# CHAPTER-01: INTRODUCTION

This chapter is a part of our software requirement specification for the project “Programming Platform”. In this chapter we focus on the intended audience for this project.

## 1.1 INTRODUCTION

This document briefly describes the Software Requirement Analysis of “Programming Platform”. It contains functional, non-functional and supporting requirements and establishes a requirements baseline for the development of the system. The requirements contained in the SRS are independent, uniquely numbered and organized by topic. The SRS serves as an official means of communicating user requirements to the developer and provides a common reference point for both the developer team and the stakeholder community. The SRS will evolve over time as users and developers work together to validate, clarify and expand its contents.

## 1.2 INTENDED AUDIENCE

This SRS is intended for several audiences including project managers, designers, developers, and testers.

The project managers of the developer team will use this SRS to plan milestones and a delivery date, and ensure that the developing team is on track during development of the system.

The designers will use this SRS as a basis for creating the system’s design. The designers will continually refer back to this SRS to ensure that the system they are designing will fulfill the customer’s needs.

The developers will use this SRS as a basis for developing the system’s functionality. The developers will link the requirements defined in this SRS to the software they create to ensure that they have created a software that will fulfill all of the customer’s documented requirements.

The testers will use this SRS to derive test plans and test cases for each documented requirement. When portions of the software are complete, the testers will run their tests on that software to ensure that the software fulfills the requirements documented in this SRS. The testers will again run their tests on the entire system when it is complete and ensure that all requirements documented in this SRS have been fulfilled.

## 1.3 CONCLUSION

This analysis of the audience helped us to focus on the users who will be using our analysis. This overall document will help each and every person related to this project to have a better idea about the project.

# CHAPTER-02: INCEPTION OF PROGRAMMING PLATFORM

## 2.1 INTRODUCTION

Inception is the beginning phase of requirements engineering. It defines how a software project gets started and what the scope and nature of the problem to be solved is. The goal of the inception phase is to identify concurrent needs and conflicting requirements among the stakeholders of a software project. At project inception, we establish a basic understanding of the problem, the people who want a solution, the nature of the solution that is desired and the effectiveness of preliminary communication and collaborations between the other stakeholders and the software team. The purpose of the document is to represent a short description of “Programming Platform”

To establish the groundwork we have worked with the following factors related to the inception phases:

* List of stakeholders
* Recognizing multiple viewpoints
* Working towards collaboration
* Requirements questionnaire

### 2.1.1 LIST OF STAKEHOLDERS

Stakeholder refers to any person or group who will be affected by the system directly or indirectly. Stakeholders include end-users who interact with the system and everyone else in an organization that may be affected by its installation. At inception, a list of people who will contribute input as requirements are elicited. To identify the stakeholders we tried to find out those following questions:

* Who is going to use the platform?
* Who is going to manage the platform?
* Who is going to get benefit from the platform?
* Who is going to give feedback to us?

We identified the following stakeholders for our “Programming Platform”.

* Programming Platform Manager:
* Contest Regulator:
* Problem Solver:

### 2.1.2 RECOGNIZING MULTIPLE VIEWPOINTS

Different stakeholders achieve different benefits from the system. Consequently, each of them has a different view of the system. So we have to recognize the requirements from multiple points of view, as well as multiple views of requirements. Assumptions are given below:

Programming Platform Administrator viewpoints:

* User friendly and efficient system
* Strong Authentication
* Strong security system

Contest Regulator viewpoints:

* User friendly and efficient system
* Efficient contest management system
* Giving live rank of contenders of any contest
* Minimum to process verdict on solution submission
* Users are supposed to view her/his own source code

Problem Solver viewpoints:

* User friendly and efficient system
* Fascinating outlook
* Minimum time on returning verdict
* User profile statistics of user submission in contest
* Source code reading opportunity of provided source code

### 2.1.3 WORKING TOWARDS COLLABORATION

Every stakeholder has their own requirements. There are some common and conflicting requirements of our stakeholder. That’s why we followed the following steps to merge these requirements-

