# 1. INTRODUCTION

A web application is a program that runs on a computer with a web server, while its users interact with it via a web browser or similar user agent. Saral Programming Interface is a Web application which provides facilities to students and faculty for software development such as code completing and fixing, source code editing and management, automated testing, hosting coding contest over the internet. Cloud computing, according to the definition, is usage of computer resources (both hardware and software) which is served over the internet. Providing resources over Internet to multiple users at a time for better use of the resources and making the resources available to the users all the time over the Internet. The aim of this project is providing an easy-to-use web service providing many powerful feature of Saral Programming Interface by combining with the power of Cloud Computing for application developers.

## 1.1. Purpose

The purpose of this project is to build a User-Friendly interface to run their code in multiple programming and scripting languages, so learning of programming languages is easy and fun. The application is platform independent and a standalone application that can be run on any web browser with JavaScript enabled. This Application is made for both students and faculty for easy learning and instruction for an effective way of programming with better use of resources and time.

## 1.2. Scope

This project is intended for making use of today's popular technology Cloud Computing for Saral Programming Interface. Currently, there are lots of IDE's, both open-source and commercial, in the market. Usually they provide lots of extensive features to developers to ease application developer's life. However, there are two simple but substantial problems with today's IDE's. First is they require intensive CPU and memory usage which is not available all the time and since these applications are installed on specific system, it prevents portability. By combining Cloud Computing technology, this project will remove the requirement for powerful systems and provide portability to developer.

# 2. RELATED WORK AND LITERATURE SURVEY

## 2.1 Compilers and Interpreters

A compiler is a computer program (or a set of programs) that transforms source code written in a programming language (the source language) into another computer language (the target language), with the latter often having a binary form known as object code. The most common reason for converting source code is to create an executable program.

The name "compiler" is primarily used for programs that translate source code from a high-level programming language to a lower level language (e.g., assembly language or machine code). If the compiled program can run on a computer whose CPU or operating system is different from the one on which the compiler runs, the compiler is known as a cross-compiler. More generally, compilers are a specific type of translator.

While all programs that take a set of programming specifications and translate them, i.e. create a means to execute those specifications, are technically "compilers", the term generally means a program that produces a separate executable from the compiler (that may require a run time library or subsystem to operate), a compiler that merely executes the original specifications is usually referred to as an "interpreter", although because of differing methods of analysing what represents compilation and what represents interpretation, there is some overlap between the two terms.

A program that translates from a low-level language to a higher level one is a decompiler. A program that translates between high-level languages is usually called a source-to-source compiler or transpiler. A language rewriter is usually a program that translates the form of expressions without a change of language. The term compiler-compiler is sometimes used to refer to a parser generator, a tool often used to help create the lexer and parser. A compiler is likely to perform many or all of the following operations: lexical analysis, preprocessing, parsing, semantic analysis (syntax-directed translation), code generation, and code optimization. Program faults caused by incorrect compiler behavior can be very difficult to track down and work around; therefore, compiler implementors invest significant effort to ensure compiler correctness. Where an interpreter is a computer program that directly executes, i.e. performs, instructions written in a programming or scripting language, without previously compiling them into a machine language program. An interpreter generally uses one of the following strategies for program execution:

I. Parse the source code and perform its behavior directly.
II. Translate source code into some efficient intermediate representation and immediately execute this.
III. Explicitly execute stored precompiled code made by a compiler which is part of the interpreter system.

Early versions of Lisp programming language and Dartmouth BASIC would be examples of the first type. Perl, Python, MATLAB, and Ruby are examples of the second, while UCSD Pascal is an example of the third type. Source programs are compiled ahead of time and stored as machine independent code, which is then linked at run-time and executed by an interpreter and/or compiler (for JIT systems). Some systems, such as Smalltalk and contemporary versions of BASIC and Java may also combine two and three. Interpreters of various types have also been constructed for many languages traditionally associated with compilation, such as Algol, Fortran, Cobol and C/C++. While interpretation and compilation are the two-main means by which programming languages are implemented, they are not mutually exclusive, as most interpreting systems also perform some translation work, just like compilers. The terms "interpreted language" or "compiled language" signify that the canonical implementation of that language is an interpreter or a compiler, respectively. A high-level language is ideally an abstraction independent of particular implementations.

### *2.1.1 GCC Compilers*

The GNU Compiler Collection (GCC) is a compiler system produced by the GNU Project supporting various programming languages. GCC is a key component of the GNU toolchain and the standard compiler for most Unix-like Operating Systems. The Free Software Foundation (FSF) distributes GCC under the GNU General Public License (GNU GPL). GCC has plaGCC's external interface follows Unix conventions. Users invoke a language-specific driver program (gcc for C, g++ for C++, etc.), which interprets command arguments, calls the actual compiler, runs the assembler on the output, and then optionally runs the linker to produce a complete executable binary.

Each of the language compilers is a separate program that reads source code and outputs machine code. All have a common internal structure. A per-language front end parses the source code in that language and produces an abstract syntax tree ("tree" for short).

These are, if necessary, converted to the middle end's input representation, called GENERIC form; the middle end then gradually transforms the program towards its final form. Compiler optimizations and static code analysis techniques (such as FORTIFY_SOURCE, a compiler directive that attempts to discover some buffer overflows) are applied to the code. These work on multiple representations, mostly the architecture-independent GIMPLE representation and the architecture-dependent RTL representation. Finally, machine code is produced using architecture-specific pattern matching originally based on an algorithm of Jack Davidson and Chris Fraser.

GCC was written primarily in C except for parts of the Ada front end. The distribution includes the standard libraries for Ada, C++, and Java whose code is mostly written in those languages [2]. On some platforms, the distribution also includes a low-level runtime library, libgcc, written in a combination of machine-independent C and processor-specific machine code, designed primarily to handle arithmetic operations that the target processor cannot perform directly. Shed an important role in the growth of free software, as both a tool and an example.

### *2.1.2 Java Compilers*

A Java compiler is a compiler for the programming language Java. The most common form of output from a Java compiler is Java class files containing platform-neutral Java bytecode, but there are also compilers that emit optimized native machine code for a particular hardware/operating system combination. Most Java-to-bytecode compilers [1], Jikes being a well-known exception, do virtually no optimization, leaving this until run time to be done by the JRE. The Java virtual machine (JVM) loads the class files and either interprets the bytecode or just-in-time compiles it to machine code and then possibly optimizes it using dynamic compilation. A standard on how to interact with Java compilers programmatically was specified in JSR 199 [5].

***Major Java compilers***

As of 2017, the following are major Java compilers:

I. The Java Programming Language Compiler (javac), included in the Java Development Kit from Oracle Corporation, open-sourced since 13 November 2006 [7].

II. GNU Compiler for Java (GCJ), a part of the GNU Compiler Collection, which compiles C, C++, Fortran, Pascal and other programming languages besides Java. It can also generate native code using the back-end of GCC [7].

### *2.1.3 Python Interpreters*

Picking an Interpreter for Python.

When choosing a Python interpreter, one looming question is always present: "Should I choose Python 2 or Python 3"? The answer is a bit subtler than one might think. The basic gist of the state of things is as follows:

1. Most production applications today use Python 2.7.
2. Python 3 is ready for the production deployment of applications today.
3. Python 2.7 will only receive necessary security updates until 2020[16].
4. The brand name "Python" encapsulates both Python 3 and Python 2.

Recommendations

• Use Python 3 for new Python applications.

• If you're learning Python for the first time, familiarizing yourself with Python 2.7 will be very useful, but not more useful than learning Python 3.

• Learn both. They are both "Python".

• Software that is already built often depends on Python 2.7.

• If you are writing a new open source Python library, it's best to write it for both Python 2 and 3 simultaneously.

Only supporting Python 3 for a new library you want to be widely adopted is a political statement and will alienate many of your users. This is not a problem — slowly, over the next three years, this will become less the case.

If we're choosing a Python interpreter to use, I recommend you use the newest Python 3.x, since every version brings new and improved standard library modules, security and bug fixes. Given such, only use Python 2 if you have a strong reason to, such as a pre-existing code-base, a Python 2 exclusive library, simplicity/familiarity, or, of course, you absolutely love and are inspired by Python 2. No harm in that. Check out Can I Use Python 3? To see if any software you're depending on will block your adoption of Python 3. It is possible to write code that works on Python 2.6, 2.7, and Python 3. This ranges from trivial to hard depending upon the kind of software you are writing; if you're a beginner there are far more important things to worry about.

***Implementations***

When people speak of Python they often mean not just the language but also the CPython implementation. Python is actually a specification for a language that can be implemented in many different ways.

### *2.3.1.1 CPython*

CPython is the reference implementation of Python, written in C. It compiles Python code to intermediate bytecode which is then interpreted by a virtual machine. CPython provides the highest level of compatibility with Python packages and C extension modules. If you are writing open-source Python code and want to reach the widest possible audience, targeting CPython is best. To use packages which rely on C extensions to function, CPython is your only implementation option. All versions of the Python language are implemented in C because CPython is the reference implementation.

### *2.3.1.2 PyPy*

PyPy is a Python interpreter implemented in a restricted statically-typed subset of the Python language called RPython. The interpreter features a just-in-time compiler and supports multiple back-ends. PyPy aims for maximum compatibility with the reference CPython implementation while improving performance. If you are looking to increase performance of your Python code, it's worth giving PyPy a try. On a suite of benchmarks, it's currently over 5 times faster than CPython. PyPy supports Python 2.7. PyPy3.1 [11], released in beta, targets Python 3.

### *2.3.1.3 Jython*

Jython is a Python implementation that compiles Python code to Java bytecode which is then executed by the JVM (Java Virtual Machine). Additionally, it is able to import and

use any Java class like a Python module. If you need to interface with an existing Java codebase or have other reasons to need to write Python code for the JVM, Jython is the best choice.
Jython currently supports up to Python 2.7. 2 [12].

### *2.3.1.4 IronPython*

IronPython is an implementation of Python for the .NET framework. It can use both Python and .NET framework libraries, and can also expose Python code to other languages in the .NET framework. Python Tools for Visual Studio integrates IronPython directly into the Visual Studio development environment, making it an ideal choice for Windows developers. IronPython supports Python 2.7. 3 [13].

### *2.3.1.5 PythonNet*

Python for .NET is a package which provides near seamless integration of a natively installed Python installation with the .NET Common Language Runtime (CLR). This is the inverse approach to that taken by IronPython (see above), to which it is more complementary than competing with. In conjunction with Mono, PythonNet enables native Python installations on non-Windows operating systems, such as OS X and Linux, to operate within the .NET framework. It can be run in addition to IronPython without conflict. PythonNet supports from Python 2.6 up to Python 3.5.4 [14] [15].

## 2.2 Web Development Environment

A web integrated development environment (Web IDE or WIDE), also known as cloud IDE, is a browser based IDE that allows for software development or web development. A web IDE can be accessed from a web browser, such as Google Chrome or Internet Explorer, allowing for a portable work environment. A web IDE does not usually contain all of the same features as a traditional, or desktop, IDE, although all of the basic IDE features, such as syntax highlighting, are typically present.

A web IDE, like most websites, is usually composed of two pieces: a frontend and a backend. The frontend is usually written in JavaScript, using AJAX methods to communicate with the backend using a HTTP API, although in some cases, a browser extension or desktop application serves as the frontend and communicates with the backend without the need for a browser. The backend takes care of creating, saving, and opening files, as well as running any terminal commands if the IDE supports it. This setup allows for portability and continuity. The state of the IDE can be saved and reopened on another machine. This also allows for compiling or running programs to continue while the user is away.

Many web IDEs support several programming languages, while others only support a specific language. Most web IDEs allow access to a Command-line interface (CLI) that allows the user to install or run any software that is needed for development, allowing "full" control over the development environment. Open source web IDEs allow for installation on local servers or machines and can be used to give the developer more control over the development environment.

### *2.2.1 XAMPP Web Server*

XAMPP is a free and open source cross-platform web server solution stack package developed by Apache Friends, consisting mainly of the Apache HTTP Server, MariaDB database, and interpreters for scripts written in the PHP and Perl programming languages. XAMPP stands for Cross-Platform (X), Apache (A), MariaDB (M), PHP (P) and Perl (P). It is a simple, lightweight Apache distribution that makes it extremely easy for developers to create a local web server for testing and deployment purposes. Everything needed to set up a web server – server application (Apache), database (MariaDB), and scripting language (PHP) – is included in an extractable file. XAMPP is also cross-platform, which means it works equally well on Linux, Mac and Windows. Since most actual web server deployments used for the application development use the same components as XAMPP, it makes transitioning from a local test server to a live server is extremely easy as well.

# 3. SYSTEM DESCRIPTION

## 3.1 Existing System

In the existing environment, we have individual machines configured with all the programming and scripting language required software's to compile and run the code. They even require various tools for individual language for programming. We require high end machines with high system configuration with various different tools for different languages. To evaluate the task done by the students is an inaccurate and time taking process. Instructor has to go to each machine to evaluate the respective student performance. To host a test on programming or to get the evaluation of the student solution again is manually done by the Instructor.

All this System require many resources and the process is time taking. The process involved is inaccurate and inefficient for both students and faculty. As this require more human efforts and CPU time.

## 3.2 Proposed System

The proposed system is a Web Application which is hosted on a central server with a static address. This web Application hosts the services required for the user by hosting all the software's required for various programming and Scripting languages on server which can be accessed by one or many users at a time in no time and use those resources through internet.

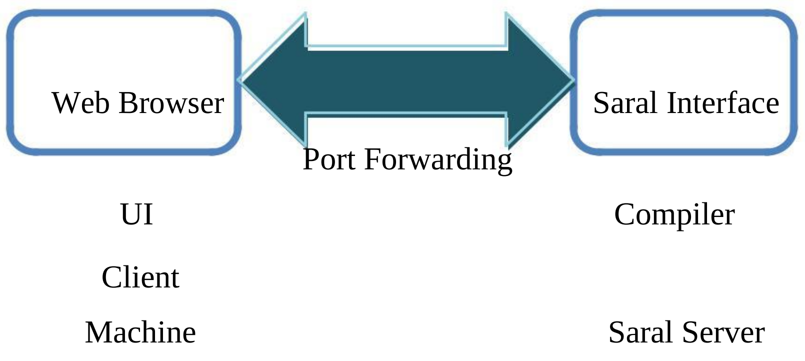


**Figure 3.1: Structure of Saral Interface communication**

The Saral Programming Interface is implemented using HTTP request and response services between client and server, forwarding all the client request to a single server address and with a specific port make it possible for programs to compile source code and these programs can reside on a remote server other than the client machine. The Port forwarding provides a communication channel between the client machine and the Saral-server, as shown in Figure 3.1.

### *3.2.1 Saral User Interface*

Saral Programming Interface System is designed and developed in PHP and JavaScript. Application has well defined and organized structures for each of its modules. The interface gives the user the initial index screen for the user with the brief details of the application. The user enters his login credentials in the login form or registers into the application. User credentials are validated for the minimum required fields and format of each field and displays the validation message on screen. On successful validation of user credentials the user gets access to the home page of the Saral Interface. In case of incorrect credentials error message is displayed on the screen. The home page is specific for each the user with various utilities provided by the Saral Interface. There are different types of functionalities like Practice IDE, Active Contests, upcoming contests and score analysis. Each of these functionalities extends default structure and are common for all the users. The Interface provides Administrator privileges to the application Administrator or contest administrator with add features like creating a contest, deleting a contest, adding a member as user and administrator and removing a member from the database. The application consists of easy navigation system from various pages of application to cash on various utilities of the application.

#### *3.2.1.1 Administrator View*

Saral Interface provides two different views one is for Administrator and another for students, as is the case for all users. The administrator view allows administrators to create new contest and putting information about contest. Administrators can put information including Program name, expected output and logic that gives idea to students (optional). Administrators can also specify number of attempts that each student can use. This information generates unique contest and problem id for type and get stored in the database. The Saral Interface contest type uses Saral external database to store data. There is  need of creating new schema structure to store data. The Saral Interface student view is shown in Figure 3.2 below.

