Online Java IDE

A Project Report for Summer Industrial Training

Submitted by

Abhishek Saha
Pramit Rana
Arnab Mallik
Akashdeep Ghosh
Dwaipayan Sen
Aakash Mandal

in partial fulfillment for the award of the degree of

B. Tech

in

Information Technology

At

RCC Institute of Information Technology

श्रमम् बिना न किमपि साध्यम्

from

Ardent Computech Pvt. Ltd

ARDENT
COMPUTECH PVT. LTD
Knowledge is power, power leads to growth, growth is life

Jun - Jul 2014

Ardent Computech Pvt. Ltd.

BONAFIDE CERTIFICATE

Certified that this project work was carried out under my supervision

"Online Java IDE"
is the bonafide work of

Name of the student	***Signature***
Abhishek Saha	_ _ _ _ _ _ _ _ _ _ _ _
Pramit Rana	_ _ _ _ _ _ _ _ _ _ _ _
Arnab Mallik	_ _ _ _ _ _ _ _ _ _ _ _
Akashdeep Ghosh	_ _ _ _ _ _ _ _ _ _ _ _
Dwaipayan Sen	_ _ _ _ _ _ _ _ _ _ _ _
Aakash Mandal	_ _ _ _ _ _ _ _ _ _ _ _

SIGNATURE

Name :
PROJECT MENTOR

SIGNATURE

Name:
EXAMINERS

Ardent Original Seal

Acknowledgement

I take this opportunity to express my deep gratitude and sincerest thank to my project mentor, *Dhruba Ray* for giving most valuable suggestion, helpful guidance and encouragement in the execution of this project work.

I will like to give a special mention to my colleagues. Last but not the least I am grateful to all the faculty members of Ardent Computech Pvt. Ltd. for their support.

1. Title of the Project

Online Java IDE

2. Introduction and Objectives of the Project

Need of Project

The definition of our problem lies in the difference between the existing compilers which reside on a user's system and an ONLINE COMPILER which caters to the needs of a user in different programming languages.
An Online Compiler gives the programmer the option to keep his/her own system light by transferring the load of compilation and running the code to the remote server. This remote server will hold the compilers for different programming languages. An online system will also help the user to keep track of his previous interactions with the system in shape of the various source codes that the user has run on the system. Thus, the user can browse through his entire 'HISTORY' of files. In addition, he can upload source files directly, and can run them directly later.
The end user's task is much simplified, as he doesn't have to worry about keeping his compilers updated. This task is handled by the developers, and the whole compilation and running load is taken by the servers. The system provides the user with an interface for providing code and input, and removes all the other hassles in between. It is a very handy tool for testing code before implementing.

The Programming Process:

These standard steps are to be followed while creating a J2EE Project:
1. Deciding what your application is doing by creating an overall design.
2. Creating the Visual elements (frontend) of the application (the interfaces and menus that the users will interact with). This is done by creating JSP pages and applying required CSS elements.

3. Project Category

RDBMS/OOPS/Networking/Web Application

4. Tools/Platform, Hardware and Software Requirement specifications.

Tools

1. Eclipse JUNO
2. Dia

Platform

1. Microsoft Windows 7/8

Hardware Requirement Specification

Client Machine		Server Machine	
Disk Space	200 MB	**Disk Space**	10 GB (recommended)
Processor	2nd-generation Core i5 (2GHz+), 3rd/4th-generation Core i5 processor, or equivalent	**Processor**	2nd-generation Core i5 (2GHz+), 3rd/4th-generation Core i5 processor, or equivalent
Memory	512 MB	**Memory**	2 GB (recommended)

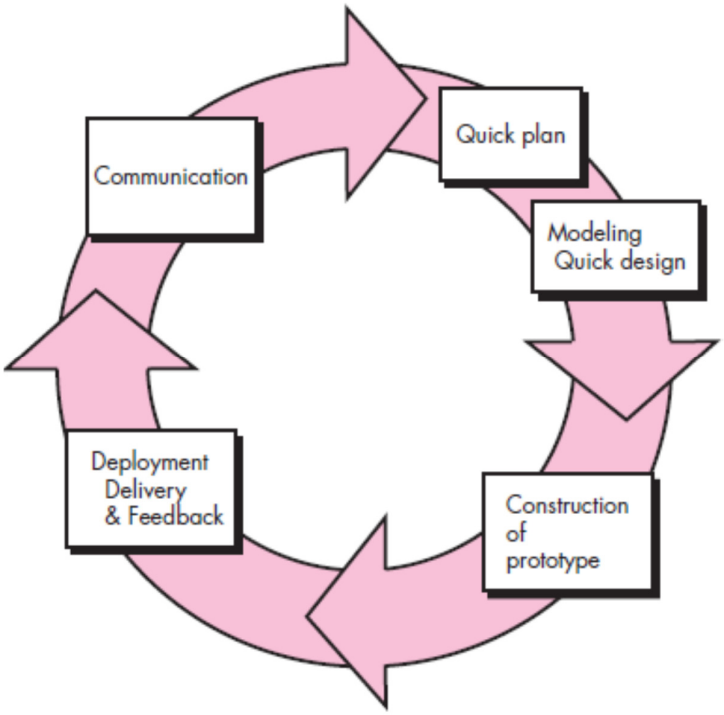
Software Requirement Specification

Client Machine		Server Machine	
Browser	Any standard browser with Javascript interpreter	**Software**	Java 7, Apache Tomcat 7.0
Client side mark up / scripting languages	HTML, Javascript	**Database Management System Software**	Oracle 11g
		Specification	Servlet 3.1 and JSP 2.3

5. Goals of Implementation

The implementation aims at seamless document sharing across the institution.

