**Chapter-1**

**INTRODUCTION**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

The project is a client server based application which has a great scope for implementation in organizations that require automated systems to make the complex and time consuming task of conducting programming contests and practical examinations in the computer labs easier and less error prone by diminishing the scope of human error by installing such a system.

**1.1 MOTIVATION AND OVERVIEW**

The main purpose of this web based project is to conduct online coding examinations or the user can enhance their skills by attempting the coding challenges. This can be very useful for deployment in computer programming labs in various universities or schools.

There will be mainly two users: Teacher, who can conduct the test on the portal and Students, who will attempt them. On submitting, the code will be executed on the server and the result will be displayed to the student by displaying either the number of test cases passed or compiling error. At the end of exam the teacher will be able to see the submitted code of students and number of test cases passed by them.

**1.2 OBJECTIVE**

The objective of this project is to create an online coding platform which serves the purpose to organize coding competitions at a single platform and give user a platform to showcase or test there skills, along with that it is very useful during practical examinations in the labs where examiner does not need to visit every students’ individually to see his performance in the given test or program but it provides the ease to the examiner to check the status of all the students on his own system and at the same time it removes the drawback of ordinary system in which some students can simply copy the code from somewhere else as the editor in this portal has disabled that functionality so there is no chance that student can cheat by anyway.

**1.3 SUMMARY OF SIMILAR APPLICATIONS**

**Openjudge[1]** is a LAN based automated computer program grading solution designed for trainings on computer languages, data structures and algorithms. It also extends to software testing and profiling tasks emphasizing security and flexibility program judging implementation in Python. Openjudge bridges the gap between online judges and offline contests. It offers all the capabilities of an online judge for an offline system, with very few requirements. One can set up a programming competition with minimal effort spent in developing the infrastructure needed for it.

**DOMjudge[2]** is a system for running programming contests, like the ACM ICPC regional and world championship programming contests. This means that teams are on-site and have a specific time period (mostly 5 hours) and one computer to solve a number of problems (mostly 8-11). Problems are solved by writing a program in one of the allowed languages, that reads input according to the problem input specification and writes the correct, corresponding output. The judging is done by submitting the source code of the solution to the jury. There the jury system automatically compiles and runs the program and compares the program output with the expected output. This software can be used to handle the submission and judging during such contests. It also handles feedback to the teams and communication on problems (clarification requests). It has web interfaces for the jury, the teams (their submissions and clarification requests) and the public (scoreboard).

**1.4 ORGANIZATION OF THE PROJECT**

* We have divided our work in the form of chapters namely: Software requirement Analysis, Software Design, Implementation and user Interface, Software Testing and Conclusion.
* In **Chapter 2 Software Requirement Analysis** first one is feasibility in which we have discussed economic, technical, operational, time and resource feasibilities and second one is functional requirements, software requirements and hardware requirements of the projects and in third part we have described the Summary of modules inn which we have discussed the summary of various modules of the project.
* In **Chapter 3 Software design** we have discussed design for the software which includes DFD Diagram, ER-Diagram, USE Case Diagram for the project.
* In **Chapter 4 Implementation and User interface**, we have included all user interface, output screens and description of the project.
* In **Chapter 5 Software Testing**, we have discussed generated test cases in black box testing and white box testing.
* In **Chapter 6 Conclusion** we have discussed the summary and conclusion of the project.