ISSN:0975-9646

# Automatic Evaluation System for Student Code

**Pratik Saraf**
*Department Of IT*
*KJSCE,Vidyavihar(E)*
*Mumbai, India*

**Shankar Ramesh**
*Department Of IT*
*KJSCE,Vidyavihar(E)*
*Mumbai, India*

**Sachin Patel**
*Department Of IT*
*KJSCE,Vidyavihar(E)*
*Mumbai, India*

**Prof. Sujata Pathak**
*Department of IT*
*KJSCE,Vidyavihar(E)*
*Mumbai, India*

***Abstract*— Grading of student programs and coding assignments requires verifying correctness of a great many programs, all against identical inputs and outputs. It becomes a very tedious task for a professor as well as the student to get their programs evaluated. This situation is a candidate for automation, and calls for a system which would allow students to upload their programs on the server and get them evaluated based on test cases specified by the professor. Automated testing is very common in the industrial domain, and it seems logical to extend this practice to academia. Since the fundamentals of software testing and related activities are often elusive in undergraduate curricula, the proposed system would encourage test driven development and hence better programmers for the industry**

***Keywords*— Computer Science Education, Education, Program Testing Automation**

## I. Introduction

The task of evaluating a student program is a time consuming, but important, part of any computer programming course. Generally, it is not feasible to test each student's program thoroughly during lab sessions. Another problem is the student does not always perform the programs.

Interesting work has been reported on the development of criteria for grading student programs, but little seems to have been done in order to provide assistance with the submission and evaluation of assignments.

In this paper a web based system is described, that helps students to program more effectively and generate better programs and to support the professor for the evaluation and testing of student programs and coding assignment.[1]The evaluation of student program can be simplified by actually compiling and running the student program against the test cases provided by the professor, and comparing the output produced by the student program to that of the professor's output. The system automatically compiles and executes the program and stores the files on the server as well as the student's grade in a database.

## II. Background And Motivation

Previously programs were individually checked for correctness by the professor but it would be a great burden on the professor and also the student to get their programs evaluated. Also some students would never perform the program.

Transparency of marks is a very important factor for the students. Our system would keep the marking transparent by displaying the professor and the student the marks he has scored and also provides for resubmission of an assignment if a student is not satisfied with his performance in a particular submission.

It is very important for a student to compile, debug and execute programs. Defective and bug-riddled software has been one of the major problems of the software industry and has accounted for huge losses, as well as the failures of large projects.[2] Test Driven Development is an important aspect of the industry, if inculcated in a student at early stage it would be very helpful for the student as well as the industry.

This system would encourage the students to participate in various competitive programming contests, since they follow a similar pattern of evaluation. It would also make the transition of academia to industry much smoother, since a large part of the industry uses automated testing.

## III. Design Goals

Since most of our programming courses are in Java, we decided to build this system for Java, specifically. Moreover we needed it to withstand buggy code and a large number of submissions at a given time. Also we wanted the environment to accommodate the flexibility of different courses in our curriculum and fairly complex programs that can often be designed in any number of ways. And so we set out with these design goals in mind for the Student and Professor:

### *A. Student*

- To register himself without any hassles in the system.
- To login to the system.
- To select the subject and assignment for which he wants to submit the program for evaluation.
- Provide an interface in a web browser where the code can be written and submitted. For this purpose ace editor has been integrated into the system.
- To submit and download the program.
- To get feedback of final grade after submission and provide for resubmission if not satisfied.

### *B. Professor*

- To login to the system.
- To upload file for test cases and outputs.
- To obtain results of student in a database.

Before proceeding to implement, we vetted existing solutions. In the web-based space are ideone [3], codepad [4], CodeEval [5], and compilr [6], codechef [7] all of which support browserbased compilation and interpretation of code. Studying all these systems gave us an idea on how to proceed with the implementation and design.

## IV. Implementation

At a high level, Automatic Evaluation System is a web-based black box that allows for secure execution of student code. Its inputs include program code and standard inputs. And its outputs include grades attained. This system is not meant to be used by humans directly, as via a GUI. Rather, it accepts inputs via HTTP, and it returns outputs via HTTP.

### *A. Architecture*

As shown in Fig 1, the architecture consists of below-specified components:

1. Client Tier: At the client tier, we have the web browser that is capable of sending requests and receiving a response from the server.
2. Middle Tier: This tier is the web server tier. All Java classes and other business logic are located at this tier. The web server serves HTML and PHP pages to users and responds to user requests.
3. Database Tier: The third tier is made up of the database and the file server. Database used is MySQL. The file server holds the submitted files of the user, while the database stores information related to users and his grades in programs.

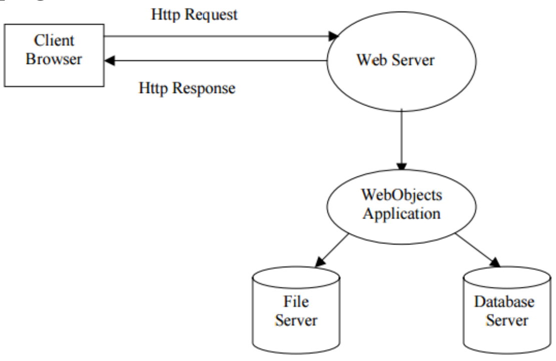


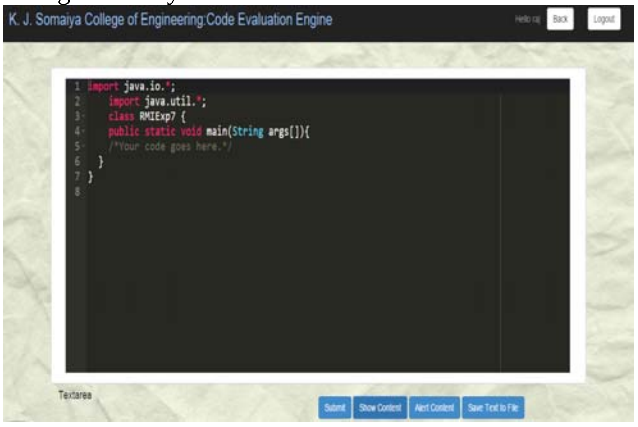
Fig 1. Architecture

### *B. Front End*

Automatic Evaluation System's front end is an HTML and PHP based API. Bootstrap framework has been used to improve the look and feel of the web pages.