6. SDLC Process Applied



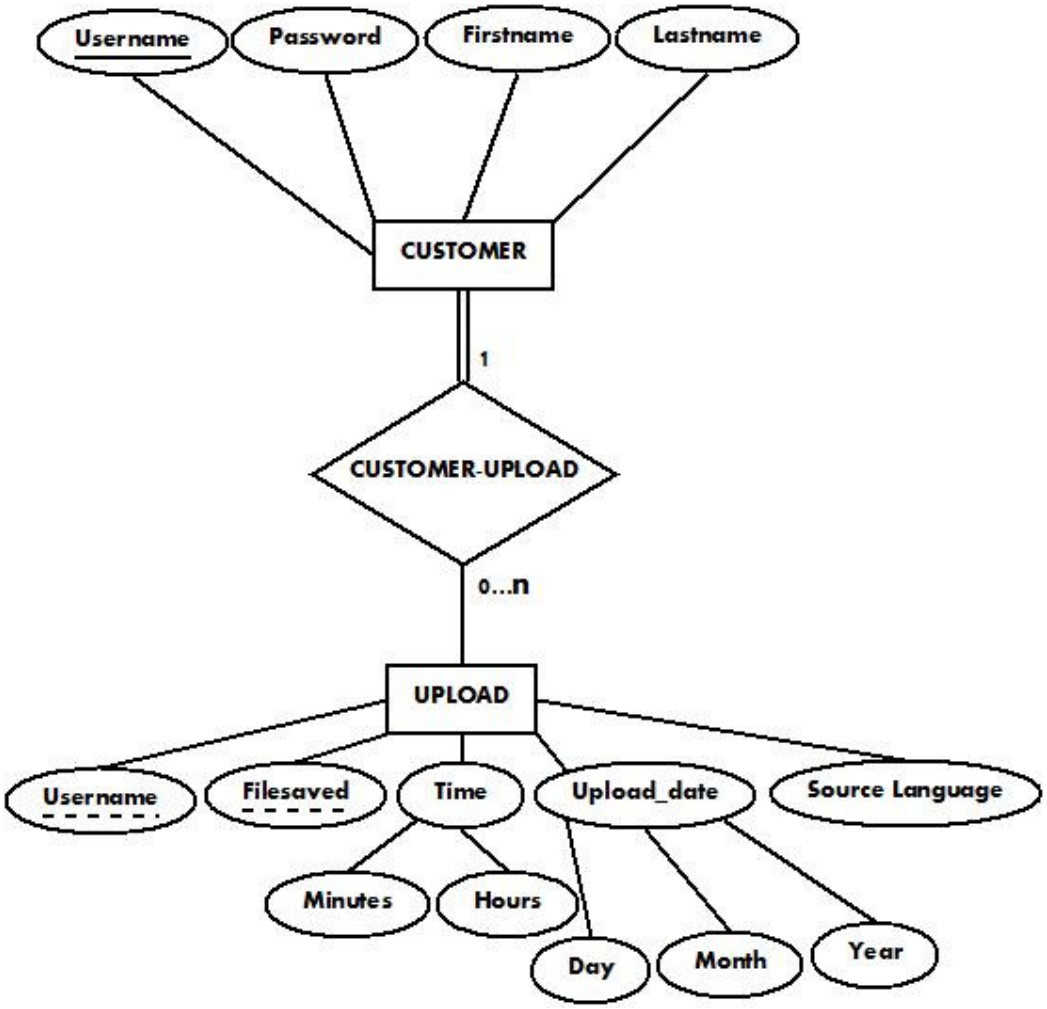
Often, a customer defines a set of general objectives for software but does not identify detailed input, processing, or output requirements. In other cases, the developer may be unsure of the efficiency of an algorithm, the adaptability of an operating system, or the form that human/machine interaction should take. In these, and many other situations, a prototyping paradigm may offer the best approach.

The prototyping paradigm begins with **requirements gathering**. Developer and customer meet and define the overall objectives for the software, identify whatever requirements are known, and outline areas where further definition is mandatory. A **"quick design"** then occurs. The quick design focuses on a representation of those aspects of the software that will be visible to the customer/user (e.g., input approaches and output formats). The quick design leads to the construction of a prototype. The prototype is evaluated by the customer/user and used to refine requirements for the software to be developed. Iteration occurs as the prototype is tuned to satisfy the needs of the customer, while at the same time enabling the developer to better understand what needs to be done.