* Find the common and conflicting requirements
* Categorize them
* List the requirements based on stakeholder’s priority points
* Make final decision about requirements

Common requirements:

* User friendly and efficient platform
* Fascinating outlook
* Minimum time on returning verdict
* User profile statistics of user submission in contest
* Giving live rank of contenders of contests
* Source code reading opportunity of a user’s own source code

Conflicting requirements:

* Limited development time

Final requirements: We finalize the following requirements based on stakeholder’s priority point:

* User friendly and efficient platform
* Fascinating outlook
* Minimum time on returning verdict
* User profile statistics of user submission in contest
* Giving live rank of contenders of contests
* Source code reading opportunity of a user’s own source code

### 2.1.4 REQUIREMENTS QUESTIONNAIRE

We first ask the stakeholder some context free questions to understand the project’s overall performance and goals. These questions are mentioned in section 2.1.1. These questions help us to identify the stakeholders of the project. Then we ask our next set of questions to better understand the problem and take stakeholder’s opinion about the solution.

## 2.2 CONCLUSION

The Inception phase helped us to establish basic understanding about the “Programming Platform”, identify the stakeholders who will be benefited if this system becomes automated, define the nature of the system and the tasks done by the system, and establish a preliminary communication with our stakeholders.

In our project, we have established a basic understanding of the problem, the nature of the solution that is desired and the effectiveness of preliminary communication and collaboration between the stakeholders and the software team. More studies and communication will help both sides (developer and client) to understand the future prospect of the project. Our team believes that the full functioning document will help us to define that future prospect.

# 

# CHAPTER-03: ELICITATION OF PROGRAMMING PLATFORM

After discussing on the inception phase, we need to focus on Elicitation phase. So, this chapter specifies the Elicitation phase.

## 3.1 INTRODUCTION

Requirements Elicitation is a part of requirements engineering that is the practice of gathering requirements from the stakeholders. We have faced many difficulties, like understanding the problems, making questions to the stakeholders, problems of scope and volatility. Though it is not easy to gather requirements within a very short time, we have surpassed these problems in an organized and systematic manner.

## 3.2 ELICITING REQUIREMENTS

We have seen Question and Answer (Q&A) approach in the previous chapter, where the inception phase of requirement engineering has been described. Requirements Elicitation (also called requirements gathering) combines problem solving, elaboration, negotiation and specification. The collaborative working approach of the stakeholders is required to elicit the requirements. We have finished the following tasks for eliciting requirements-

* Collaborative Requirements Gathering
* Quality Function Deployment
* Usage scenarios
* Elicitation work products

### 3.2.1 COLLABORATIVE REQUIREMENTS GATHERING

We have met with the stakeholders of this project in the inception phase such as administrator and problem solvers. Many different approaches to collaborative requirements gathering have been proposed by the stakeholders. To solve this problem we have met with the stakeholders again to elicit the requirements. A slightly different scenario from these approaches has been found.

* The meeting were conducted with an administrator of an online judge and programming problem solvers. They were questioned about their requirements and expectations.
* They were asked about the problems they were facing the current manual system.
* Lastly we selected our final requirement list from the meetings.

### 3.2.2 QUALITY FUNCTION DEPLOYMENT

Quality Function Deployment (QFD) is a technique that translates the needs of the customer into technical requirements for software. It concentrates on maximizing customer satisfaction from the software engineering process. So we have followed this methodology to identify the requirements for the project. The requirements, which are given below, are identified successfully by the QFD.

#### 3.2.2.1 NORMAL REQUIREMENTS

Normal are generally the objectives and goals that are stated for a product or system during meetings with the stakeholders. The presence of these requirements fulfills stakeholders’ satisfaction. The normal requirements of our project-

* User-friendly design.