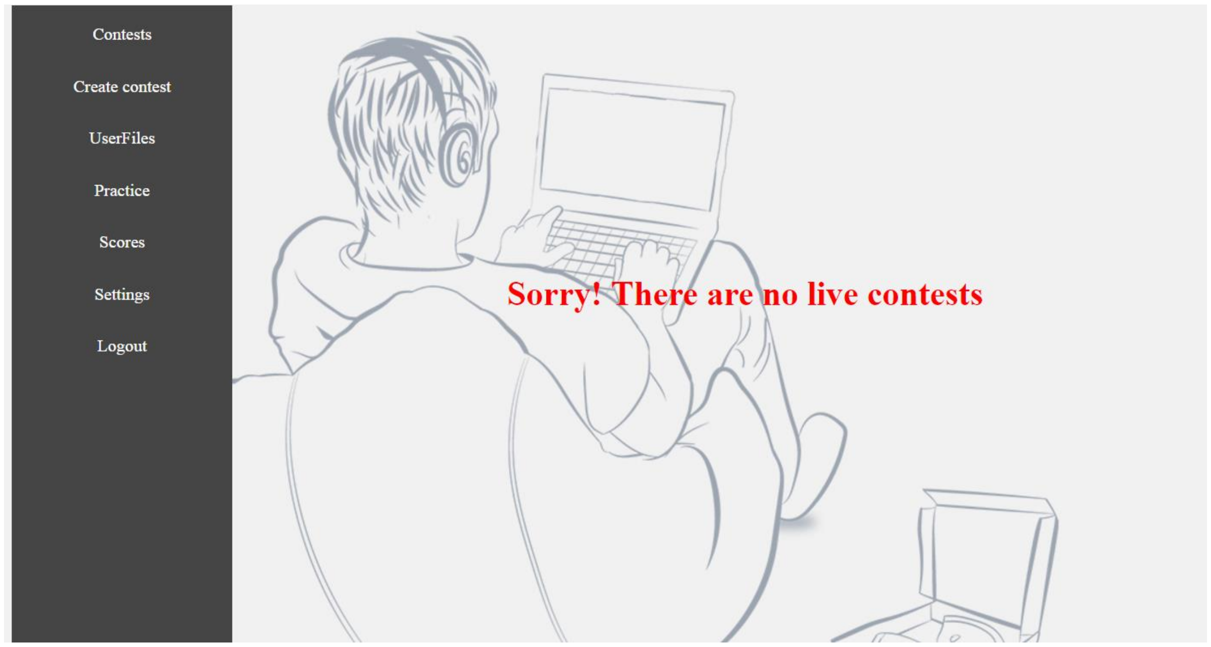
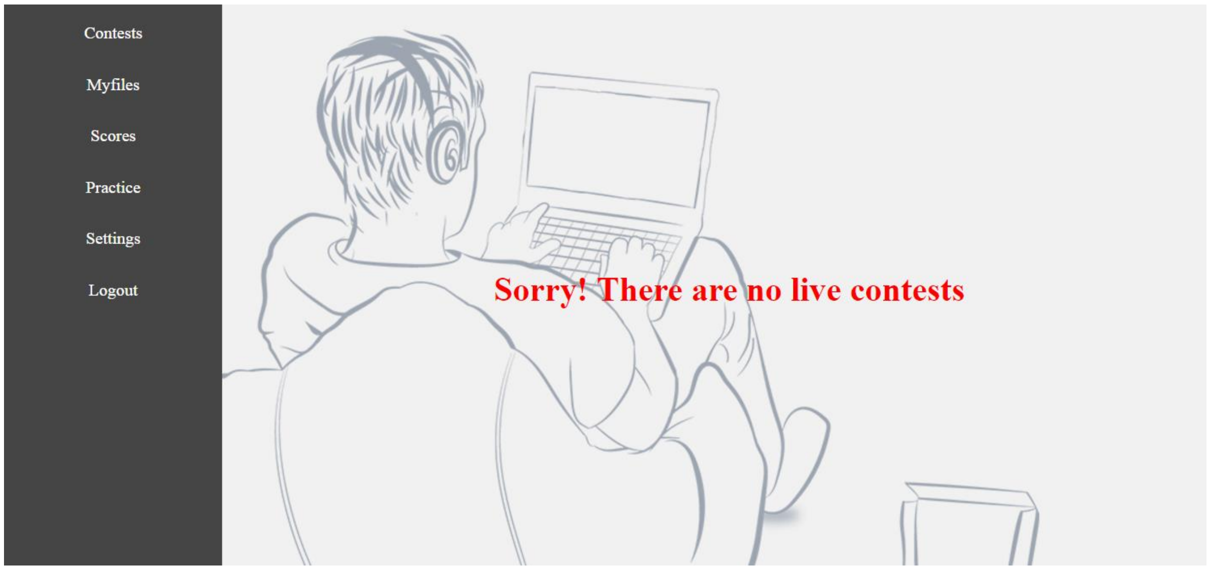


**Figure 3.2: Administrator View**

### *3.2.1.2 Student View*

In the student view, users are provided with a user details on the left side of the screen and the file explorer for navigation of their files. There is a static navigation bar on the top to navigate all along the application. The user has practice and contest features in the practice mode students are provided with text area and submit buttons. Students should enter code in the text area and click on run button which actually send program to compiler and compile it on the server. The output of the compiled program returns back with log details. The log provides better understanding of errors. If students try to compile code which may be violating security on the server, then log detail returns values like permission denied. The Output window shows compilation and execution output on successful compilation only.

There are 3 more buttons available the first button allows students to store their program without changing attempt. These means user can save their current session can retrieve back whenever they want. Another button submits program and starts new session for the student. And last button allows them to submit the program and return to main screen. The Saral Interface student view is shown in Figure 3.3 below.



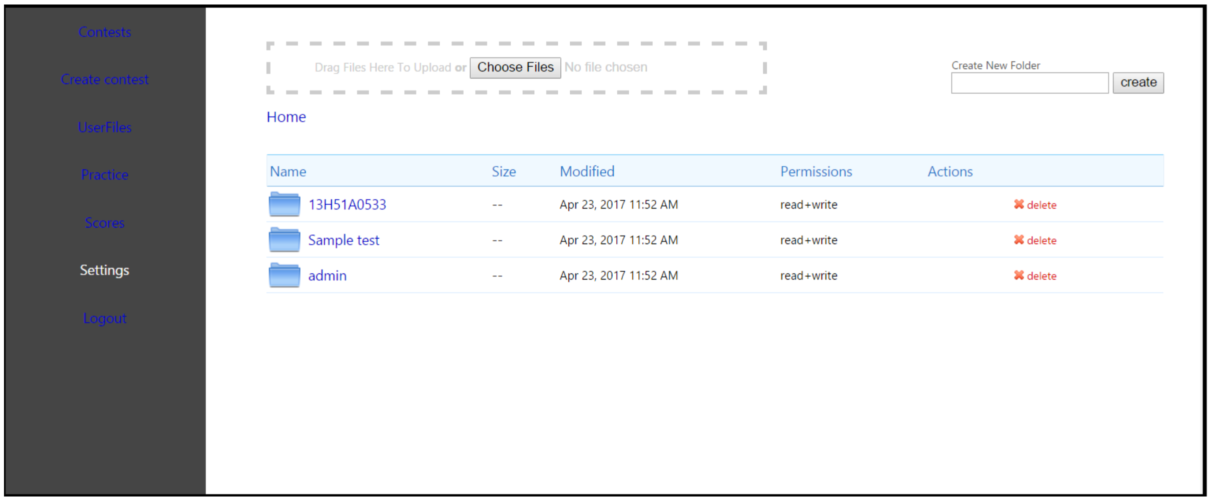
Sorry! There are no live contests

**Figure 3.3: Student View**

### *3.2.2 Session Creation and Data storing*

On the logging -In, a session with a unique Id for the specific user gets created which makes it possible for user to access his user specific directory to store and retrieve files. The Session remains activated until the user gets logs out of the application. Only the user with the specific unique session id can access the files in the remote system. Once the user gets logged off to gain access to his user specific utilities user has to log in again. Created Each click on the submit button compile the code on the server and the statistics about compilation gets stored in the database. The result of compilation is not getting store in the current version of Saral Interface but these will be available in the future enhancements. Teacher can also specify number of attempts which is different than the number of times user tries to compile.

Each attempt allows student to start problem again with the new session. If teacher has selected adaptive mode during contest setup then students would be allowed to compile only once and students are allowed to compile up to maximum numbers of attempts set by Administrator. If teacher has set unlimited attempts then student can compile any number of times. In the Saral Interface practice and contests, student's responses are stored in user specific directory. Each response is in the form of a string array that holds the program written by the student. The Saral Interface student session details are shown in Figure 3.4 below.



| Name | Size | Modified | Permissions | Actions |
|---|---|---|---|---|
| 13H51A0533 | -- | Apr 23, 2017 11:52 AM | read+write | ✖ delete |
| Sample test | -- | Apr 23, 2017 11:52 AM | read+write | ✖ delete |
| admin | -- | Apr 23, 2017 11:52 AM | read+write | ✖ delete |

**Figure 3.4: Student Folder details**

### *3.2.3 File Execution*

As the application supports various programming and scripting languages, the user has to select the language in which the current code which he wants to run and the click the run button. On clicking the run button the backend checks for the specific language selection and forwards the request headers with specific language as an argument to the server. In the server side the compiler of the specific language is invoked with the code as the input. On successful program compilation, Java compiler generates byte code in the form of class file. This file contains executable code in the form of instruction that Java virtual machine executes.

The Saral Interface then uses Runtime.exec() method to execute. On some of the platforms, Runtime.exec() throws stack overflow exception because of limited output buffer size. To avoid this situation, Saral Interface itself provide input and output buffers. On the execution, output gets stored in these buffers and passed back to the user interface. On the execution error, the Saral fetches errors from System. Err to buffer and send back to User Interface. Similarly, other languages directly generate the output from the compiled and interpreted code and is stored in the input and output buffers.

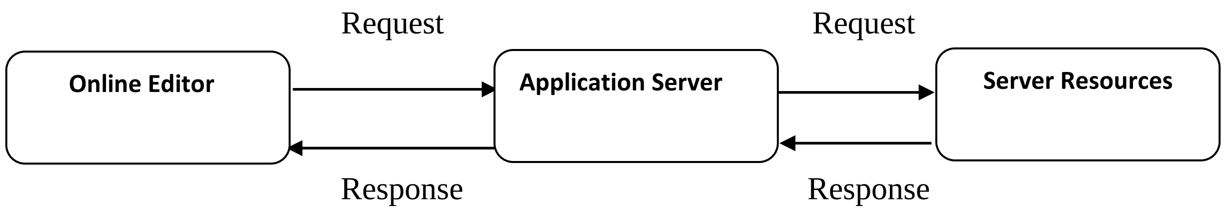


**Figure 3.5: File Execution**

### *3.2.4 Port Forwarding*

In computer networking, port forwarding or port mapping is an application of network address translation (NAT) that redirects a communication request from one address and port number combination to another while the packets are traversing a network gateway, such as a router or firewall. This technique is most commonly used to make services on a host residing on a protected or masqueraded (internal) network available to hosts on the opposite side of the gateway (external network), by remapping the destination IP address and port number of the communication to an internal host. Port forwarding allows remote computers (for example, computers on the Internet) to connect to a specific computer or service within a private local-area network (LAN).

In a typical residential network, nodes obtain Internet access through a DSL or cable modem connected to a router or network address translator (NAT/NAPT). Hosts on the private network are connected to an Ethernet switch or communicate via a wireless LAN. The NAT device's external interface is configured with a public IP address. The computers behind the router, on the other hand, are invisible to hosts on the Internet as they each communicate only with a private IP address.

When configuring port forwarding, the network administrator sets aside one port number on the gateway for the exclusive use of communicating with a service in the private network, located on a specific host. External hosts must know this port number and the address of the gateway to communicate with the network-internal service. Often, the port numbers of well-known Internet services, such as port number 80 for web services (HTTP), are used in port forwarding, so that common Internet services may be implemented on hosts within private networks.

Typical applications include the following:

I. Running a public HTTP server within a private LAN

II. Permitting Secure Shell access to a host on the private LAN from the Internet

III. Permitting FTP access to a host on a private LAN from the Internet

IV. Running a publicly available game server within a private LAN

Administrators configure port forwarding in the gateway's operating system. In Linux kernels, this is achieved by packet filter rules in the iptables or netfilter kernel components. BSD and Mac OS X operating systems prior to Yosemite (OS 10.10.X) implement it in the Ipfirewall (ipfw) module while and Mac OS X operating systems beginning with Yosemite implement it in the Packet Filter (pf) module. When used on gateway devices, a port forward may be implemented with a single rule to translate the destination address and port. (On Linux kernels, this is DNAT rule). The source address and port are, in this case, left unchanged. When used on machines that are not the default gateway of the network, the source address must be changed to be the address of the translating machine, or packets will bypass the translator and the connection will fail. When a port forward is implemented by a proxy process (such as on application layer firewalls, SOCKS based firewalls, or via TCP circuit proxies), then no packets are actually translated, only data is proxied. This usually results in the source address (and port number) being changed to that of the proxy machine.

Usually only one of the private hosts can use a specific forwarded port at one time, but configuration is sometimes possible to differentiate access by the originating host's source address.Unix-like operating systems sometimes use port forwarding where port numbers smaller than 1024 can only be created by software running as the root user. Running with superuser privileges (in order to bind the port) may be a security risk to the host, therefore port forwarding is used to redirect a low-numbered port to another high-numbered port, so that application software may execute as a common operating system user with reduced privileges.

The Universal Plug and Play protocol (UPnP) provides a feature to automatically install instances of port forwarding in residential Internet gateways. UPnP defines the Internet Gateway Device Protocol (IGD) which is a network service by which an Internet gateway advertises its presence on a private network via the Simple Service Discovery Protocol (SSDP). An application that provides an Internet-based service may discover such gateways and use the UPnP IGD protocol to reserve a port number on the gateway and cause the gateway to forward packets to its listening socket.

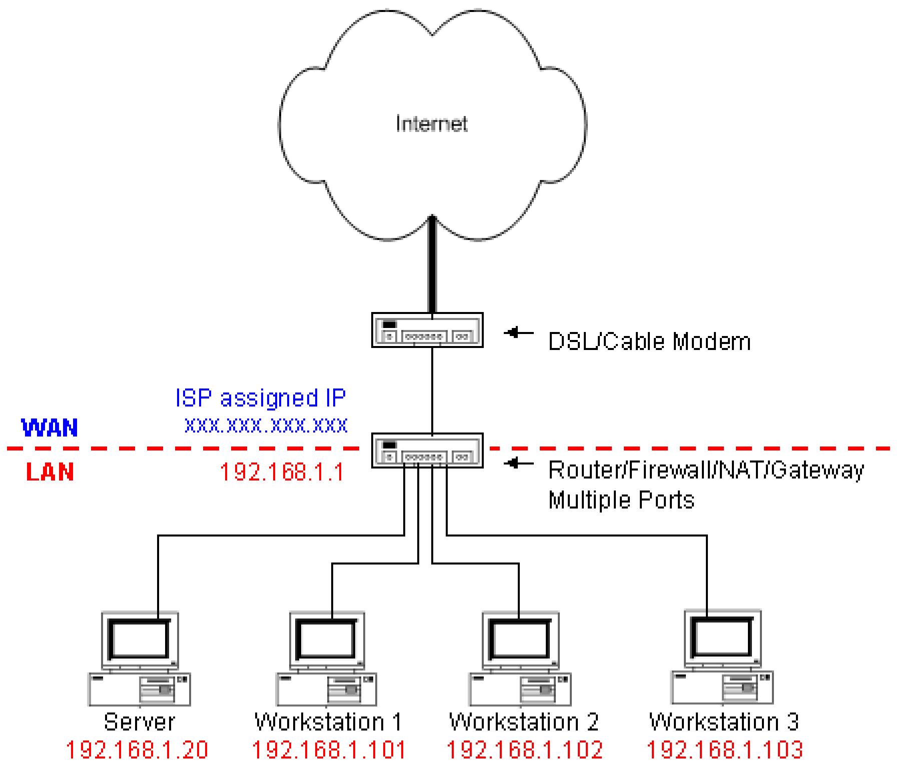


**Figure 3.6 Port Forwarding**

# 4. SYSTEM DESIGN

## 4.1 Overview

### *4.1.1 Input design*

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

***OBJECTIVES:***

1. Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.
2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.
3. When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus, the objective of input design is to create an input layout that is easy to follow

### *4.1.2 Output design*

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.
2. Select methods for presenting information.
3. Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives.

- Convey information about past activities, current status or projections of the Future.
- Signal important events, opportunities, problems, or warnings.
- Trigger an action.
- Confirm an action.

## 4.2 System Design Introduction:

The System Design Document describes the system requirements, operating environment, system and subsystem architecture, files and database design, input formats, output layouts, human-machine interfaces, detailed design, processing logic, and external interfaces.

### *4.2.1 System Architecture*

#### *a. Three Tier Architecture*

In software engineering, multitier architecture (often referred to as n-tier architecture) or multi-layered architecture is a client–server architecture in which presentation, application processing, and data management functions are physically separated. The most widespread use of multitier architecture is the three-tier architecture.

N-tier application architecture provides a model by which developers can create flexible and reusable applications. By segregating an application into tiers, developers acquire the option of modifying or adding a specific layer, instead of reworking the entire application. A three-tier architecture is typically composed of a presentation tier, a domain logic tier, and a data storage tier.

While the concepts of layer and tier are often used interchangeably, one fairly common point of view is that there is indeed a difference. This view holds that a layer is a logical structuring mechanism for the elements that make up the software solution, while a tier is a physical structuring mechanism for the system infrastructure For example, a three-layer solution could easily be deployed on a single tier, such as a personal workstation.

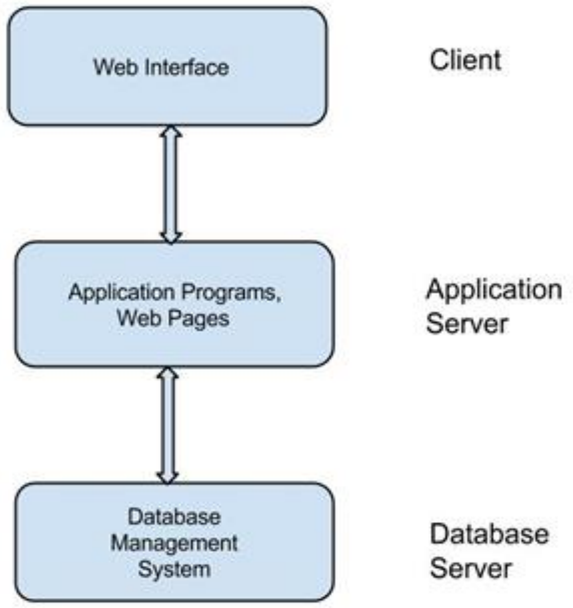


**Figure 4.1: Three tier Architecture**

## b. *The MVC Architecture*

The aim of using MVC architecture is to devide application logic from the presentation and busniess logic.