Ideally, the prototype serves as a mechanism for identifying software requirements. If a working prototype is built, the developer attempts to use existing program fragments or applies tools (e.g., report generators, window managers) that enable working programs to be generated quickly.

7. Data Model

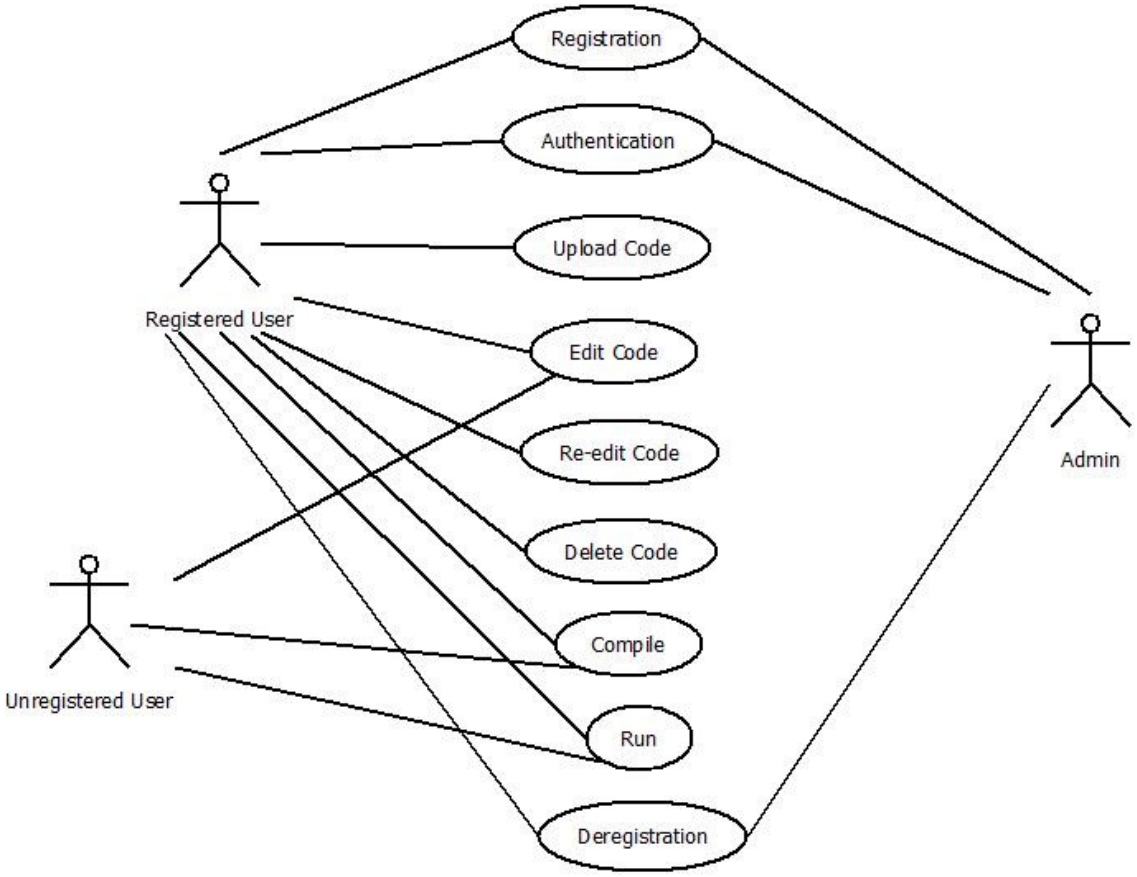
ER Diagram



8. Functional Requirements

Functional Requirements are those that refer to the functionality of the system, i.e., what services it will provide to the user. Nonfunctional (supplementary) requirements pertain to other information needed to produce the correct system and are detailed separately.

Use Case Diagram



9. Non Functional Requirements

In addition to the obvious features and functions that you will provide in your system, there are other requirements that don't actually DO anything, but are important characteristics nevertheless. These are called "non-functional requirements" or sometimes "Quality Attributes." For example, attributes such as performance, security, usability, compatibility. aren't a "feature" of the system, but are a required characteristic. You can't write a specific line of code to implement them; rather they are "emergent" properties that arise from the entire solution. The specification needs to describe any such attributes the customer requires. You must decide the kind of requirements that apply to your project and include those that are appropriate.

Each requirement is simply stated in English. Each requirement must be objective and quantifiable; there must be some measurable way to assess whether the requirement has been met.

Often deciding on quality attributes requires making tradeoffs, e.g., between performance and maintainability. In the APPENDIX you must include an engineering analysis of any significant decisions regarding tradeoffs between competing attributes.
Here are some examples of non-functional requirements:

Performance Requirements: As the server memory is not more than 2 GB, so concurrent access may decrease the performance level. For better performance The file size is restricted to 5 MB.

Operating Constraints: There are some operating constraints of the software. Operation performance is not uniform. If 4 or 5 users accesses the system at the same point of time, the server may slow down and there may be unexpected delay in the processing.

Accuracy & Precision: The software ensures accuracy but not 100%.It is incapable of detecting any runtime error and also it cannot take any user input.
On the other hand, Precision is guaranteed for the user.