The system has a facility for students to register themselves. Once registered, student can gain access to the system using his Roll Number as his unique login ID, and password. After logging in, drop down boxes are available, in which the student has to select the required subject and experiment number. Once this is done, student can proceed and submit his code in the system for evaluation. Ace Editor[8], which is an open-source editor, has been integrated into this system, which will be used by the student for submission of the programs, as shown in Fig 2. The code is evaluated automatically as and when it is submitted and the grade obtained is displayed as a feedback.

Professors too can log into the system through the same login page as that of the students', using their unique login ID and password. After logging in, the professor has two options of either uploading testcase files and output files, or viewing the database of students' grades. For uploading of files, certain fields like subject, experiment number etc. must be first selected in the respective drop-down boxes. For viewing the database of grades, the required class must be selected by the professor and database of that particular class will be available in an Excel file, in which further sorting and analysis can be done.



```
import java.io.*;
    import java.util.*;
    class RMIExp7 {
    public static void main(String args[]){
    /*Your code goes here.*/
  }
}
```

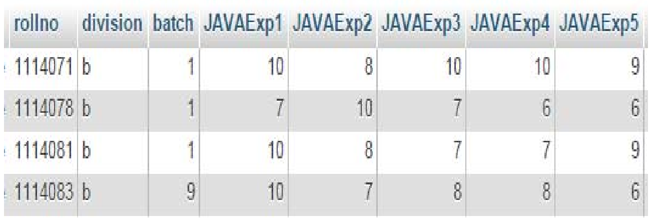
Fig 2. Interface to submit student's program

### *C. Back End*

The back end processing is implemented using PHP. Sessions are used to validate users in all the pages being accessed. Also, MySQL is the database technology used. Two databases are used; one for user information and the other for students' grades. Names of all the files uploaded by students and professors are automatically standardized by the system.

Whenever a student registers, a new table entry in the user database is done. Also, a folder of his user ID is created at a specified location on the file server. This folder will consist of all his files uploaded and generated by the system. The class name of the program will automatically be generated according to the subject and experiment number specified by him. After a student uploads his program, the test cases that have already been uploaded by the professor will be fed as inputs to his program automatically and an output file will be created. This output file will be then compared with the already-uploaded output file of the professor and the student will be graded according to the results of the comparison. This grade will be shown to the student as feedback, and also will be recorded in the grades database, as shown in Fig 3.

Files uploaded by the professor will be automatically renamed according to subject and experiment number that have been specified by him; and then stored in the folders specified for tesctases and outputs. There is a facility to view the grades of students, in which the professor needs to first specify his class of interest and the database of the respective class will be downloaded into an Excel file. This file can further be used for analysis.



| rollno | division | batch | JAVAExp1 | JAVAExp2 | JAVAExp3 | JAVAExp4 | JAVAExp5 |
|---|---|---|---|---|---|---|---|
| 1114071 | b | 1 | 10 | 8 | 10 | 10 | 9 |
| 1114078 | b | 1 | 7 | 10 | 7 | 6 | 6 |
| 1114081 | b | 1 | 10 | 8 | 7 | 7 | 9 |
| 1114083 | b | 9 | 10 | 7 | 8 | 8 | 6 |

Fig 3. Database of student grades

## V. Conclusion And Future Work

In this paper, a web-based solution for code evaluation is introduced and discussed. For this application, we have implemented an algorithm which is devised by us to correctly assess student code. This facilitates to evaluate the user automatically and grade them systematically.

There is significant potential to develop this system further, through any or all of the following additions to the system.

It is feasible to automatically send more intelligent feedback to the students, informing them of how they are doing throughout the year. These messages should make it possible to automatically give a higher level of individual feedback to students, even when the number of students has become quite large.

This system can be further extended to include more languages such as c++ and python. The system can be further developed, to provide a complete IDE like environment in a browser. [9]For better optimization of student programs, they can be run in a resource constrained environment in a sandbox.

## Acknowledgment

We are grateful to all the faculty members of the Information Technology department of our college, K. J. Somaiya College of Engineering for their support and motivation to this work. Special thanks to Prof. Sujata Pathak for her valuable guidance.

## References

[1] Kenneth M. Dawson-Howe. Automatic Submission and Administration of Programming Assignments, Department of Computer Science, Trinity College, Dublin Ireland. 1995.
[2] Anuj R. Shah. Web-CAT: A Web-based Center for Automated Testing. Master’s thesis, Virginia Polytechnic Institute and State University, 2003.
[3] Sphere Research Labs. ideone. http://www.ideone.com/.
[4] Steven Hazel. codepad. http://www.codepad.org/.
[5] CodeEval Inc. CodeEval. http://www.codeeval.com/.
[6] Compilr Inc. compilr. https://www.compilr.com/.
[7] Directi Inc. Codechef. http://www.codechef.com.
[8] The Ace Project. Ace Editor http://www.ace.c9.io/.
[9] CS50 Sandbox Secure Execution of Untrusted Code , David J. Malan School of Engineering and Applied Sciences Harvard University Cambridge, Massachusetts, USA. SIGCSE, 2013.