The anvantage of using MVC:

- Easier maintenance, testing, update the application
- Flexibility in planning and implementing object Model.
- Reuseability and morularoity
- Parallel developmnet of objects
- The application is extensible and scalable

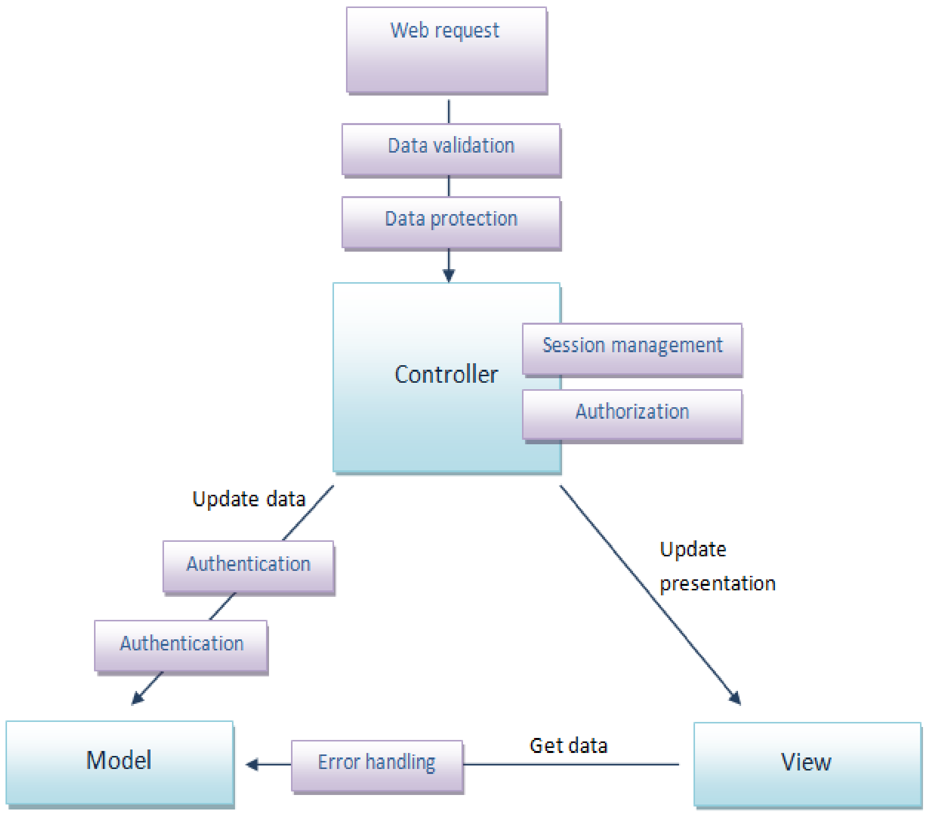


**Figure 4.2: MVC Architecture**

- **Model** – Business logic. It represents data structures. Provide functions for retrieve, insert and update information in our database.
- **View** – Presentation logic. It is the information that is being presented to a user.
- **Controller** – Application logic – contains logic of the page. It joins everything together and generates the page for the user.

### *4.2.2 UML Diagrams*

***Global Use Case Diagrams:***

*Identification of actors:*

**Actor:** Actor represents the role a user plays with respect to the system. An actor interacts with, but has no control over the use cases.

a. ***Graphical Representation***

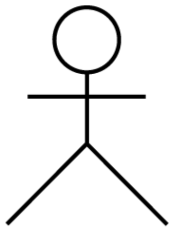


**Figure 4.3: Use Case Actor**

b. **An actor is someone or something that:**

   - Interacts with or uses the system.
   - Provides input to and receives information from the system.
   - Is external to the system and has no control over the use cases.

c. **Actors are discovered by examining:**

   - Who directly uses the system?
   - Who is responsible for maintaining the system? External hardware used by the system.
   - Other systems that need to interact with the system.

**d. *Questions to identify actors:***

- Who is using the system? Or, who is affected by the system? Or, which groups need help from the system to perform a task?
- Who affects the system? Or, which user groups are needed by the system to perform its functions? These functions can be both main functions and secondary functions such as administration.
- Which external hardware or systems (if any) use the system to perform tasks? What problems does this application solve (that is, for whom)?
- And, finally, how do users use the system (use case)? What are they doing with the system?

**e. *The actors identified in this system are:***

- *System Administrator*
- *Student*
- Faculty

***Identification of use cases:***

***Use case:*** A use case can be described as a specific way of using the system from a user's (actor's) perspective.

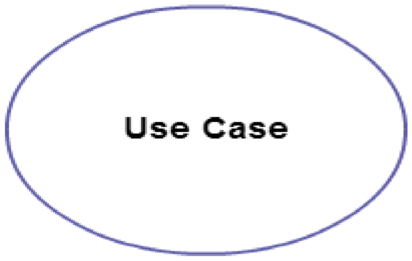


**Figure 4.4: Use Case**

**a. A more detailed description might characterize a use case as:**

- Pattern of behaviour the system exhibits.
- A sequence of related transactions performed by an actor and the system Delivering something of value to the actor.

**b. Use cases provide a means to:**

- capture system requirements.
- communicate with the end users and domain experts test the system.
- Use cases are best discovered by examining the actors and defining what the actor will be able to do with the system.

***c. Guide lines for identifying use cases:***

- For each actor, find the tasks and functions that the actor should be able to perform or that the system needs the actor to perform. The use case should represent a course of events that leads to clear goal.
- Name the use cases.
- Describe the use cases briefly by applying terms with which the user is familiar. This makes the description less ambiguous.

***d. Questions to identify use cases:***

- What are the tasks of each actor?
- Will any actor creates, store, change, remove or read information in the system? What use case will store, change, remove or read this information?
- Will any actor need to inform the system about sudden external changes? Does any actor need to inform about certain occurrences in the system? What use cases will support and maintains the system?

#### *4.2.2.1 Flow of Events*

A flow of events is a sequence of transactions (or events) performed by the system. They typically contain very detailed information, written in terms of what the system should do, not how the system accomplishes the task. Flow of events are created as separate files or documents in your favourite text editor and then attached or linked to a use case using the Files tab of a model element.

**a.** ***A flow of events should include:***

- When and how the use case starts, and ends Use case/actor interactions.
- Data needed by the use case.
- Normal sequence of events for the use case Alternate or exceptional flows.

***b. Construction of Use case diagrams:***

Use-case diagrams graphically depict system behaviour (use cases). These diagrams present a high-level view of how the system is used as viewed from an outsider's (actor's) perspective. A use-case diagram may depict all or some of the use cases of a system.

**c.** ***A use-case diagram can contain:***

- actors ("things" outside the system).
- use cases (system boundaries identifying what the system should do).
- Interactions or relationships between actors and use cases in the system including the associations, dependencies, and generalizations.

#### *4.2.2.2 Relationships in use cases:*

***a. Communication:***

The communication relationship of an actor in a use case is shown by connecting the actor symbol to the use case symbol with a solid path. The actor is said to communicate with the use case.

***b. Uses:***

A Uses relationship between the use cases is shown by generalization arrow from the use case.

***c. Extends:***

The extend relationship is used when we have one usecase that is similar to another usecase but does a bit more. In essence it is like subclass.

### *4.2.3 Sequence Diagrams*

A sequence diagram is a graphical view of a scenario that shows object interaction in a time-based sequence what happens first, what happens next. Sequence diagrams establish the roles of objects and help provide essential information to determine class responsibilities and interfaces.

There are two main differences between sequence and collaboration diagrams: sequence diagrams show time-based object interaction while collaboration diagrams show how objects associate with each other. A sequence diagram has two dimensions: typically, vertical placement represents time and horizontal placement represents different objects.

#### *4.2.3.1 Object:*

An object has state, behaviour, and identity. The structure and behaviour of similar objects are defined in their common class. Each object in a diagram indicates some instance of a class. An object that is not named is referred to as a class instance. The object icon is similar to a class icon except that the name is underlined. An object's concurrency is defined by the concurrency of its class.

#### *4.2.3.2 Message:*

A message is the communication carried between two objects that trigger an event. A message carries information from the source focus of control to the destination focus of control. The synchronization of a message can be modified through the message specification. Synchronization means a message where the sending object pauses to wait for results.

#### *Link:*

A link should exist between two objects, including class utilities, only if there is a relationship between their corresponding classes. The existence of a relationship between two classes symbolizes a path of communication between instances of the classes: one object may send messages to another. The link is depicted as a straight line between objects or objects and class instances in a collaboration diagram. If an object links to itself, use the loop version of the icon.

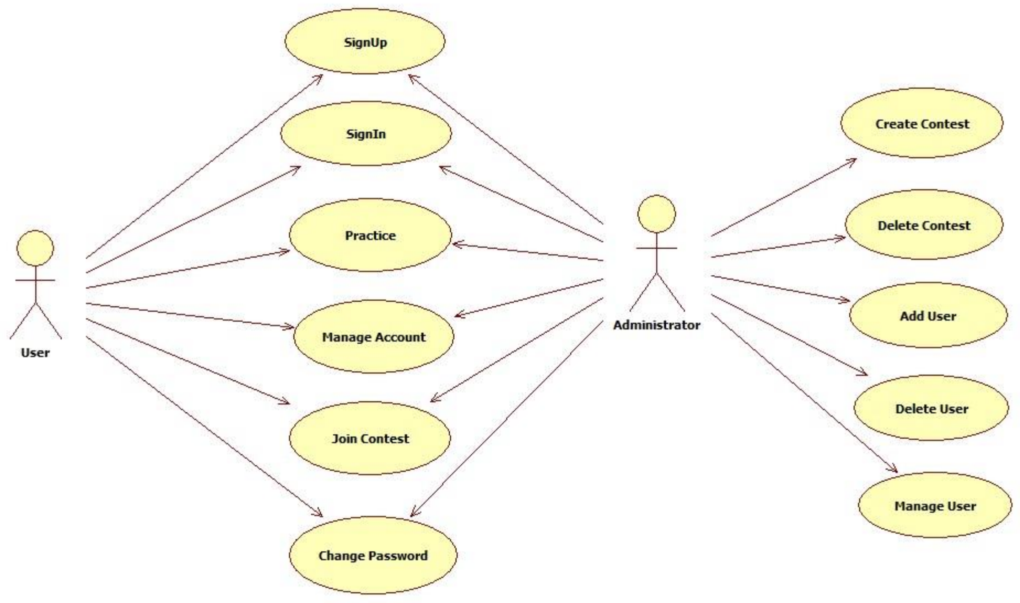


**Figure 4.5: Project use case**

Project use case diagram shows various use cases or functionalities of the actors of the project. As depicted above user and administrator have common functionalities and administrator have few extra functionalities.

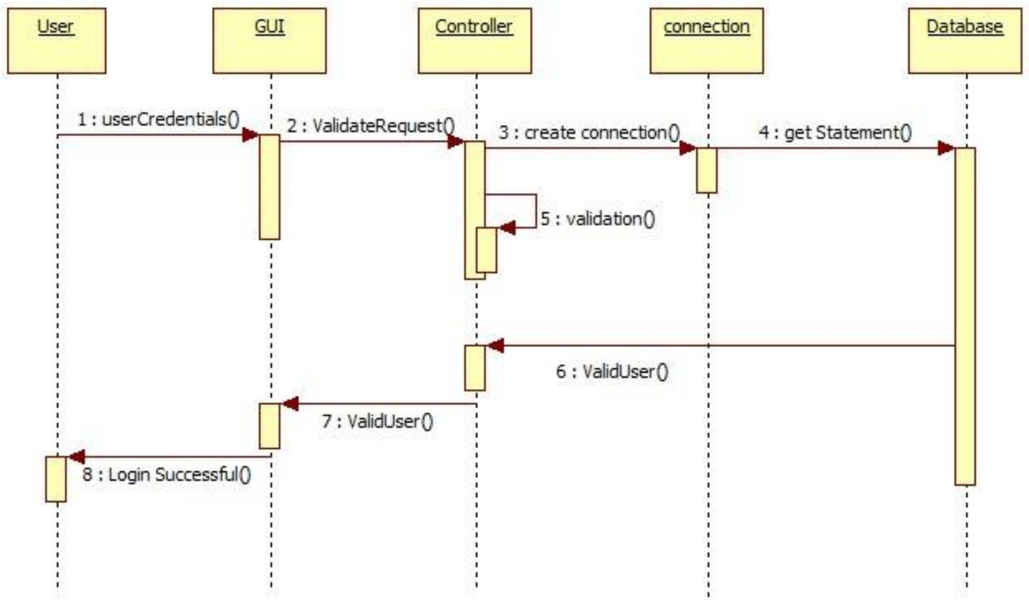


**Figure 4.6: Login Sequence**

Login sequence Diagram describes the sequence of steps for logging into the application.

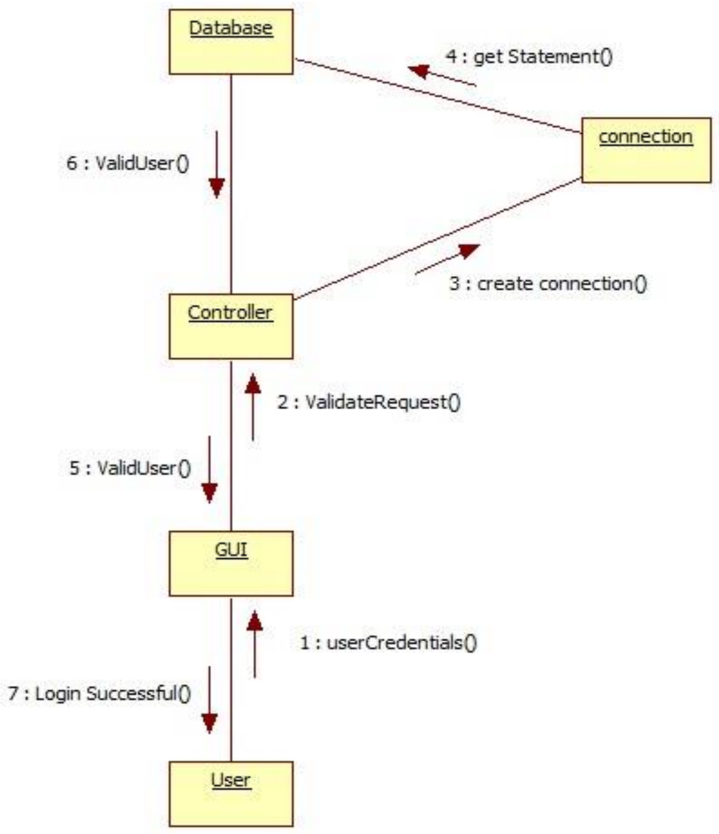


**Figure 4.7: Login Collaboration**

Login Collaboration diagram shows the various nodes and steps in logging into the application.

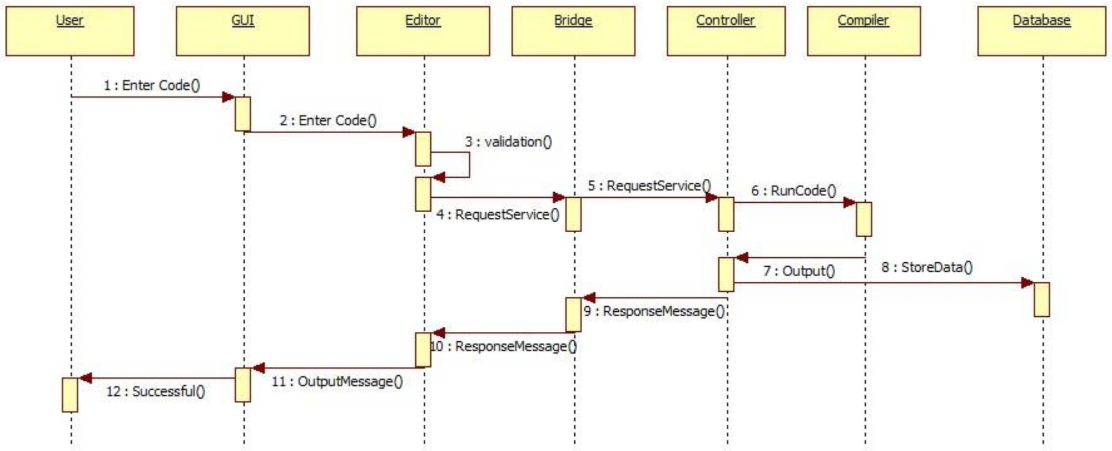


**Figure 4.8: File Execution Sequence**

File Execution Sequence Diagram describes sequence of steps involved in the execution of a code in the Saral Interface.