Usability: A quick glance at the use case diagram, ER diagram and the guidelines will be helpful enough for a user to access the online IDE. The entire system is designed to ensure user-friendliness.

Security: Each and every user account is secured. Individual log in is dealt with a session. So one cannot simple access an independent page. The persistent database stores the user details and upload details in a secured way.

10. Feasibility Study

You should provide a feasibility report in the following format:

- **Product:** This product will help any user with an email-id to be registered on the system, who will be provided with facilities like upload, edit, compile, run his/her

own code. It will even provide some facilities for an unregistered user, such as edit, compile and run his/her code; but obviously he won't be able to use the database.

- **Technical Feasibility:** As far as the requirements specification is concerned, the proposed system development can be carried out within the scope of the technical infrastructure of the organization.

- **Alternative Solution:** We could have developed a Desktop application instead of this online version. But then the application couldn't be an online one. Each user would have to install it on their computers and wouldn't have enjoyed the lightweight and convenient way of running their codes online without the hassle of setting up the compilers etc on their desktops.

  Here, the complete overhead is borne by the server itself and the clients enjoy a hazardless platform to perform their tasks on.

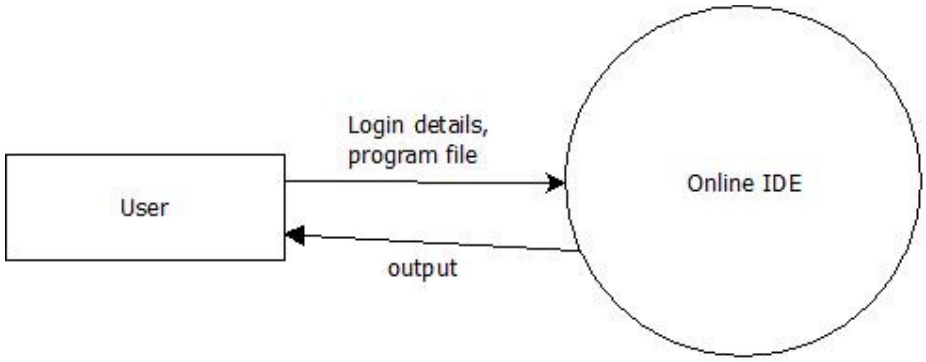
  At this point, all of the planning for the project has been done and if the feasibility study has shown that the project is likely to succeed within its constraints, then it only remains for us to start the requirements analysis and thus proceed with the project.

Feasibility Study	
System: Online IDE	Date: 21/06/2014
Author: AAPDAA	Page: 1
Product	
The project requires a web application to be developed that will allow online knowledge/document/paper sharing.	
Technical Feasibility	
The web application will be developed using JEE and Oracle. The team is competent in that.	
Alternate Solution	
Could be a desktop system but that would not allow documents to be shared online.	

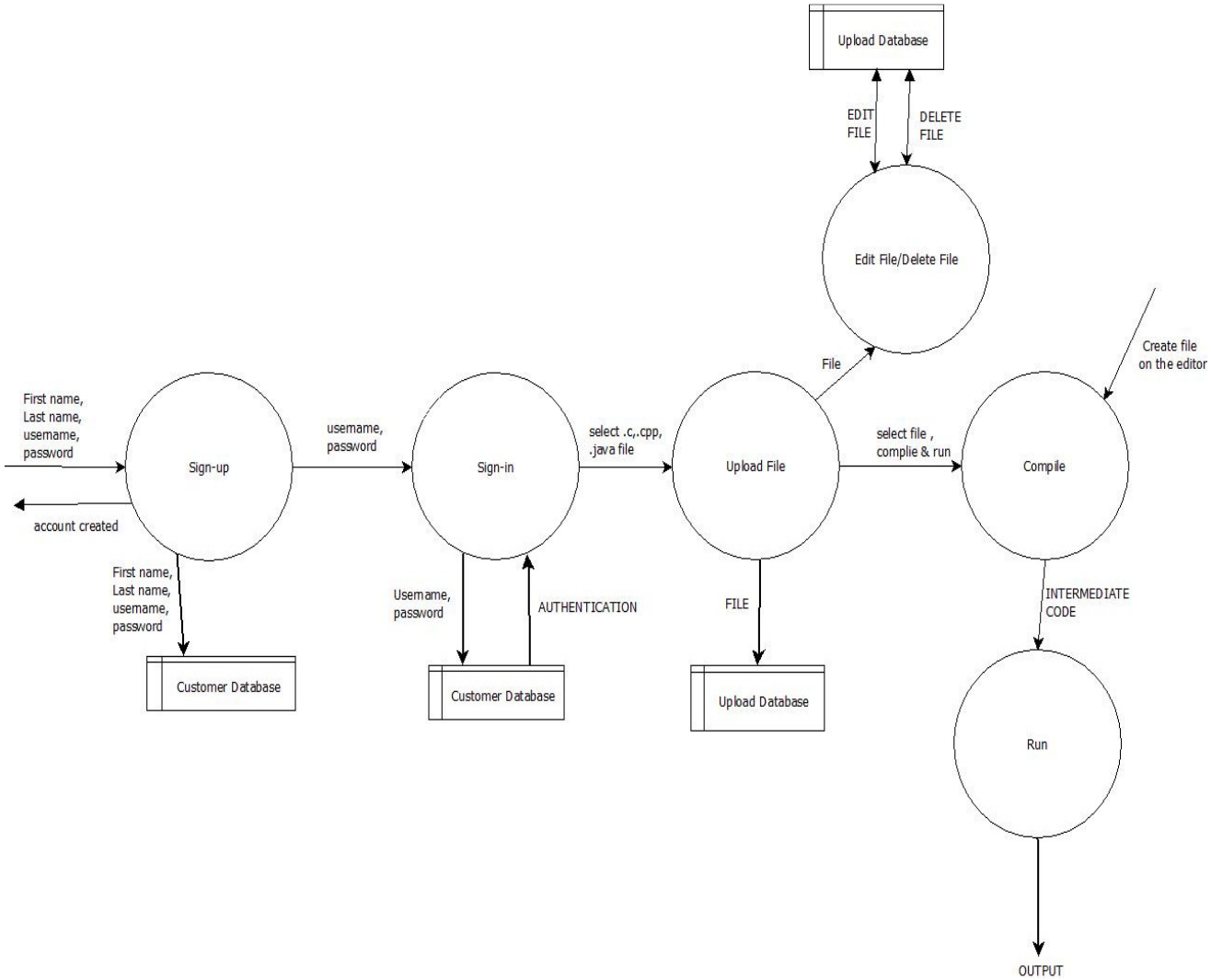
11. Software Engineering Paradigm Applied