#### 3.2.2.2 EXPECTED REQUIREMENTS

These requirements are implicit to the product or system and may be so fundamental that the customer does not explicitly state them. Their absence will be a cause for significant dissatisfaction. Below the expected requirements are described

* ////

#### 3.2.2.3 EXCITING REQUIREMENTS

These features go beyond the customer’s expectations and prove to be very satisfying when present. The exciting requirements are described below

### 3.2.3 USAGE SCENERIO

Programming Platform is an automated System for the following purposes-

* Authentication
* Problem Set Management
* Programming Contest Management
* Solution Assessment
* User Profile Management

##### Authentication

Programming platform has two types of users.

* Administrator
* Problem solver

At the time of installation of this system an administrator account will be created who will maintain the system. For registration any type of user has to provide these information

* Username/Email
* Password
* Recovery Pin
* Institute

User’s password can be at most fifteen characters and pin four characters. Only authenticated user can enter to the system.

A user can log into the system entering her/his username/email and password. If provided username/email and password matches, user can enter to the system. Otherwise an error message will be generated.

If a user forget her/his password then she/he can recover her/his account. For user account recovery the system will ask for her/his username/email and pin number. If provided email and pin number matches then she/he will be able to see her/his password. Then using that password she/he can enter into the system.

##### Problem Set Management

One of the basic features of a Programming Platform is to give the programmers the opportunity to practice programming problems to enhance their coding skills. A programming platform is supposed to have some predefined problems which have been set by the administrator.

The problem set will hold the programming problems that have been set in previously arranged contests. This problem set will allow the users to practice contest problems even the contest duration is over.

All the problems will be set by the administrator. Users can surf through the problem set and solve the problems of their choice.

##### Programming Contest Management

Arrangement of contest is the greatest feature of a programming platform or judge. An administrator can arrange programming contests.

To arrange a contest, the administrator needs to set problems. The administrator will set a problem by uploading a problem description file which will be a portable document format (pdf) file. The administrator will also upload a test input file and a solution file corresponding to every problem. The duration of the contest will also be set by the administrator.

The users can participate in any of contest. Users will be ranked based on their performance in the contest.

After the contest, the problems of the contest will also be added to problem set.

##### Solution Assessment

A user can submit the solution of a problem through an input text field which will be underneath that particular problem. At the same time s/he will also select the programming language. The available programming language in this case will be C, C++ and Java.

The submitted solution will be first compiled by the system based on the programming language s/he has selected. Then this solution will be run against the pre-set test input file. The output of the given input set will be matched against the solution file which has also been set by the administrator. If the output matches with the inputted solution file, the solution code will be accepted. Otherwise, the solution will be not accepted. In response to the submitted solution code, the system will provide one of the following verdicts:

* Accepted *: If the output satisfies the provided solution file.*
* Wrong answer : *If the output does not satisfy the provided solution file.*
* Compilation error*: If the system fails to compile the solution code.*
* Time limit exceed: *If the code takes more time than it is expected to execute that code.*

##### User Profile Management

Every user will be provided a dedicated profile against the email he has provided. In her/his profile, a user can see the problems s/he has solved in the contests or after the contests. He can also see in which contest s/he has participated.

A user can also modify her or his profile information (institute, password and recovery pin).

### 3.2.4 ELICITATION WORK PRODUCTS

The work products produced as a sequence of requirements elicitation will vary depending on the size of the system or product to be built. Here, the Elicitation work product includes

* Making a statement of our requirements for the Programming Platform.
* Making a bounded statement of scope of our system
* Making a list of the stakeholders who participated in the requirement elicitation
* A description of the system’s technical environment.
* A list of requirements that are organized by function and domain constraints that apply to each other.
* A set of usage scenario that provide insight into the use of the system

# CHAPTER 4: SCENARIO BASED MODELING

This chapter describes the Scenario Based Model for the Programming Platform.

## 4.1 INTRODUCTION

Although the success of a computer-based system or product is measured in many ways, user satisfaction resides at the top of the list. If we understand how end users (and other actors) want to interact with a system, our software team will be better able to properly characterize requirements and build meaningful analysis and design models. Hence, requirements modeling begins with the creation of scenarios in the form of Use Cases, activity diagrams and swim lane diagrams.