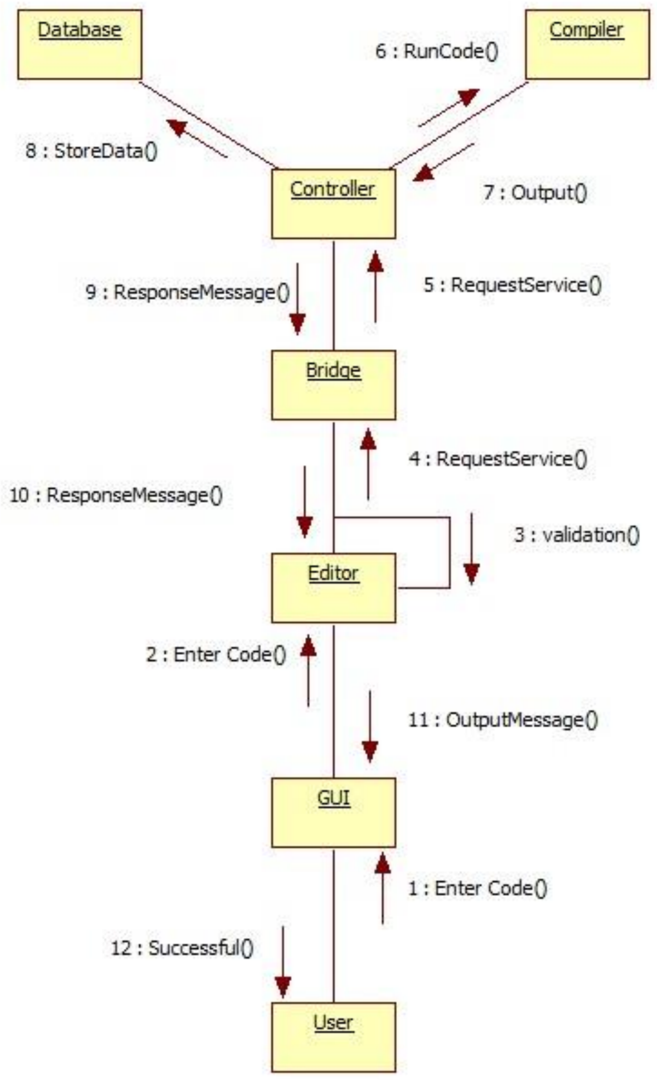


**Figure 4.9: File Execution Collaboration**

File Execution Collaboration diagram shows various nodes, operation and flow involved in the source code present in a file execution in Saral

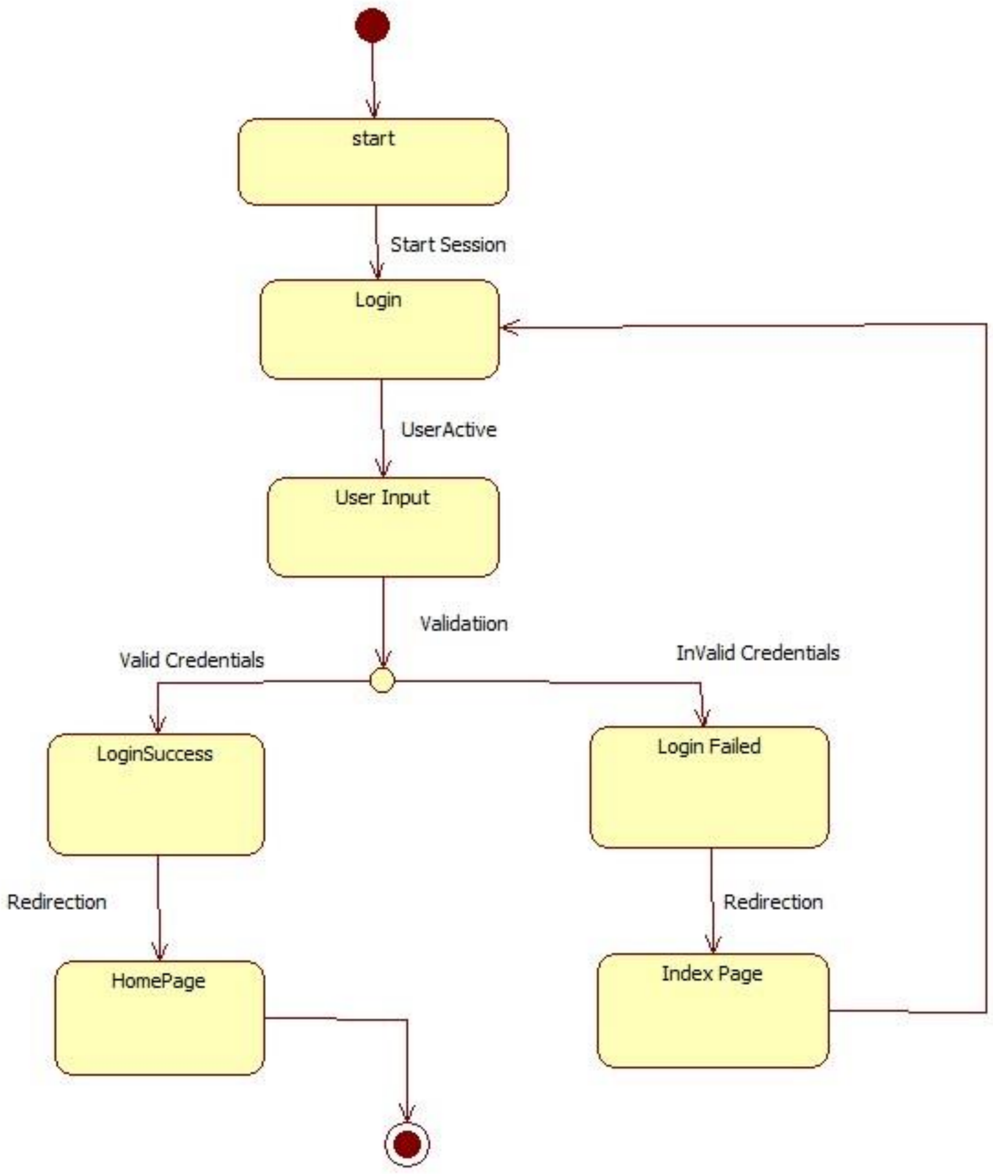


**Figure 4.10: Login State Chart**

Login state Chart Diagram describes various states and flow involved in the Saral application to log into the application.

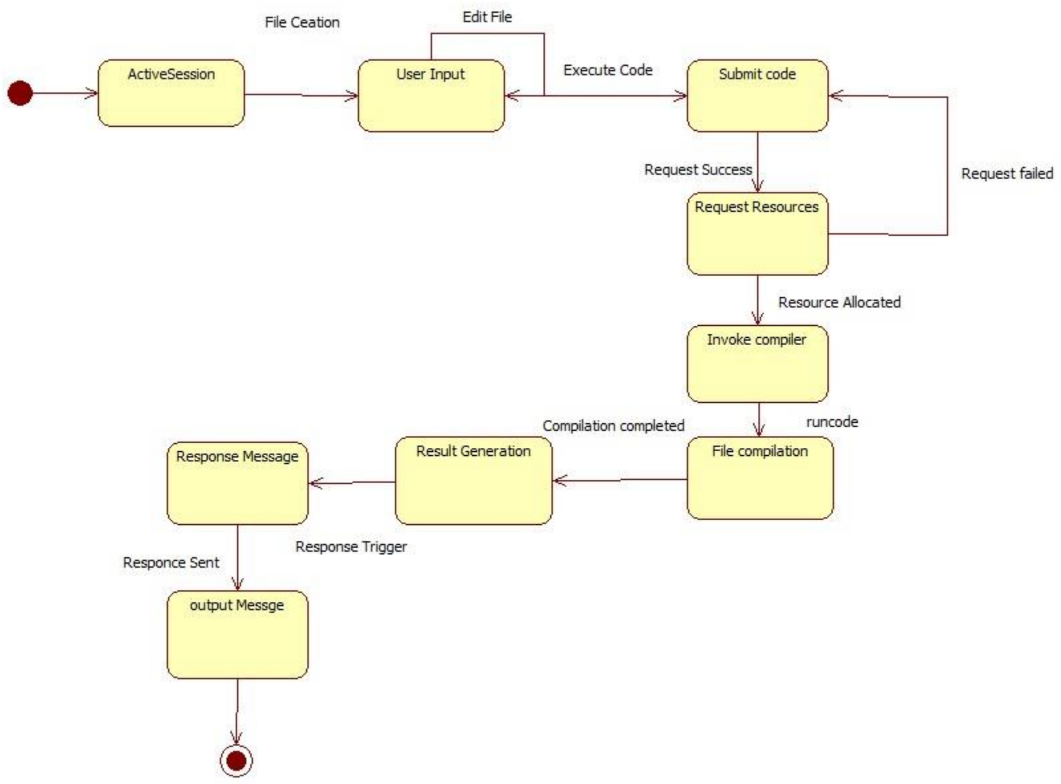


**Figure 4.11: File Execution State Chart**

File Execution state chart diagram describes about the states involved in the execution of code in Saral Interface.

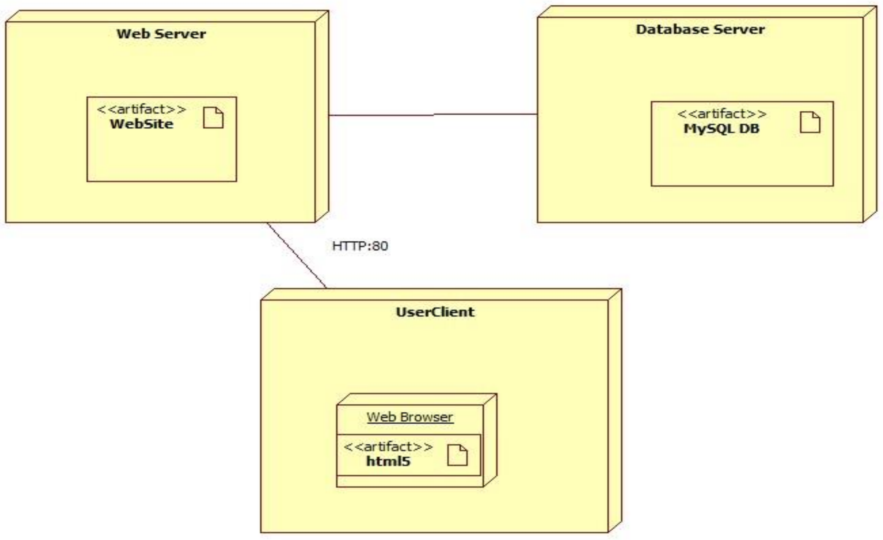


**Figure 4.12: Project Deployment**

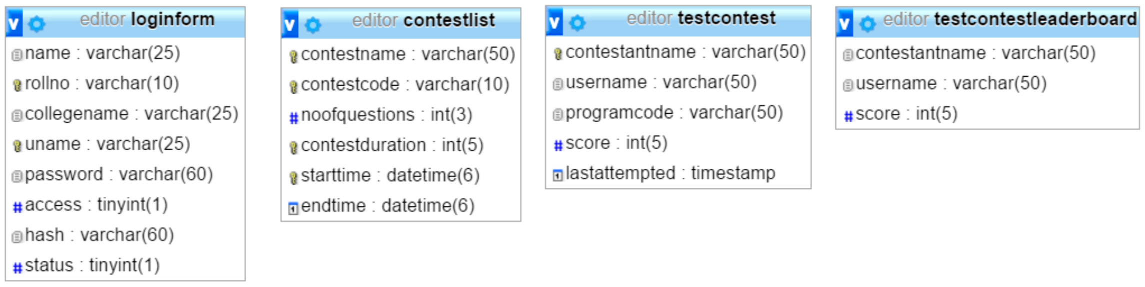
Project Deployment Diagram shows the Various nodes of the project and there instances linked together.

## 4.3 Database Design

The design of the database is the important aspect of developing the project, it deals with the schema, and tables etc. which are involved the software. Designing a perfect database will overcome all the issues which can occur in future like in case of scalability, performance etc.

For the proposed system, we basically need a database with some multiple tables to store the details of the user, contest and the problems. There is a big need of database, as files cannot be sufficient in order to query a logic to store and compare the signature for validity and other information we use a database in this context.

***The structure of the database:***



editor **loginform**
name : varchar(25)
rollno : varchar(10)
collegename : varchar(25)
uname : varchar(25)
password : varchar(60)
access : tinyint(1)
hash : varchar(60)
status : tinyint(1)

editor **contestlist**
contestname : varchar(50)
contestcode : varchar(10)
noofquestions : int(3)
contestduration : int(5)
starttime : datetime(6)
endtime : datetime(6)

editor **testcontest**
contestantname : varchar(50)
username : varchar(50)
programcode : varchar(50)
score : int(5)
lastattempted : timestamp

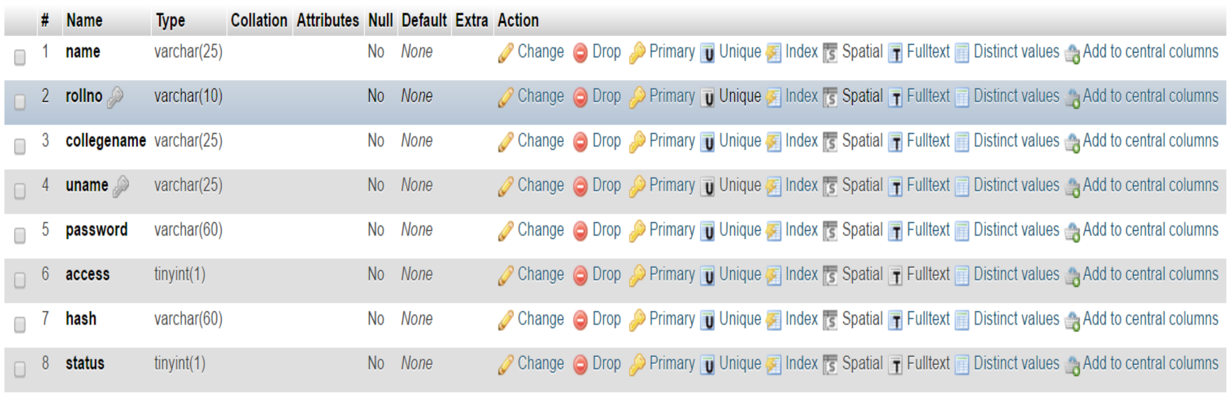
editor **testcontestleaderboard**
contestantname : varchar(50)
username : varchar(50)
score : int(5)

**Figure 4.13: Database structure**

In the Database, we have four table each serves as a source of data for each functionality in the application.

- In the **Loginform table** all the user details for logging into the application is stored.
- In the **Contestlist table** all the details about the contest upcoming and active are stored.
- In the **testcontest table** the details about a single contest problems in it and score details of the particular contest are stored.
- In the **testcontestleaderboard table** score details of the details with their user credentials are stored for analysis of the student or user performance in the particular contest.

***After querying the table, we get the results as follows:***



| # | Name | Type | Collation | Attributes | Null | Default | Extra | Action |
|---|---|---|---|---|---|---|---|---|
| 1 | name | varchar(25) | | | No | None | | Change Drop Primary Unique Index Spatial Fulltext Distinct values Add to central columns |
| 2 | rollno | varchar(10) | | | No | None | | Change Drop Primary Unique Index Spatial Fulltext Distinct values Add to central columns |
| 3 | collegename | varchar(25) | | | No | None | | Change Drop Primary Unique Index Spatial Fulltext Distinct values Add to central columns |
| 4 | uname | varchar(25) | | | No | None | | Change Drop Primary Unique Index Spatial Fulltext Distinct values Add to central columns |
| 5 | password | varchar(60) | | | No | None | | Change Drop Primary Unique Index Spatial Fulltext Distinct values Add to central columns |
| 6 | access | tinyint(1) | | | No | None | | Change Drop Primary Unique Index Spatial Fulltext Distinct values Add to central columns |
| 7 | hash | varchar(60) | | | No | None | | Change Drop Primary Unique Index Spatial Fulltext Distinct values Add to central columns |
| 8 | status | tinyint(1) | | | No | None | | Change Drop Primary Unique Index Spatial Fulltext Distinct values Add to central columns |

**Figure 4.14: Queried table Description**

This figure shows the description of the Loginform table with all its columns and constrains.

# 5. SECURITY

Security of a server is major concern when considering online compilation. Online compilation on the Saral Interface server means allowing users to compile and run programs on the server. A program may be written by a beginner or someone who is experts in the programming. Both scenarios have a potential threat to the server. In the case of expert students, he/she may try to program using different resource requests which may not be configured or protected in the security policies setup by the Administrator. In the case of normal students, they may not be aware of exception handling or memory management. For an example consider the situation where lecturer has asked for a program which does the simple file operation of copying content of one file to another file. In this case if student put extra loop in their program which just keeps making files on the server then the server will run out of memory and eventually crash.

Till now all discussion has considered single users but the Saral Interface is a web based which means that multiple users can access it at the same time and as such may create problems related to resource sharing. If there are hundreds of users try to compile programs that require database or file resources or may be accessing shared memory space. In this case the outcome of this compilation may be deadlock if someone holds a resource and is not releasing it for other students. Thus process management and thread management need to be considered during configuration of the server. This might be one of the reasons that only few of the online learning systems have implemented web-based compilation.

There are few techniques which minimize threats on the servers and help to achieve goals of online compilation. Configuration of the Saral Interface server will play an important role here. Administrator needs to isolate the Saral Interface server and web-based compilation server so that if web-based compilation server crashes, it does not effect on other activities of the same server. Web-based compilation should be configured in such a way that for each user gets restricted amount of memory and resources. There should be an active agent which should throw interrupts after specific time so that if user has not finished compilation within specific time period then it should forcefully terminate and this way deadlocks and other resource intensive operation can be avoided.

More security would be provided with gate keeper which checks every input and output of compilation submitted by student before giving control back to the student. This gate keeper should analyse program by checking the header files used in the program. If user tries to include header files which do system specific operation such as java security manager, java IO then program should restrict the student. But Implementation requires deep analysis of system resources. If a program is asking or setting any system specific changes then those changes should be avoided and the connection should be terminated. Syntax checks and some error handling should be done by the analyser to reduce the load on the server. The light weight analyser could also be implemented using a Java script for the client side validation process. The other activity that analyser should perform is to check the number of clicks on the compile button and avoid it if unnecessary.