Data Flow Diagrams

Level 0

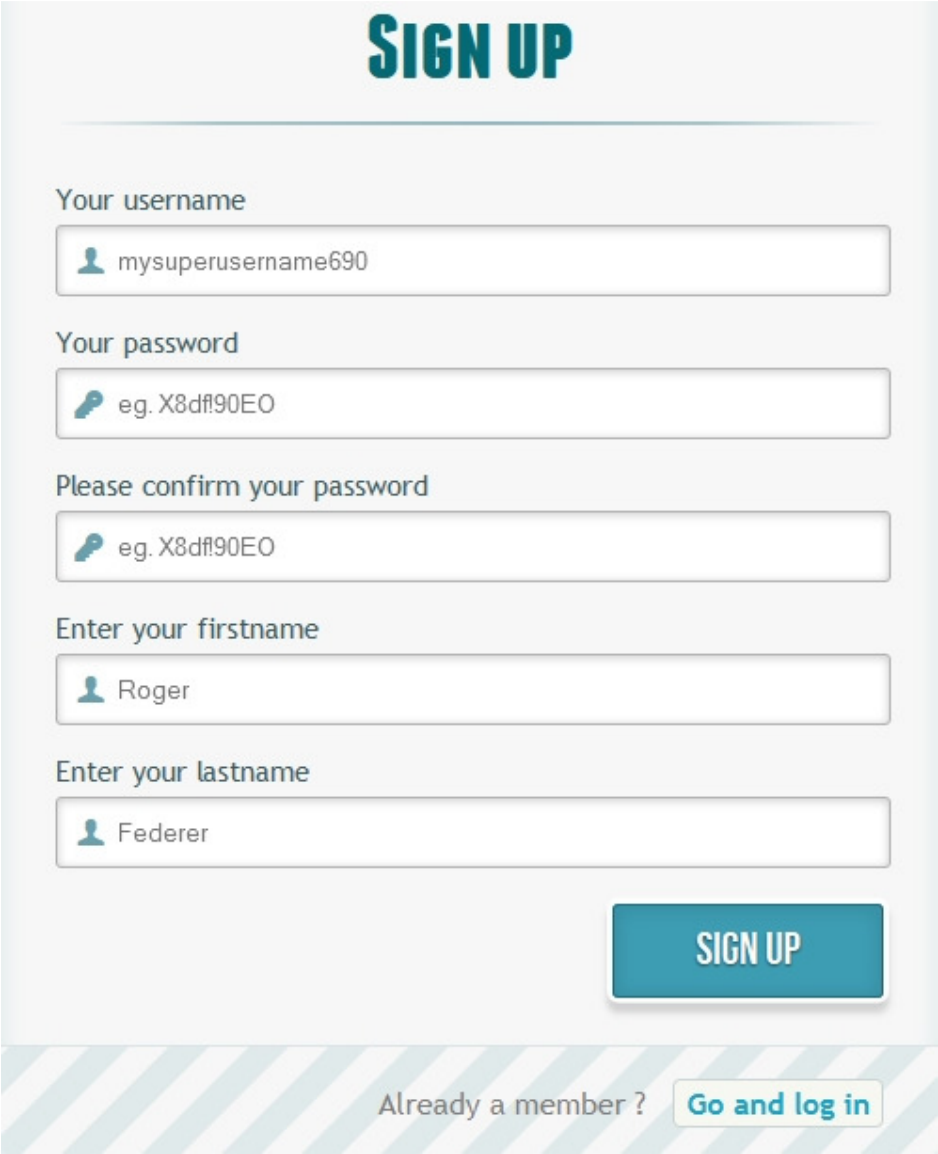


Level 1

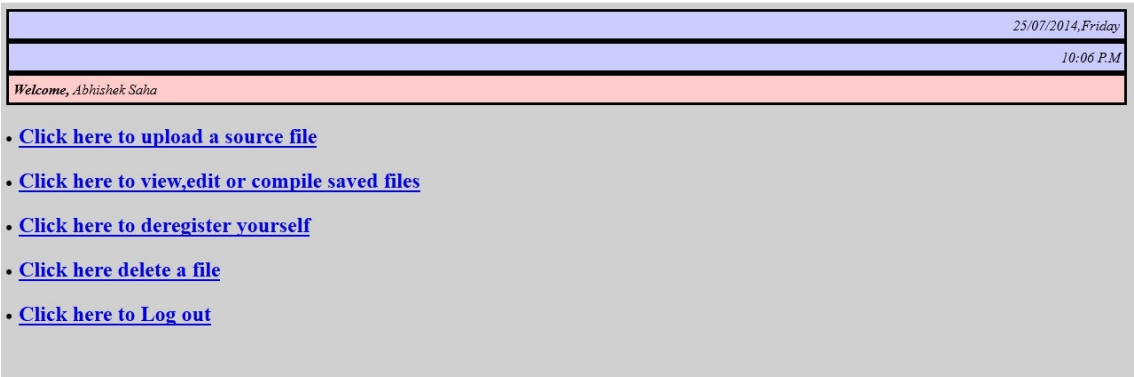


12. User Interface Design

Registration Page

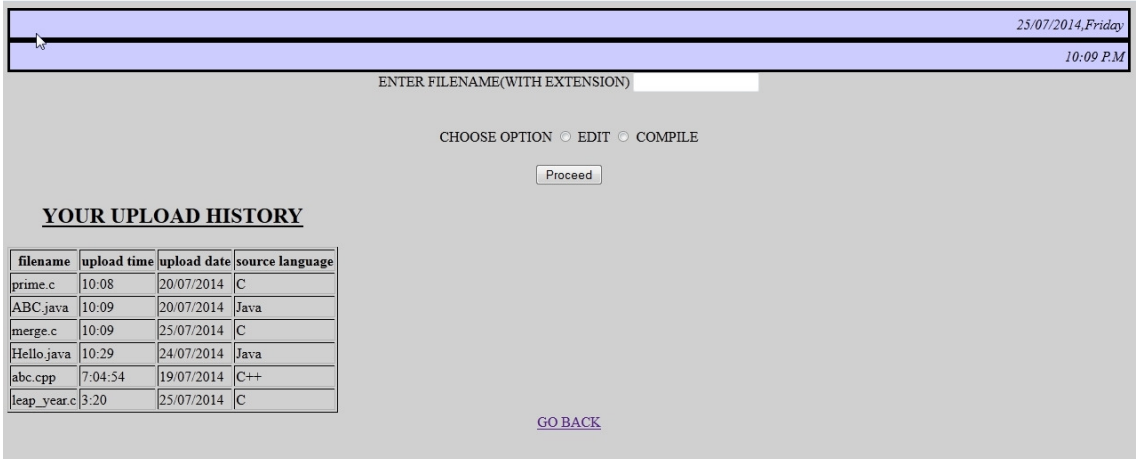


Options for Registered User



Upload a file

Upload history along with the options to edit or compile



filename	upload time	upload date	source language
prime.c	10:08	20/07/2014	C
ABC.java	10:09	20/07/2014	Java
merge.c	10:09	25/07/2014	C
Hello.java	10:29	24/07/2014	Java
abc.cpp	7:04:54	19/07/2014	C++
leap_year.c	3:20	25/07/2014	C

