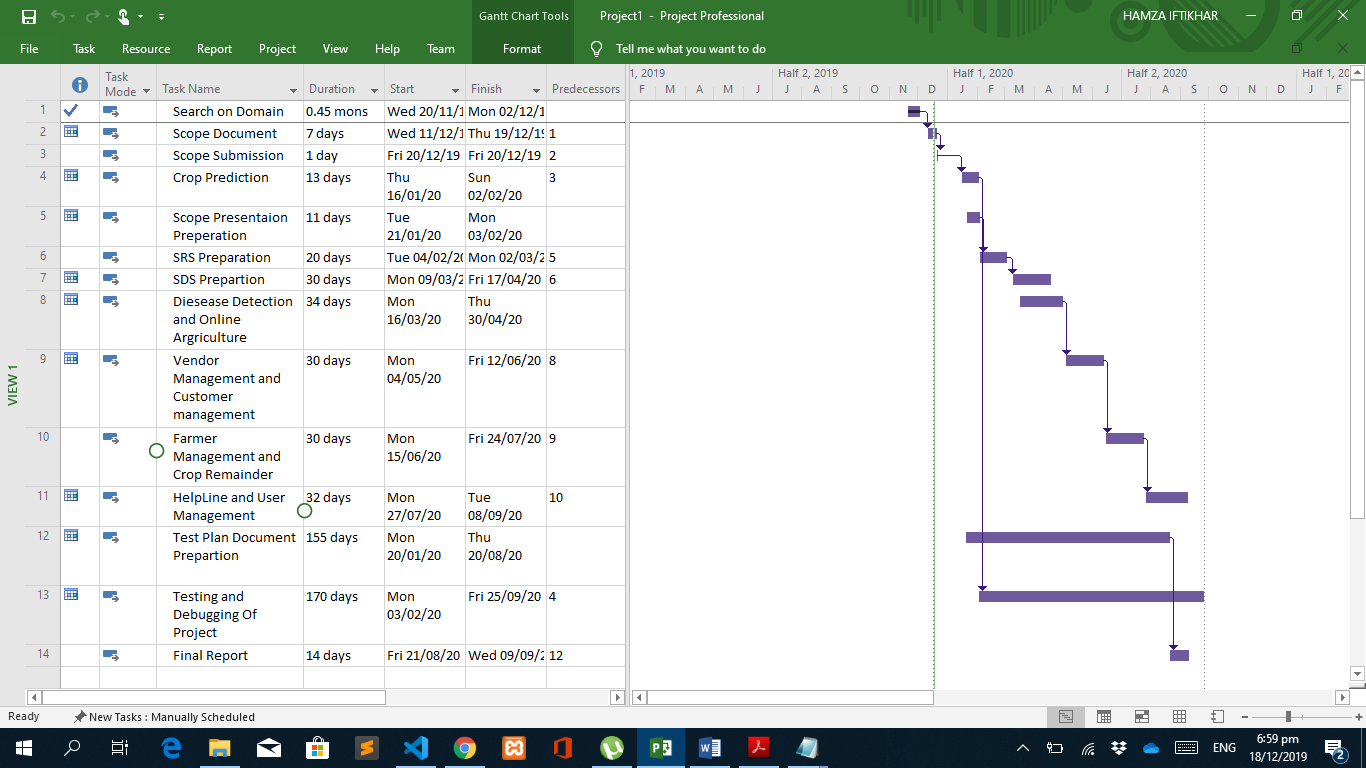


Figure 1: Gant Chart of the Targeted Project.

# References

List any documents or other resources to which this SRS refers, if any. These might include user interface style guides, standards, system requirements specifications, interface specifications, or the SRS for a related product. The following are a few examples of different resources i.e. journal article, book, and website.

* 1. Lyda M.S. Lau, Jayne Curson, Richard Drew, Peter Dew and Christine Leigh, (1999), Use Of VSP Resource Rooms to Support Group Work in a Learning Environment, ACM 99, pp-2. **(Journal paper example)**
  2. Hideyuki Nakanishi, Chikara Yoshida, Toshikazu Nishmora and TuruIshada, (1996), FreeWalk: Supporting Casual Meetings in a Network, pp 308-314 **(paper on web)** http://www.acm.org/pubs/articles/proceedings/cscw/240080/p308-nakanishi.pdf
  3. Ali Behforooz& Frederick J.Hudson, (1996), Software Engineering Fundamentals, Oxford University Press. Chapter 8, pp255-235. **(book reference example)**
  4. Page Author, Page Title, http://www.bt.com/bttj/archive.htm, Last date accessed**. (web site)**

# Plagiarism Report

# Attach the Plagiarism report of the targeted project scope document from the library staff of Turnitin tool (<http://turnitin.com>).