The Saral Interface always checks for size of directory allowed for each student before compiling. If this size reaches to maximum allowed size then they get error message and Saral Interface disables submit buttons. In the Saral Interface, students are only allowed to access their directory. They do not have access outside their directory. This framework allows administrator to configure server with limited access of resources. This policy file kept on the web server and gets initialize on the start-up of the web server.

# 6. SYSTEM REQUIREMENTS

## 6.1 Hardware and Software Requirements

| | **Windows, Linux** |
|---|---|
| **XAMPP** | Windows7 and newer version.<br>Kubuntu — Ubuntu with the K Desktop environment.<br>Lubuntu — Ubuntu that uses LXDE.<br>Ubuntu Budgie —— Budgie desktop powered by Ubuntu.<br>Ubuntu GNOME — Ubuntu with the GNOME desktop environment. |
| **Processor** | Dual core, core2duo, Intel I3, I5,I7. |
| **RAM** | 8GB RAM |
| **DISK Space** | Disk space varies depending on size of partition and installation of online help files. The XAMPP Installer will inform you of the hard disk space requirement for your particular partition |
| **Graphics adapter** | 8-bit graphics adapter and display (for 256 simultaneous colours |
| **CD-ROM drive** | For installation from CD. |

**Table 6.1: Minimum Requirements**

| | | **Windows, Linux** | | |
|---|---|---|---|---|
| | **Processor** | **RAM** | **DISK Space** | **Graphics adapter** |
| **XAMPP** | Intel I3, I5, I7 | 8GB | 1 GB for XAMPP only, 5 GB for a typical installation | A 32-bit or 64-bit OpenGL capable graphics adapter is strongly recommended |
| | | | | |

**Table 6.2: Recommended Requirements**

## 6.2 High Level Specifications

- XAMPP with PHP version: 7.1.1.x
- Intel Dual core or core2duo, Intel I3, I5, I7 based personal computer
- 8GB RAM recommended
- 8-bit graphics adapter and display (for 256 simultaneous colors). A 32-bit or 64bit OpenGL capable graphics adapter is strongly recommended.

## 6.3 Low Level Specifications

- Microsoft Windows and Ubuntu supported graphics accelerator card, printer, and sound card.
- Microsoft Word Office 2013, Office 2016.
- TCP/IP is required on all platforms when using a license server.
- Some license types require a license server running FLEXlm 8.0d, which is provided by the XAMPP installer.

## 6.4 Functional Requirements

- The system should process the input given by the user only in text form and store them in the user specific directory in server.
- System shall show the error message to the user when the input given is not in the required format.
- System should detect code present in the editor and send it them server.
- System should retrieve result as a response from the server and display the same to the user on the screen.

## 6.5 Non-Functional Requirements

- Performance: Application compiles code of six different languages and displays the output with in fraction of seconds.
- Functionality: This software will deliver on the functional requirements mentioned in this document.
- Availability: This system will display the result of the code it the editor contains valid code with arguments in it.
- Flexibility: It provides the users to compile or interpret code easily.
- Learn ability: The software is very easy to use and reduces the learning work.
- Reliability: This software will work reliably for low configuration system.

## 6.6 Software Specific Requirements

### *6.6.1 Technologies Used*

To deploy Saral interface using PHP and MySQL it is required to set up Web Development Environment with a server to host your application source files which consist of add-ons for PHP, Perl and MySQL.

#### *6.6.1.1 Setting up Your Web Development Environment with the XAMPP Package*

XAMPP is a popular all-in-one kit that installs Apache, MySQL, and PHP in one procedure. XAMPP also installs phpMyAdmin, a web application you can use to administer your MySQL databases. XAMPP can greatly simplify the installation process. The XAMPP installation installs all the software you need for the Saral application. According to the XAMPP website, XAMPP is intended as a development environment on a local computer. As a development environment, XAMPP is configured to be as open as possible. XAMPP isn't intended for production.

Use it isn't secure as a production environment. Before using XAMPP to make a website available to the public, you need to tighten the security. XAMPP has stable versions available for Windows, Mac, and several versions of Linux. Because XAMPP installs Apache, MySQL, and PHP, it is appropriate to use for installation only on a computer that doesn't have any of the three packages already installed. Because Apache is preinstalled on many Linux and Mac computers and often MySQL and/or PHP are as well, you're most likely to use XAMPP for installation in a Windows environment.

#### *6.6.1.2 Obtaining XAMPP*

You can download XAMPP for Windows from www.apachefriends. org/en/xampp-windows.html. As of this writing, the current version of XAMPP installs the following:

- MySQL
- PHP
- Apache
- PhpMyAdmin

Scroll down the web page until you come to the Download section. Under the listing for XAMPP for Windows, click The Installer link to download the Installer version. The downloaded file is named xampp-win32-, followed by the version and library number, followed by -installer.exe, such as xampp-win32- 1.8.0-VC9-installer.exe. Save the downloaded file on your hard drive in an easy-to-find place, such as the desktop.

### *6.6.1.3 Installing XAMPP*

After you've downloaded XAMPP, follow these steps to install it:

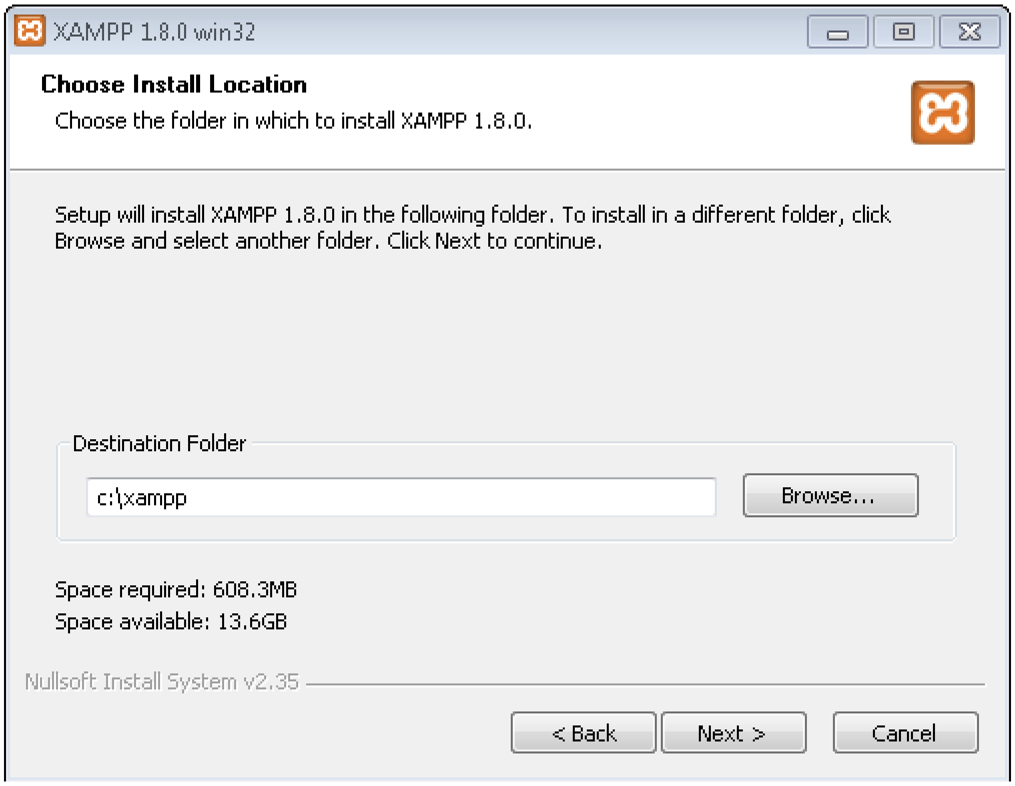
1. **Navigate to the location where you saved the downloaded XAMPP file.**
The file is named something like xampp-win32-1.8.0-VC9-installer.exe.

2. **Double-click the file.**
The Setup Wizard starts. If you're installing on Windows Vista, 7, or 8, you cannot install in the Program Files folder because of a protection problem. Also, PHP sometimes has a problem running if it's installed in a folder with a space in the path or filename, such as Program Files.

3. **Read and click through the next few screens until the Choose Install Location screen appears, as shown in Figure 6.1.**
It's best to accept the default location (c:\xampp) unless you have a really good reason to choose another location. You can click Browse to select another install folder.

4. **When you've chosen the install folder, click Next.**
The XAMPP Options screen appears, as shown in Figure 6.2.

5. **Under Service Section, select the Install Apache as Service and the Install MySQL as Service check boxes.**
This installs the tools as Windows services, which causes them to start automatically when the computer starts.



**Choose Install Location**

Choose the folder in which to install XAMPP 1.8.0.

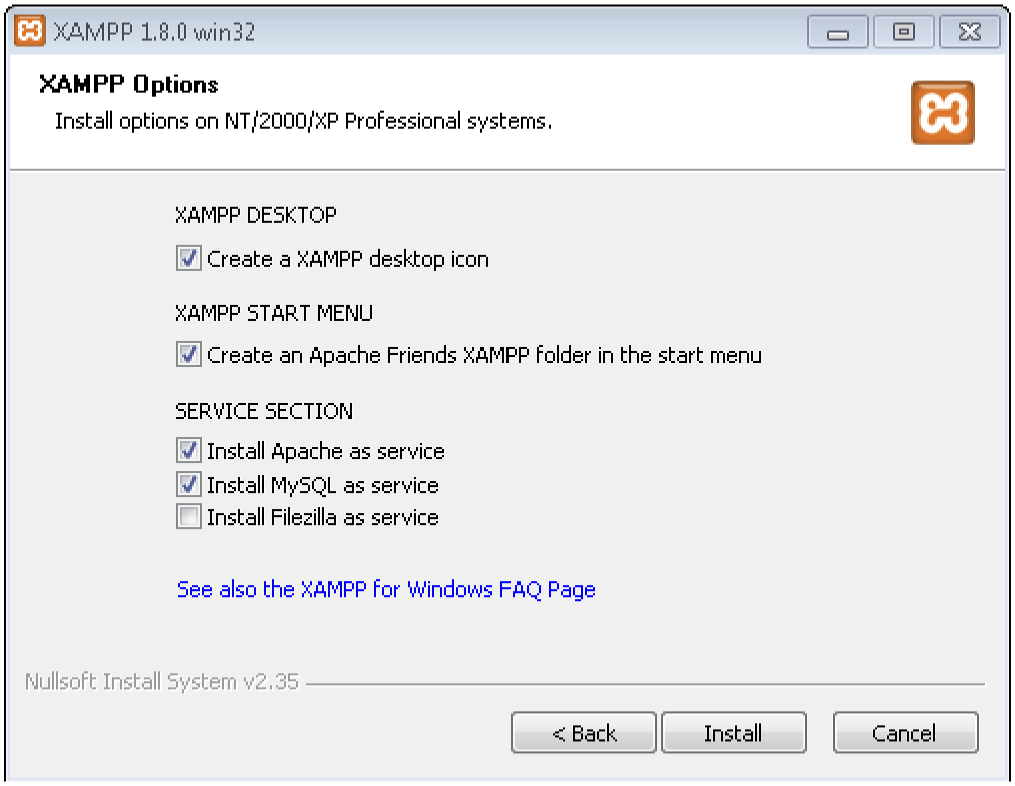
Setup will install XAMPP 1.8.0 in the following folder. To install in a different folder, click Browse and select another folder. Click Next to continue.

Destination Folder

c:\xampp

Space required: 608.3MB
Space available: 13.6GB

**Figure 6.1: The Choose Install Location screen of the Setup Wizard.**



**XAMPP Options**

Install options on NT/2000/XP Professional systems.

XAMPP DESKTOP
☑ Create a XAMPP desktop icon

XAMPP START MENU
☑ Create an Apache Friends XAMPP folder in the start menu

SERVICE SECTION
☑ Install Apache as service
☑ Install MySQL as service
☐ Install Filezilla as service

See also the XAMPP for Windows FAQ Page

**Figure 6.2: The XAMPP Options screen of the Setup Wizard.**

6. **Click the Install button.**

The installation process takes a few minutes to complete. As the installation proceeds, you see various files and components being installed on your system, in the location you specified. A status bar shows the installation progress.

When the installation is complete, the Installation Complete screen appears.

7. **Click Finish.**

A small window opens, and additional messages are displayed. When this part of the installation is finished, a screen displays a message letting you know that the service installation is finished.

8. **Click OK.**

The following question is displayed:

Start the XAMPP Control Panel now?

The screen displays a Yes and a No button.

9. **Click Yes.**

The XAMPP Control Panel appears.

### *6.6.1.4 Using the XAMPP Control Panel*

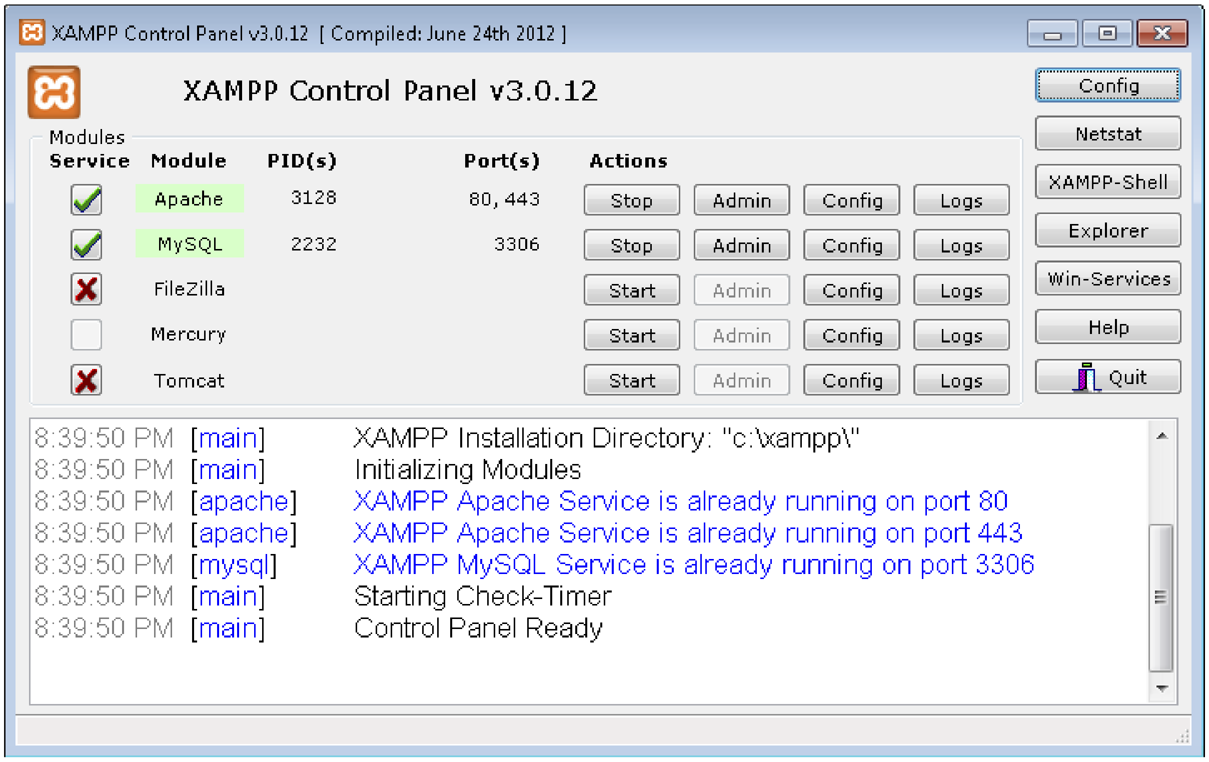
XAMPP provides a Control Panel for efficient management of the software in the XAMPP package. You can use the Control Panel to determine whether Apache and MySQL are currently running and to start or stop them. Before you can use your development environment, Apache and MySQL must be running. This section tells you how to use the Control Panel to start and stop Apache and MySQL. The XAMPP Control Panel can run continuously, ready for you to use at all times. When the Control Panel is running, you see an orange icon in the system tray at the bottom right of your computer screen, as shown in Figure 6.3.

**Figure 6.3: The XAMPP Control Panel icon.**

If the XAMPP icon is in your system tray, you can click it to open the Control Panel. If you don't have the icon in your system tray, you can open the Control Panel by choosing

Start⇨All programs⇨Apache Friends⇨XAMPP⇨XAMPP Control Panel.

If you attempt to open the Control Panel when it's already running, an error message is displayed. Figure 6.4 shows the open Control Panel with Apache and MySQL running. If the installation went smoothly, your control panel will appear like this when you open it after installation. Both Apache and MySQL are shown as running, and the Service check boxes are selected. Your development environment is ready to go.