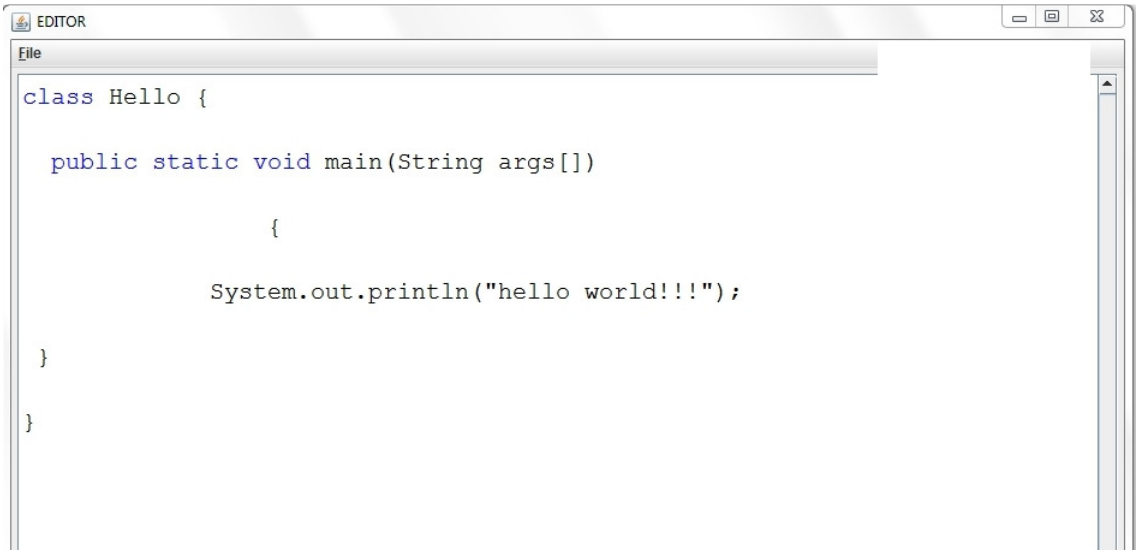
Compilation result

compilation successful click here to run

Run output

Output: hello world!!! Go back to home page

Editor look and feel



13. Coding

Screenshot for deleteFile.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1" import="dao.*"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Insert title here</title>
 <style>
    div { border: 3px solid; padding: 5px; }
    div.left { text-align: left; background: #ffcccc; }
    div.right { text-align: right; background: #ccccff; }

  </style>
</head>
<body bgcolor=D0D0D0>
<div class="right"><em><%=Currenttime.getdate() %>,<%=Currenttime.getday() %></em></div>
<div class="right"><em><%=Currenttime.gettime() %> <%=Currenttime.getAMPM() %></em></div>
<form method="post" action="deleteFile2.jsp">
Select source language
<input type="radio" name="lang" value="C">C
<input type="radio" name="lang" value="Java">Java
<input type="radio" name="lang" value="C++">C++
<input type="submit" value="proceed">
</form>
</body>
</html>
```

Screenshot for UploadServlet.java

```java
        }
        else
        {
            File f=new File(savePath+"\\"+un+"\\C++files");
            if(!f.exists())
                f.mkdirs();
           final_path=savePath+"\\"+un+"\\C++files";
           lang_type="C++";
        }
        part.write(final_path+File.separator + fileName);
         fn=final_path+File.separator + fileName;
        fn1=fileName;
    }
   User u=(User)request.getSession().getAttribute("user");
   DAO d=new DAO();
   d.openConnection();
   d.file_add(u.getUsername(),fn1,Currenttime.gettime(),Currenttime.getdate(),lang_type);
   try {
      d.closeConnection();
    } catch (SQLException e) {
      e.printStackTrace();
    }
   request.getSession().setAttribute("path",fn);
   out.println("Upload has been done successfully!");
   out.println("<a href='success.jsp'>Click here to go to home page</a>");
   out.close();
}
```

Screenshot for CreconpileServlet.java

```
deleteFile.jsp   editcompile.jsp   choice.jsp   login.html   DBupload.jsp   editor.java   *Crecompile...   »14

31
32      /**
33       * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
34       */
35      protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
36          String filename=(String)request.getSession().getAttribute("filename");
37          System.out.println(filename);
38          String un=((User)request.getSession().getAttribute("user")).getUsername();
39          String file=Path.savepath+"\\"+un+"\\Cfiles\\"+filename;
40          response.setContentType("text/html");
41          PrintWriter out=response.getWriter();
42          try {
43              Process p=Runtime.getRuntime().exec("gcc "+file);
44              p.waitFor();
45              BufferedReader input=new  BufferedReader(new InputStreamReader(p.getErrorStream()));
46              String line;
47              if((line=input.readLine())==null)
48              {
49                  out.println("compilation successful");
50                  out.println("<a href='CrunServlet'>click here to run</a>");
51              }
52              else {
53                  out.println("<font color='red'>Compilation Unsuccessful</font>");
54                  while((line=input.readLine())!=null) {
55                      out.println(line);
56                  }
57                  input.close();
58              out.println("<br><br><a href='CdebugServlet'>click here to debug</a>");
59
60              out.close();
61
62              }
63
```

Screenshot for JavaRunServlet.java

```
UploadServl...   ForwardServl...   CrunServlet....   JavaRunServl...   message.jsp   »15