## 4.2 DEFINITION OF USE CASE

A Use Case captures a contract that describes the system behavior under various conditions as the system responds to a request from one of its stakeholders. In essence, a Use Case tells a stylized story about how an end user interacts with the system under a specific set of circumstances. A Use Case diagram simply describes a story using corresponding actors who perform important roles in the story and makes the story understandable for the users.

The first step in writing a Use Case is to define that set of “actors” that will be involved in the story. Actors are the different people that use the system or product within the context of the function and behavior that is to be described. Actors represent the roles that people play as the system operators. Every user has one or more goals when using system.

**Primary Actor**

Primary actors interact directly to achieve required system function and derive the intended benefit from the system. They work directly and frequently with the software.

**Secondary Actor**

Secondary actors support the system so that primary actors can do their work. They either produce or consume information.

## 4.3 USE CASE DIAGRAMS

Use Case diagrams give the non-technical view of overall system.

### 4.3.1 Level-0 Use Case Diagram

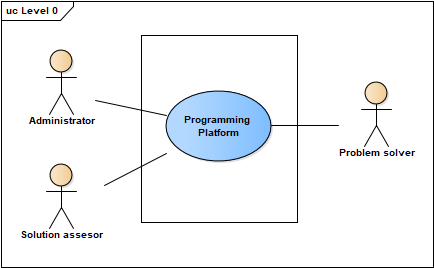


Figure: Level 0 Use Case Diagram- Programming Platform

### 4.3.2 Level-1: Use Case Diagram-Subsystem

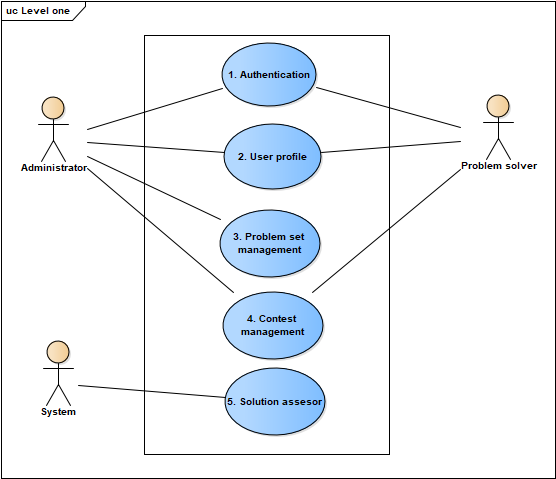


Figure: Level 1 Use Case diagram - Subsystems

**Description of Use Case Diagram Level 1-**

Primary Actor: Administrator, Problem Solver, System.

There are five subsystems in the programming platform. They are:

1. Authentication.

2. User Profile.

3. Problem Set Management.

4. Contest Management.

5. Solution Assessment.

### 4.3.3 Level-1.1: Use Case Diagram-Authentication

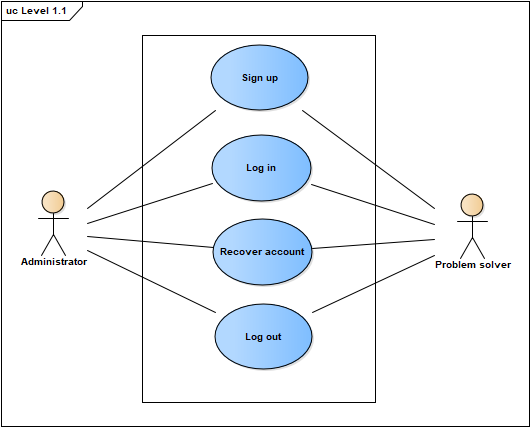


Figure: Level 1.1 Use Case Diagram- Authentication

**Primary Actor:** Administrator, Problem Solver.

The Authentication Subsystem can be divided into four parts:

1. Sign Up
2. Log in
3. Recover Account
4. Log Out

At the time of installation of this system an administrator account will be created who will maintain the system. For account creation any type of user has to provide these information

* Username/Email
* Password
* Recovery Pin
* Institute

User’s password can be at most fifteen characters and pin four characters. Only authenticated user can enter to the system.