XAMPP Control Panel v3.0.12 [ Compiled: June 24th 2012 ]
XAMPP Control Panel v3.0.12
Config
Netstat
XAMPP-Shell
Explorer
Win-Services
Help
Quit
Modules

| Service | Module | PID(s) | Port(s) | Actions | | | |
|---|---|---|---|---|---|---|---|
| ✓ | Apache | 3128 | 80, 443 | Stop | Admin | Config | Logs |
| ✓ | MySQL | 2232 | 3306 | Stop | Admin | Config | Logs |
| ✗ | FileZilla | | | Start | Admin | Config | Logs |
| | Mercury | | | Start | Admin | Config | Logs |
| ✗ | Tomcat | | | Start | Admin | Config | Logs |

```
8:39:50 PM  [main]     XAMPP Installation Directory: "c:\xampp\"
8:39:50 PM  [main]     Initializing Modules
8:39:50 PM  [apache]   XAMPP Apache Service is already running on port 80
8:39:50 PM  [apache]   XAMPP Apache Service is already running on port 443
8:39:50 PM  [mysql]    XAMPP MySQL Service is already running on port 3306
8:39:50 PM  [main]     Starting Check-Timer
8:39:50 PM  [main]     Control Panel Ready
```

**Figure 6.4: The XAMPP Control Panel.**

Occasionally, XAMPP isn't able to start either Apache or MySQL as a service during installation. The Control Panel lists the software, showing that it was installed, but the status does not display as running. Both Apache and MySQL must be running before you can use your development environment. To start Apache or MySQL when they are not running, select the Service check box and click the Start button. If XAMPP is successful in starting the software, the status will display as running. If XAMPP is unsuccessful in starting the software as a service, you may need to start the software without selecting the Service check box. See the "Troubleshooting" section at the end of this chapter for more information on starting Apache and MySQL when you have a problem. A Stop button is displayed for each software package that's running. You can stop the software, appropriately enough, by clicking the Stop button. You sometimes need to stop the software, such as when you need to upgrade it. You need to restart Apache whenever you make changes to your PHP configuration, as described throughout this book. To restart Apache, click the Stop button and then, after Apache is stopped, click the Start button. If you close the Control Panel by clicking Exit, the program ends, and you don't have a XAMPP Control Panel icon in your system tray. If you just close the Control Panel window by clicking the X in the upper-right corner of the window, the Control Panel icon remains available in your system tray.

### *6.6.1.5 Testing Your Development Environment*

After you install the XAMPP package and start Apache and MySQL, your environment should be ready to go. You can test your installation by performing the following in any order:

- Opening the XAMPP web page.
- Opening phpMyAdmin.
- Running a test PHP script.

### *6.6.1.6 Opening the XAMPP web page*

To test the XAMPP installation, follow these steps:

1. **Open a browser.**

2. **Type** localhost **in the browser's address bar.**

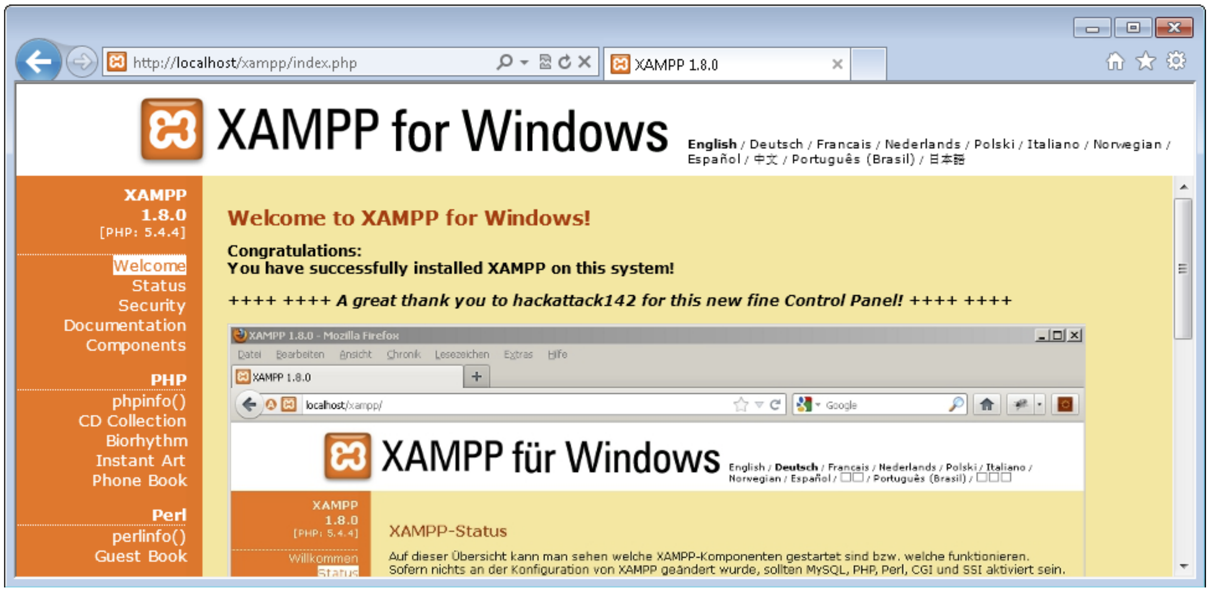
In some cases, if your local machine isn't set up to recognize localhost, you might need to type **127.0.0.1** instead. An XAMPP web page displays, providing a choice of languages. In some cases, XAMPP has already set your language choice and doesn't ask again. In this case, you don't need to do Step 3 because your browser is already at the page shown in Figure 6.5.

3. **Click your preferred language.**

The XAMPP Welcome page displays, as shown in Figure 6.5. If the web page doesn't display, Apache may not be running. Use your Control Panel to manage Apache, as described in the preceding section.

4. **Click the Status link in the panel on the left side of the page.**

A list of software appears, showing which software is activated. MySQL and PHP should be listed as activated. Apache isn't listed because if Apache isn't running, you can't see this page at all.



http://localhost/xampp/index.php
XAMPP 1.8.0
XAMPP for Windows
English / Deutsch / Francais / Nederlands / Polski / Italiano / Norwegian / Español / 中文 / Português (Brasil) / 日本語
XAMPP
1.8.0
[PHP: 5.4.4]
Welcome
Status
Security
Documentation
Components
PHP
phpinfo()
CD Collection
Biorhythm
Instant Art
Phone Book
Perl
perlinfo()
Guest Book
XAMPP 1.8.0 - Mozilla Firefox
Datei Bearbeiten Ansicht Chronik Lesezeichen Extras Hilfe
localhost/xampp/
Google
XAMPP für Windows
Willkommen
Status

Welcome to XAMPP for Windows!

Congratulations:
You have successfully installed XAMPP on this system!

++++ ++++ *A great thank you to hackattack142 for this new fine Control Panel!* ++++ ++++

XAMPP-Status

Auf dieser Übersicht kann man sehen welche XAMPP-Komponenten gestartet sind bzw. welche funktionieren. Sofern nichts an der Konfiguration von XAMPP geändert wurde, sollten MySQL, PHP, Perl, CGI und SSI aktiviert sein.

**Figure 6.5: The XAMPP Welcome page.**

### *6.6.1.7 Testing phpMyAdmin*

From the XAMPP Welcome page (see the preceding section), you can open phpMyAdmin to test whether it's installed. Click the phpMyAdmin link in the Tools section toward the bottom of the left panel. If phpMyAdmin is installed, it opens in your browser. If the phpMyAdmin page doesn’t open, be sure Apache is started. You can manage Apache as described in the “Using the XAMPP Control Panel” section, earlier in this chapter.

### *6.6.1.7 Testing PHP*

To test whether PHP is installed and working, follow these steps:

1. **Locate the directory in which your PHP scripts need to be saved.**

This directory and the subdirectories within it are your *web space.* This is the space where Apache looks for your scripts when you type **localhost**. This directory is called htdocs and is located in the directory where you installed XAMPP, such as c:\xampp\htdocs. You can change the location of your web space in the Apache configuration
file. Changing Apache configuration is described in the section, “Configuring Apache,” later in this chapter.

2. **Create a text file in your web space with the name test.php.**

The file should contain the following content:

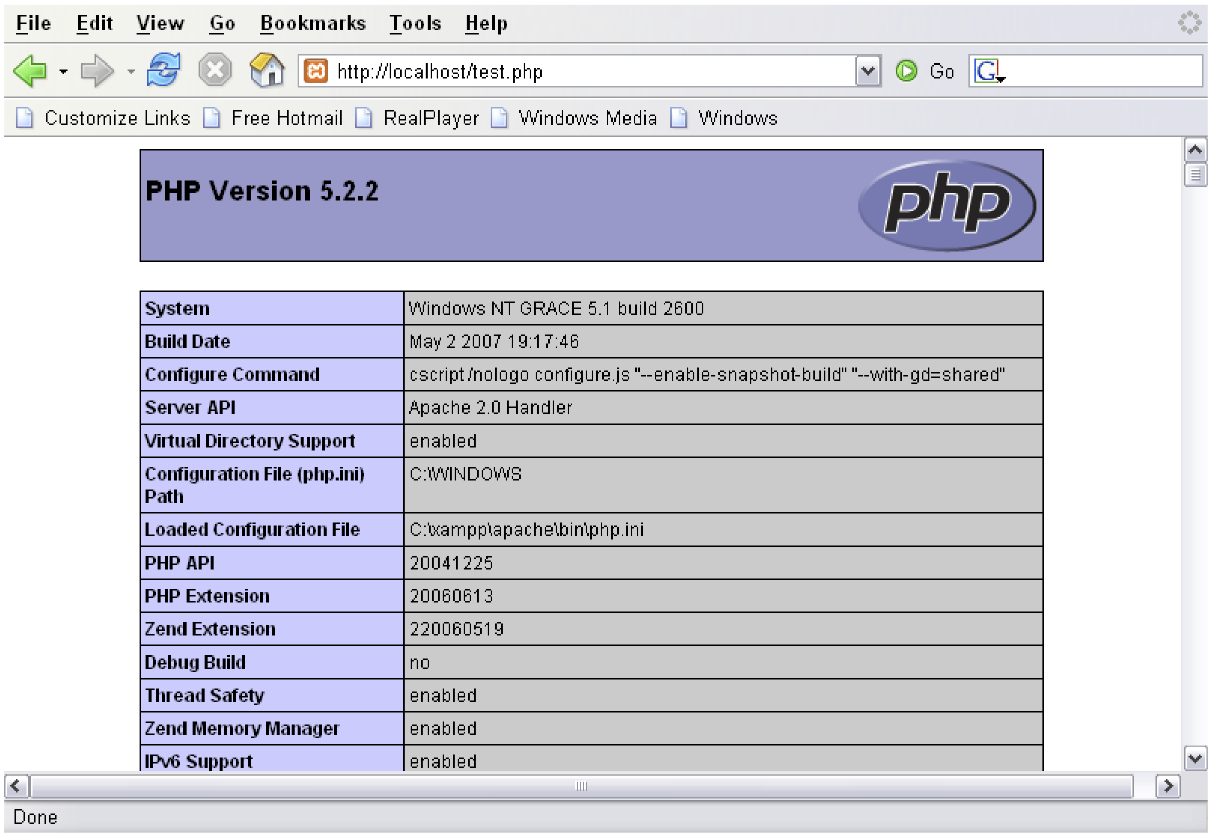
```
<html>
<head><title>PHP test</title></head>
<body>
<?php
phpinfo();
?>
</body>
</html>
```

3. **Open a browser and type** localhost/test.php **into the address bar.**

The output from this PHP script is a long list of settings and variables for your PHP installation, as shown in Figure 6-6.

4. **Scroll down the list to find a section of settings for MySQL.**

The software sections are listed in alphabetical order, starting with bcmath. The MySQL sections are located about halfway down the list. You find two blocks, one headed mysql and one headed mysqli.


File Edit View Go Bookmarks Tools Help
http://localhost/test.php
Go
Customize Links
Free Hotmail
RealPlayer
Windows Media
Windows
Done

PHP Version 5.2.2

| | |
|---|---|
| System | Windows NT GRACE 5.1 build 2600 |
| Build Date | May 2 2007 19:17:46 |
| Configure Command | cscript /nologo configure.js "--enable-snapshot-build" "--with-gd=shared" |
| Server API | Apache 2.0 Handler |
| Virtual Directory Support | enabled |
| Configuration File (php.ini) Path | C:WINDOWS |
| Loaded Configuration File | C:\xampp\apache\bin\php.ini |
| PHP API | 20041225 |
| PHP Extension | 20060613 |
| Zend Extension | 220060519 |
| Debug Build | no |
| Thread Safety | enabled |
| Zend Memory Manager | enabled |
| IPv6 Support | enabled |

**Figure 6.6: Output from the PHP script.**

When your PHP script runs correctly and the output includes a block of settings for MySQL support, your environment is ready for your development work. If the PHP script doesn't run, be sure Apache is started. You can manage Apache as described in the "Using the XAMPP Control Panel" section, earlier in this chapter.

### *6.6.1.8 Configuring Your Development Environment*

Apache, MySQL, and PHP can be configured. Their configuration settings are stored in text files, which you can edit. When XAMPP installs the software, it creates configuration files with default settings so that the software runs with common settings. However, you might need to change the configuration for various reasons. Configuration settings are described throughout the book when the particular feature being configured is discussed. XAMPP installs all the software in the directory you designated during installation, such as c:\xampp, which is the default directory. XAMPP configures the software to look for the configuration files in this directory. If you need to change any configuration settings, you must edit the configuration files in this directory, not in the directories that are mentioned in help files or other documentation for the individual software.

### *6.6.1.9 Configuring PHP*

PHP uses settings in a file named php.ini to control some of its behaviour. PHP looks for php.ini when it begins and uses the settings that it finds. If PHP can't find the file, it uses a set of default settings. XAMPP stores the php.ini file in the apache\bin directory in the main XAMPP folder. For example, if XAMPP is located in the default directory, you edit the file c:\xampp\apache\bin\php.ini to change PHP configuration settings. To configure PHP, follow these steps:

1. **Open the php.ini file for editing in a text editor.**

2. **Edit the settings you want to change.**

   Steps 3 and 4 mention some specific settings that you should *always* change if you're using the specified environment.

3. **Only if you're using PHP 5 *or earlier,* turn off magic quotes.**

   Look for the following line: magic-quotes-gpc On Change On to Off.

4. **Only if you're using PHP 5 *or later,* set your local time zone.**

   Find the line: date.timezone =Remove the semicolon from the beginning of the line. Add the code for your local time zone after the equals sign. For instance, the line might be date. timezone = Asia/India
   You can find a list of time zone codes at www.php.net/manual/en/ timezones.php.

5. **Save the php.ini file.**

6. **Restart Apache so that the new settings go into effect.**

   In general, the remaining default settings allow PHP to run okay, but you might need to edit some of these settings for specific reasons. We discuss settings in the php.ini file throughout this book when we discuss a topic that might require you to change settings.

### *6.6.1.10 Configuring Apache*

The Apache configuration settings are stored in a file named httpd.conf. This file needs some directives in order for PHP to work. XAMPP adds these directives when it installs the software so you don't need to configure Apache to make PHP work. However, you can change some of Apache's behaviour with directives in the httpd.conf file. For instance, you can change where Apache looks for web page files and what port number Apache listens on. Some of the directives you can change. All the Apache directives are described in the Apache website at httpd.apache.org. To change the configuration for Apache that was installed using XAMPP, you need to find the httpd.conf file in the apache\conf folder in the main folder where XAMPP was installed. For instance, if XAMPP is installed in the default directory, the Apache configuration file is c:\xampp\apache\conf\httpd.conf.

### *6.6.1.11 Configuring MySQL*

MySQL creates a configuration file when it's installed. Most people don't need to change the MySQL configuration. However, you might want to change it in order to store your MySQL databases somewhere other than the default location. In fact, the XAMPP installation configures MySQL to look for the data directory in the XAMPP directory, which isn't the default location for MySQL, so XAMPP configures its data directory setting for you. If you want to store your data in a different location, you can change the setting yourself.

To change the configuration for MySQL that was installed using XAMPP, you need to find the my.cnf file in the mysql\bin folder in the main folder where XAMPP was installed. For instance, if XAMPP is installed in the default directory, the MySQL configuration file is c:\xampp\mysql\bin\my.cnf.

### *6.6.1.12 Uninstalling and Reinstalling XAMPP*

If you feel you've made an error and want to install XAMPP again, you need to uninstall it before reinstalling. To uninstall and then reinstall XAMPP, follow these steps:

1. Stop both Apache and MySQL in the XAMPP Control Panel.

See the section, “Using the XAMPP Control Panel,” earlier in this chapter. If you don’t stop Apache and MySQL before you uninstall XAMPP, you might encounter difficulties when you reinstall XAMPP. This is especially true if you started Apache and MySQL as services.

2. Start the uninstall by choosing Start➪All Programs➪Apache Friends➪

XAMPP➪Uninstall.

The first screen of the uninstall procedure opens.

3. Move through the screens and answer the questions.

Click the Next button to move through the screens; answer the questions by selecting the appropriate options. You can save any databases or web pages you have created by selecting the appropriate options. A message is displayed when XAMPP is completely uninstalled.

4. Start the installation procedure again from the beginning.

See the earlier section, “Installing XAMPP,” for details.

### *6.6.2 Troubleshooting*

Occasionally, when you look in the XAMPP Control Panel, you find Apache and/or MySQL listed but not running, and the Service check box isn’t selected. This means that XAMPP was not able to start Apache or MySQL as a service during installation. It’s best to run MySQL and Apache as a service, but not necessary. You can start them without selecting the Service check box and your development environment will work okay. You just need to restart MySQL and Apache in the Control Panel whenever you start your computer. When MySQL and Apache are both running as a service, they start automatically when your computer starts. In most cases, you can start them as a service in the Control Panel using the methods described in this section. First, try selecting the Service check box and clicking the Start button. XAMPP attempts to start the software as a service. If XAMPP is unsuccessful, you will see a message displayed in the bottom box, stating that it isn’t started or that it stopped. A second or third try might be successful. When XAMPP is unsuccessful starting the software as a service over several tries, click the Start button with the Service check box deselected. The software will start. Then, stop the software by clicking the Stop button. Then, start the software again with the Service check box selected. Usually, XAMPP is now able to successfully start both packages as a service. If you are unable to start MySQL and/or Apache as a service even after starting

Them without selecting the Service check box and then stopping them, you can run them without running them as services. They will run okay and your development environment will work — you’ll just have to remember to start them again when you start your computer.