33       */
34      protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletExcept
35
36          String class_name=(String)request.getSession().getAttribute("classname");
37
38          String un=((User)request.getSession().getAttribute("user")).getUsername();
39
40          String path=Path.savepath+"\\"+un+"\\Javafiles";
41
42          response.setContentType("text/html");
43
44          PrintWriter out=response.getWriter();
45
46
47          System.out.println("java -classpath "+path+" "+class_name);
48
49              Process p=Runtime.getRuntime().exec("java -classpath "+path+" "+class_name);
50
51              BufferedReader br=new  BufferedReader(new InputStreamReader(p.getInputStream()));
52
53              String read;
54
55              out.println("Output:");
56
57              while((read=br.readLine())!=null)
58
59              {
60                  out.println(read);
61              }
62
63              br.close();
64
65           out.println("<a href='success.jsp'>Go back to home page</a>");
```

14. Testing

Team Interaction

The following describes the level of team interaction necessary to have a successful product.

- The Test Team will work closely with the Development Team to achieve a high quality design and user interface specifications based on customer requirements. The Test Team is responsible for visualizing test cases and raising quality issues and concerns during meetings to address issues early enough in the development cycle.

- The Test Team will work closely with Development Team to determine whether or not the application meets standards for completeness. If an area is not acceptable for testing, the code complete date will be pushed out, giving the developers additional time to stabilize the area.

- Since the application interacts with a back-end system component, the Test Team will need to include a plan for integration testing. Integration testing must be executed successfully prior to system testing.

Test Objective

The objective our test plan is to find and report as many bugs as possible to improve the integrity of our program. Although exhaustive testing is not possible, we will exercise a broad range of tests to achieve our goal. We will be testing a Binary Search Tree Application utilizing a pre-order traversal format. There will be eight key functions used to manage our application: load, store, clear, search, insert, delete, list in ascending order, and list in descending order. Our user interface to utilize these functions is designed to be user-friendly and provide easy manipulation of the tree. The application will only be used as a demonstration tool, but we would like to ensure that it could be run from a variety of platforms with little impact on performance or usability.

Test Cases

Tested By:	Student1 name
Test Type	Unit Testing
Test Case Number	1
Test Case Name	User Identification
Test Case Description	The user should enter his/ her accurate userid and password so that he/she can able to go for the further options. The test case will check the application for the same since a user can only login with the correct userid , password.
Item(s) to be tested	
1	Verification of the userid and password with the record in the database.
Specifications	

Input	**Expected Output/Result**
1) Correct User id and password	1) Successful login
2) Incorrect Id or Password	2) Failure Message

Tested By:	Student2 name
Test Type	Unit Testing
Test Case Number	2
Test Case Name	Register
Test Case Description	The user fills up the registration form.

Item(s) to be tested	
1	Check whether the user has filled up the form properly.
2	Check whether the user has selected a valid file to upload

Specifications	
Input	**Expected Output/Result**
1) Trying to submit without filling the form properly. 2) Selecting an invalid file to upload.	1) The user is redirected to the registration page with error messages. 2) The user is informed about the error & is instructed to upload a valid file.

Unit Testing

Unit Testing is done at the source or code level for language-specific programming errors such as bad syntax, logic errors, or to test particular functions or code modules. The unit test cases shall be designed to test the validity of the programs correctness.

White Box Testing

In white box testing, the UI is bypassed. Inputs and outputs are tested directly at the code level and the results are compared against specifications. This form of testing ignores the function of the program under test and will focus only on its code and the structure of that

code. Test case designers shall generate cases that not only cause each condition to take on all possible values at least once, but that cause each such condition to be executed at least once. To ensure this happens, we will be applying Branch Testing. Because the functionality of the program is relatively simple, this method will be feasible to apply.

Each function of the binary tree repository is executed independently; therefore, a program flow for each function has been derived from the code.

Black Box Testing

Black box testing typically involves running through every possible input to verify that it results in the right outputs using the software as an end-user would. We have decided to perform Equivalence Partitioning and Boundary Value Analysis testing on our application.

System Testing

The goals of system testing are to detect faults that can only be exposed by testing the entire integrated system or some major part of it. Generally, system testing is mainly concerned with areas such as performance, security, validation, load/stress, and configuration sensitivity. But in our case well focus only on function validation and performance. And in both cases we will use the black-box method of testing.

`15. System Security measures (Implementation of security for the project developed)

- Only authorized users are allowed full access and unauthorized ones get a partial feel.
- Without signing in users are not allowed to go an intermediate page by typing an URL. For all such efforts, users will be redirected to the home page.

16. Database/Data security

- Database is present in remote machine.
- Oracle's default securities are applied.

17. Creation of User profiles and access rights

- The users are created by the RegisterServlet.
- The users are authenticated via database.

18. Future scope and further enhancement of the Project

The future scope of this application is that the application can be more user-friendly/interactive and more responsive.

19. Bibliography

1. Roger S. Pressman. Software Engineering: A Practioner's Approach (Sixth Edition, International Edition). McGraw-Hill, 2005.
2. George Reese, Database Programming with JDBC and Java, O'Reilly, 1997.
3. Software Development Environments on the Web: A Research Agenda Lennart C. L. Kats, Richard Vogelij, Karl Trygve Kalleberg, Eelco Visser
4. Real-Time Collaborative Coding in a Web IDE Max Goldman, Greg Little, and Robert C. Miller.MIT CSAIL.

5. Web Based Integrated Development Environment Mala Dutta,Kamal K Sethi, Ajay Khatri, International Journal of Innovative Technology and Exploring Engineering(IJITEE) ISSN:2278-3075,Volume-3,Issue-10, March 2014.
6. Database System Concepts Sixth Edition Avil Silberschatz, Henry F. Korth, S. Sudarshan.
7. Java: The Complete Reference, Seventh Edition Herbert Schildt.