A user can log into the system entering her/his username/email and password. If provided username/email and password matches, user can enter to the system. Otherwise an error message will be generated.

If a user forget her/his password then she/he can recover her/his account. For user account recovery the system will ask for her/his username/email and pin number. If provided email and pin number matches then she/he will be able to see her/his password. Then using that password she/he can enter into the system.

### 4.3.3 Level-1.2: Use Case Diagram-User Profile

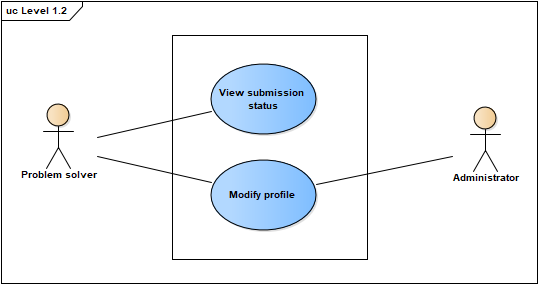


Figure: Level 1.2 Use Case Diagram- User Profile

**Primary Actor:** Problem Solver.

The User Profile Subsystem can be divided into four parts:

1. View submission status
2. Modify profile

Every user will be provided a dedicated profile against the email he has provided. In her/his profile, a user can see the problems s/he has solved in the contests or after the contests. He can also see in which contest s/he has participated.

A user can also modify her or his profile information (institute, password and recovery pin).

### 4.3.4 Level-1.3: Use Case Diagram- Problem Set Management

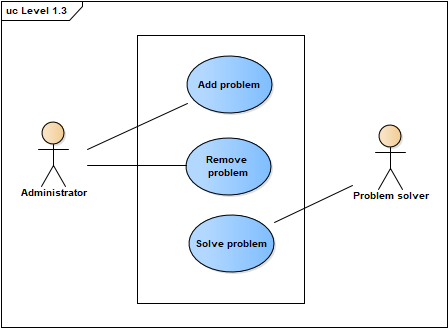


Figure: Level 1.4 Use Case Diagram- Contest Management

**Primary Actors:** Administrator, Problem Solver.

The Problem Set Management Subsystem can be divided into four parts:

1. Set Problem
2. Remove Problem
3. Solve Problem

To arrange a contest, the administrator needs to set problems. The administrator will set a problem by uploading a problem description file which will be a portable document format (pdf) file. The administrator will also upload a test input file, a solution file and time limit corresponding to every problem. The problem will also have a problem name.

The administrator can also remove any problem which has been set previously.

User can solve any problem. For that he needs to select the problem identified by contest id and problem id. Then he needs to provide the programming language name and the solution. The solution will be assessed by the subsystem “Solution assessor”.

### 4.3.5 Level-1.4: Use Case Diagram- Contest Management

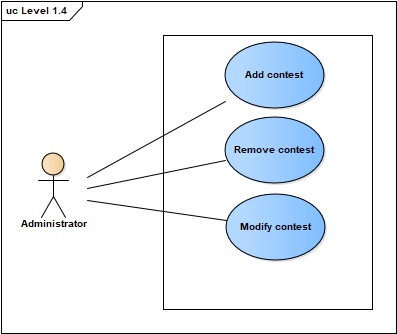


Figure: Level 1.4 Use Case Diagram- Contest Management

**Primary Actors:** Administrator, Problem Solver.

The Contest Management Subsystem can be divided into four parts:

1. Add Contest
2. Remove Contest
3. Modify Contest

To add a contest, an administrator user needs to give contest title, password, starting time, duration. An id will be generated from the system. Then administrator will set problems for the contest which has been described in “Problem Set Management”. The administrator user will also be able to modify the contest information before the contest. S/he can also remove the contest.

## 4.4 ACTIVITY DIAGRAMS OF PROGRAMMING PLATFORM