# 7.TESTING

## 7.1 Overview

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product it is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

## 7.2 Types of Testing

### *7.2.1 Unit Testing*

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

### *7.2.2 Integration Testing*

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

### *7.2.3 Functional testing*

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centred on the following items:

Valid Input : identified classes of valid input must be accepted.
Invalid Input : identified classes of invalid input must be rejected.
Functions : identified functions must be exercised.
Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

### *7.2.4 System Testing*

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

### *7.2.5 White Box Testing*

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is used to test areas that cannot be reached from a black box level.

### *7.2.6 Black Box Testing*

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. You cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

## 7.3 Testing Performed

### *7.3.1 Unit Testing*

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

***a. Test strategy and approach***

Field testing will be performed manually and functional tests will be written in detail.

***b. Test objectives***

- ➢ All field entries must work properly.
- ➢ Pages must be activated from the identified link.
- ➢ The entry screen, messages and responses must not be delayed.

***c. Features to be tested***

- ➢ Verify that the entries are of the correct format No duplicate entries should be allowed
- ➢ All links should take the user to the correct page.

### *7.3.2 Integration Testing*

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects. The task of the integration test is to check that components or software applications, e.g. components in a software system or – one steps up – software applications at the company level – interact without error.

***Test Results: All the test cases mentioned above passed successfully. No defects encountered.***

### *7.3.3 Acceptance Testing*

User Acceptance Testing is a critical phase of any project and requires significant participation by the user. It also ensures that the system meets the functional requirements.

***Test Results: All the test cases mentioned above passed successfully. No defects encountered.***

#### *7.3.3.1 Positive Test Case Specifications:*

| Test case number | Test Case | Input | Expected output | Obtained output(pass/fail) |
|---|---|---|---|---|
| 1 | Check Registered or not | Must Register then go to Login | Rows updated | Pass |
| 2 | Valid User Credentials | It Should Accept | Home Page | Pass |
| 3 | Valid contest creation | It should accept | Rows updated | pass |
| 4 | Run code | Must give Output | Output for the code | pass |
| 5 | Get Scores | Submit code and check | Table of Scores Updated | pass |

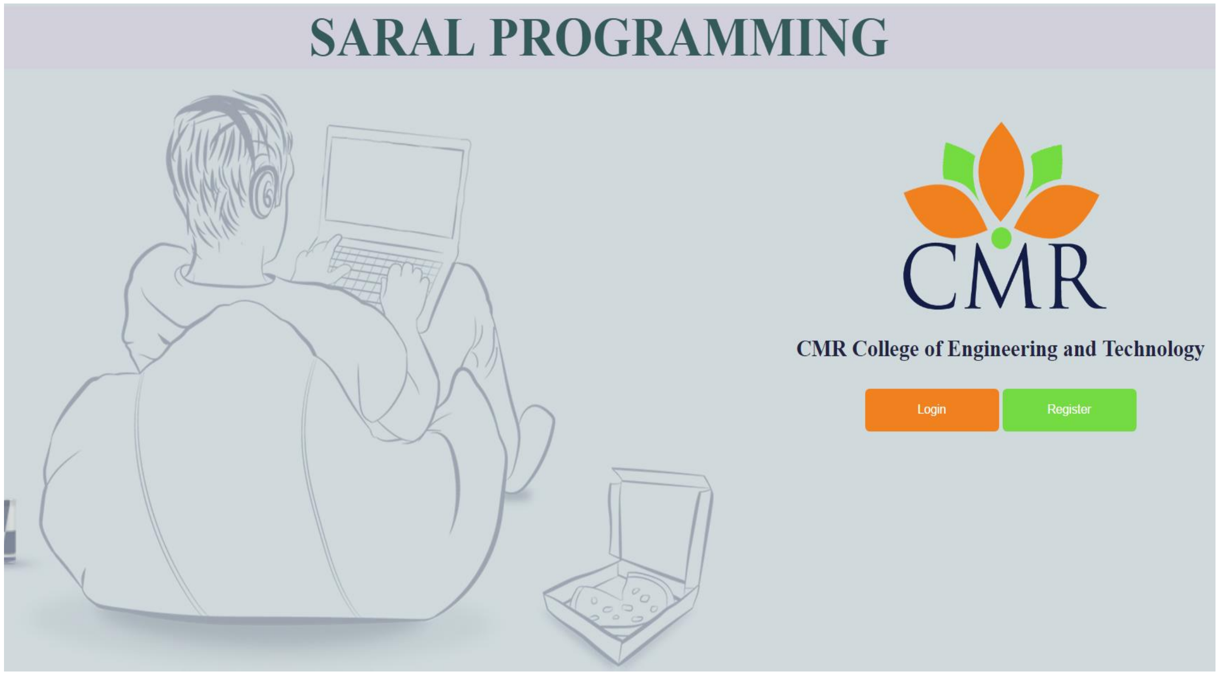
**Table 7.1: Positive Test Cases**

### *7.3.3.2 Negative Test Case Specifications:*

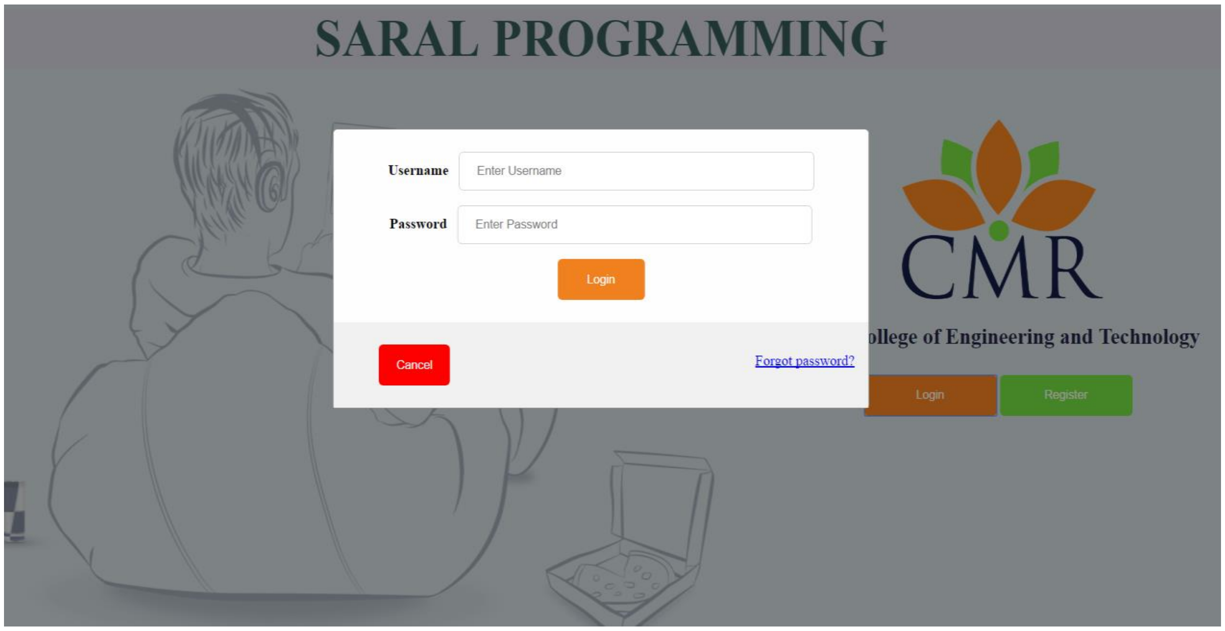
| **Test case number** | **Test Case** | **Input** | **Expected output** | **Obtained Output (pass/fail)** |
|---|---|---|---|---|
| 1 | Empty Credentials | Must not go to Home Page | Enter credentials | Pass |
| 2 | Wrong Credentials | It Shouldn’t Accept | Incorrect Credentials | Pass |
| 3 | Default Class Miss Match | It Shouldn’t Accept | Error Message | Pass |

**Table 7.2: Negative Test Cases**

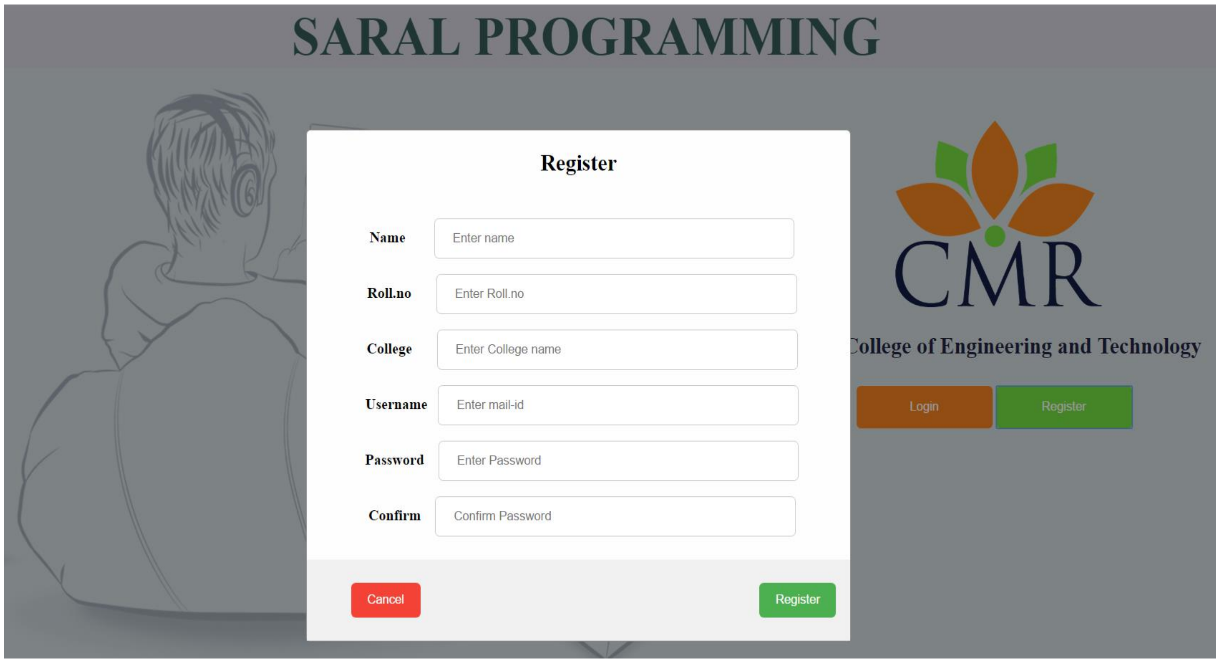
# 8. RESULT SCREENS



**Description: Saral Programming Interface**



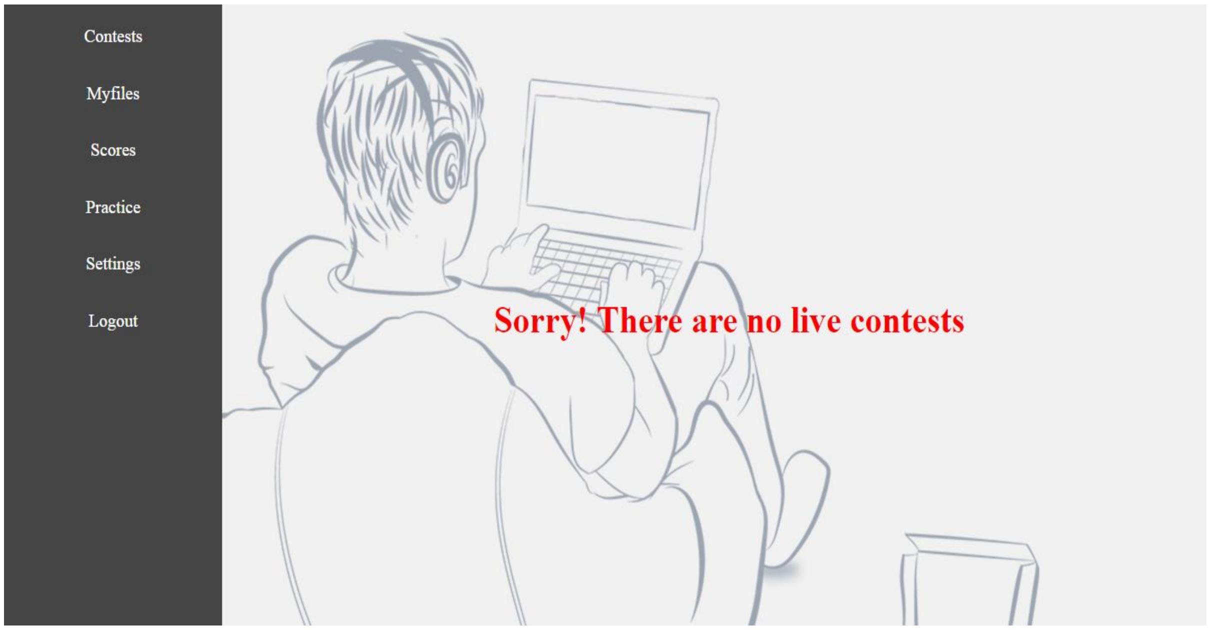
**Description: Login Model**



Register

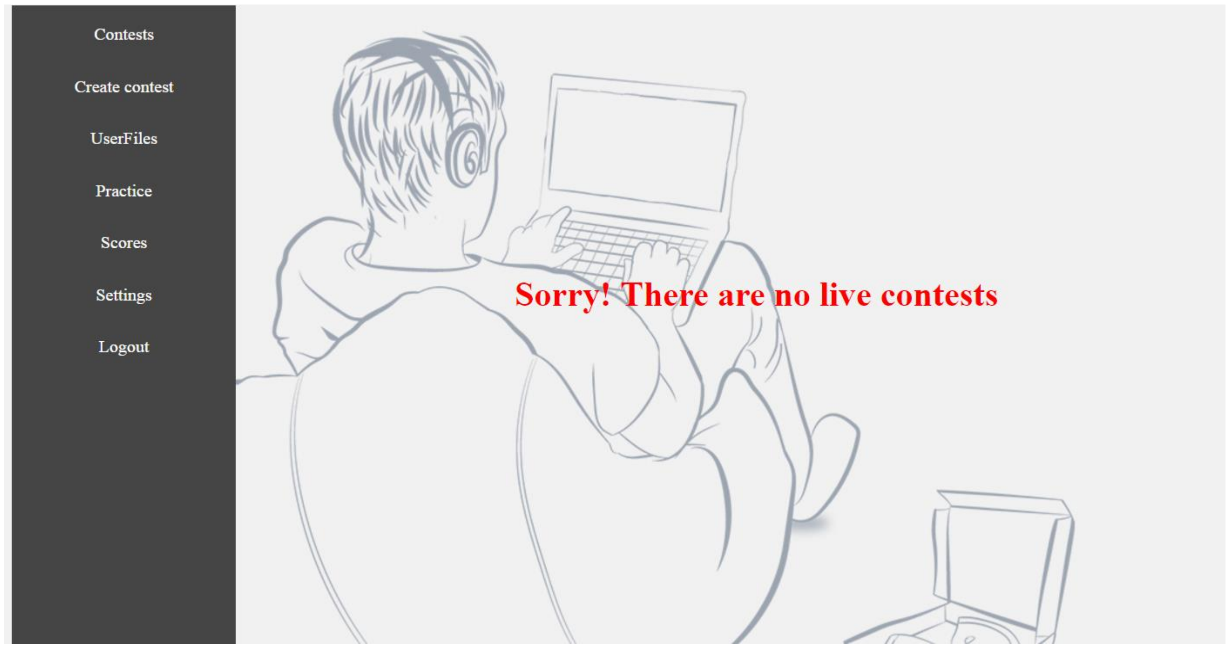
| | |
|---|---|
| Name | Enter name |
| Roll.no | Enter Roll.no |
| College | Enter College name |
| Username | Enter mail-id |
| Password | Enter Password |
| Confirm | Confirm Password |

**Description: Registration Model**



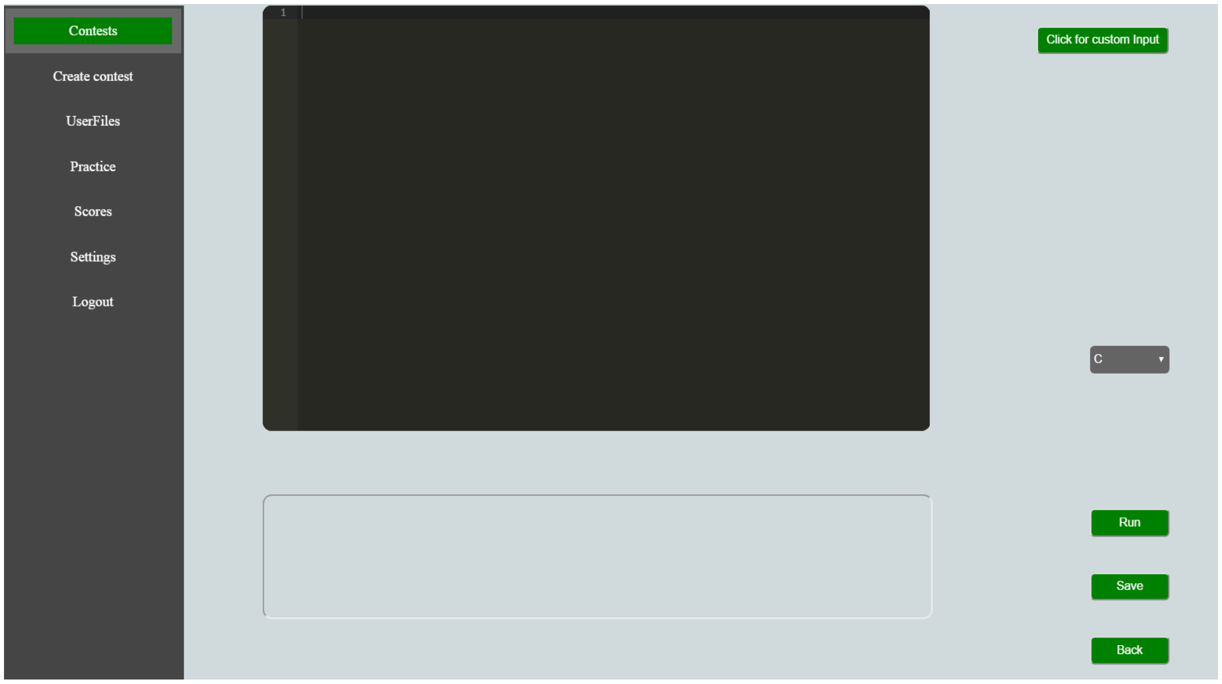
Sorry! There are no live contests

**Description: User Homepage**

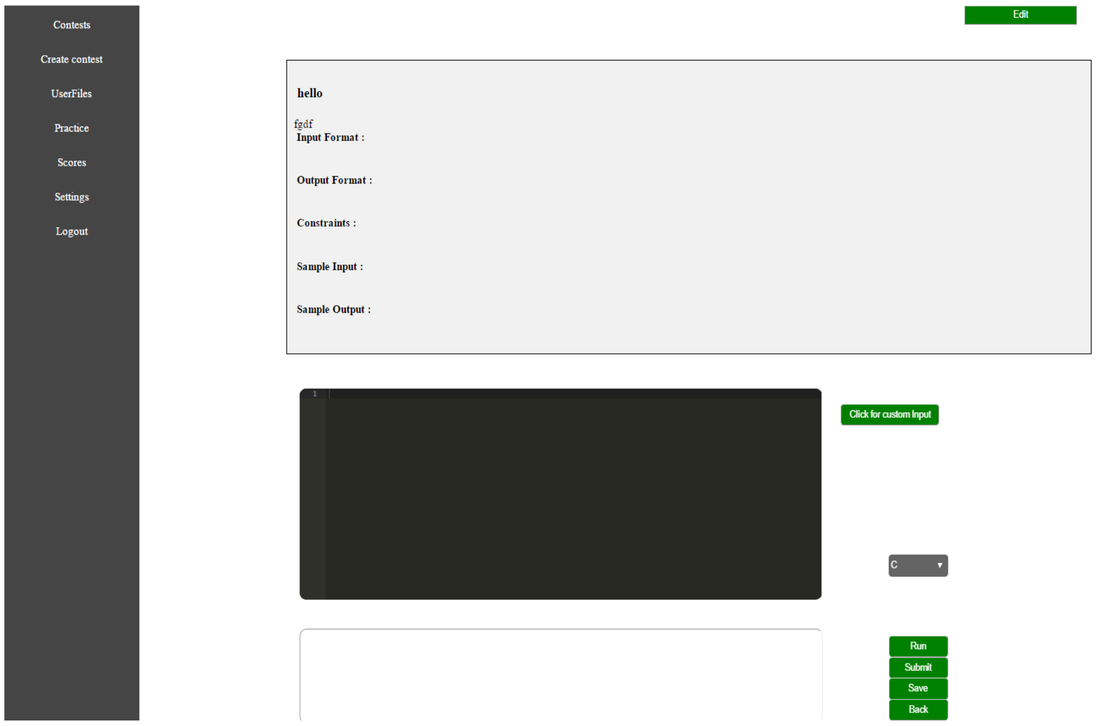


Sorry! There are no live contests

**Description: Administrator Homepage**



**Description: Saral Practice Editor Screen**



hello

fgdf

Input Format :

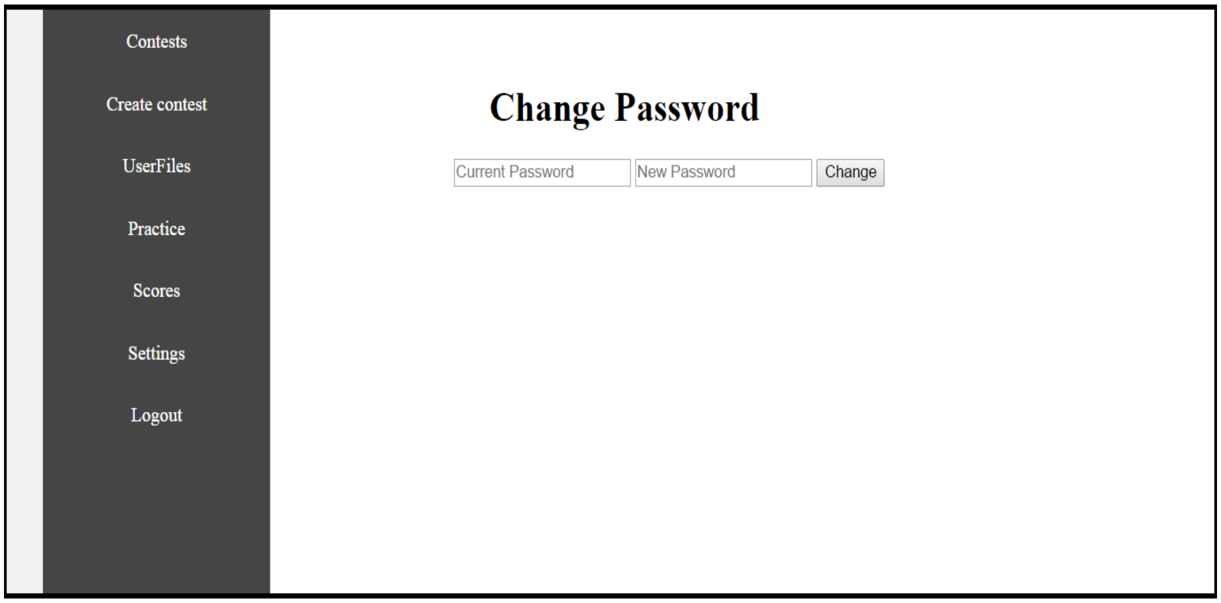
Output Format :

Constraints :

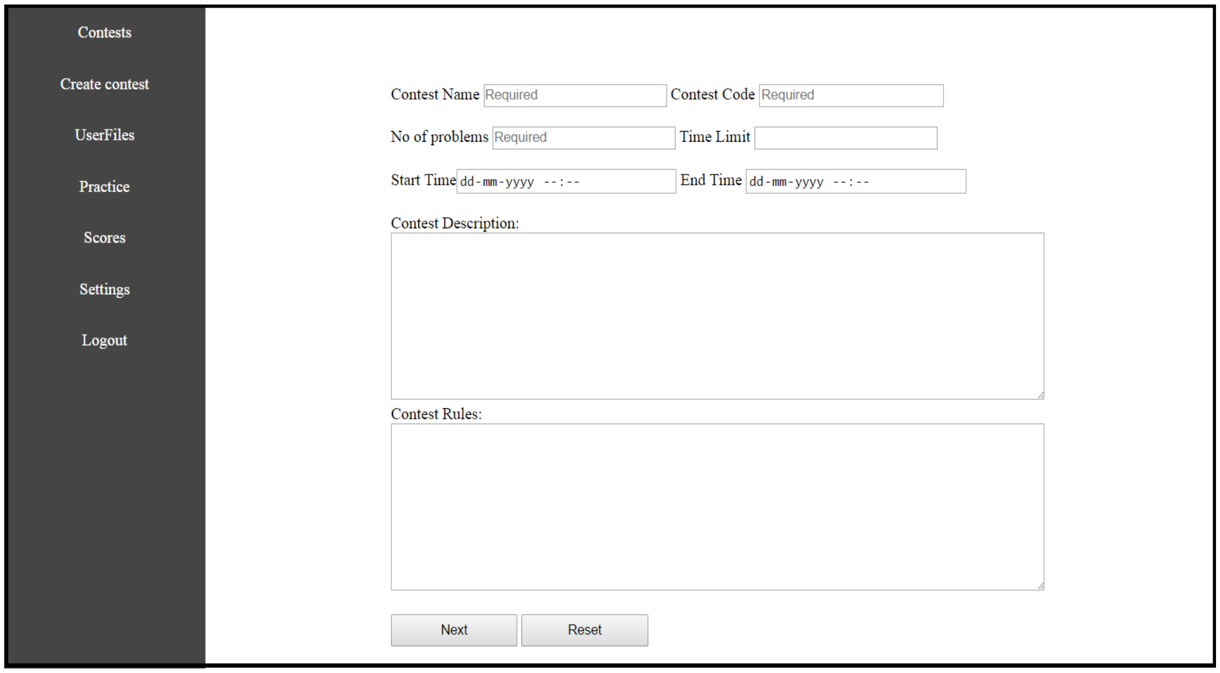
Sample Input :

Sample Output :

**Description: Saral Contest Editor Screen**

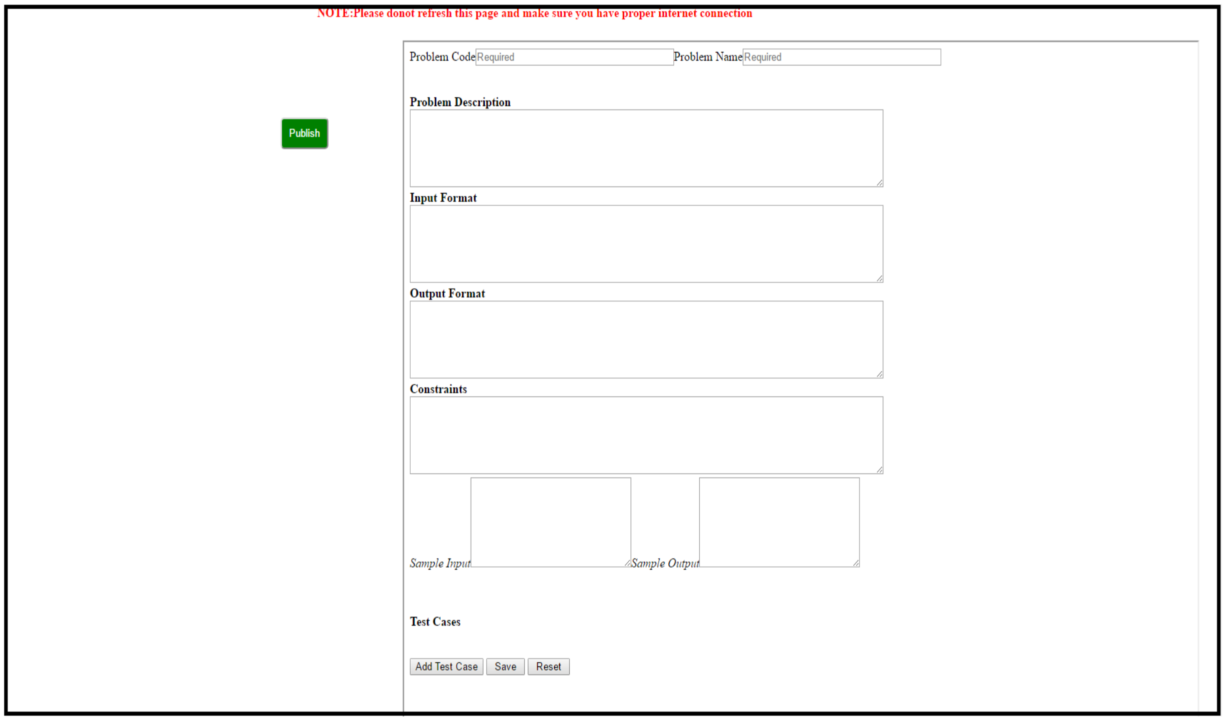


**Description: Change Password Screen**



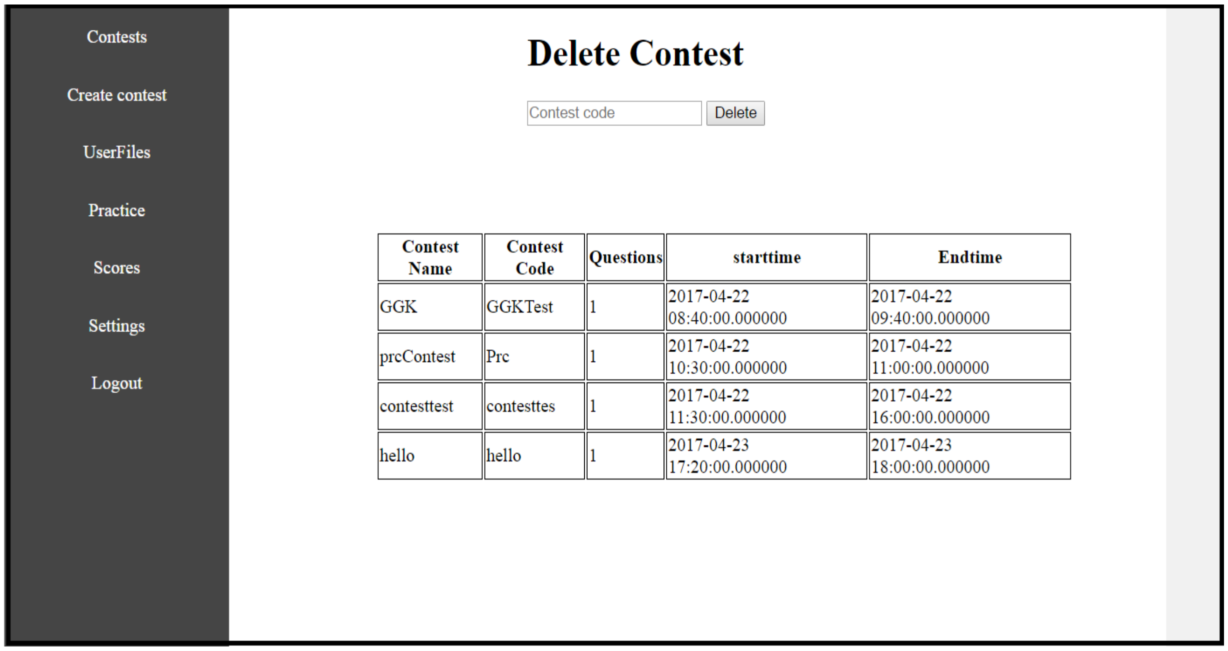
Contest Name Required Contest Code Required
No of problems Required Time Limit
Start Time dd-mm-yyyy --:-- End Time dd-mm-yyyy --:--
Contest Description:
Contest Rules:

**Description: Create Contest Form**


NOTE:Please donot refresh this page and make sure you have proper internet connection

Problem Code Required Problem Name Required
Problem Description
Input Format
Output Format
Constraints
Sample Input Sample Output
Test Cases

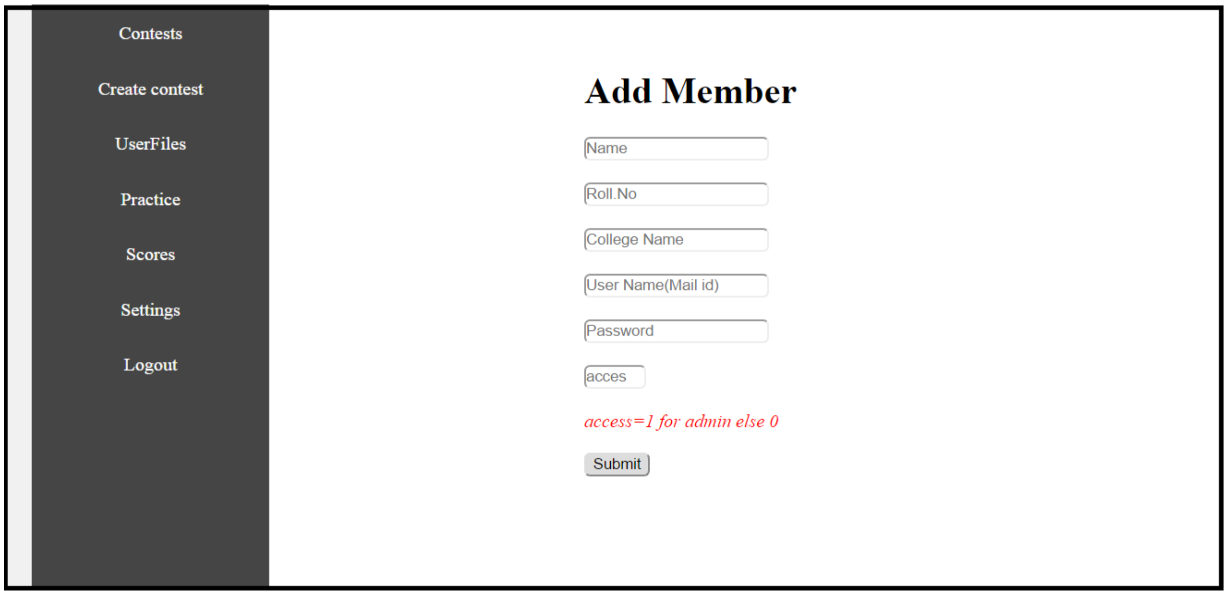
**Description: Create Problem with Test Cases Form**



Delete Contest

| Contest Name | Contest Code | Questions | starttime | Endtime |
|---|---|---|---|---|
| GGK | GGKTest | 1 | 2017-04-22 08:40:00.000000 | 2017-04-22 09:40:00.000000 |
| prcContest | Prc | 1 | 2017-04-22 10:30:00.000000 | 2017-04-22 11:00:00.000000 |
| contesttest | contesttes | 1 | 2017-04-22 11:30:00.000000 | 2017-04-22 16:00:00.000000 |
| hello | hello | 1 | 2017-04-23 17:20:00.000000 | 2017-04-23 18:00:00.000000 |

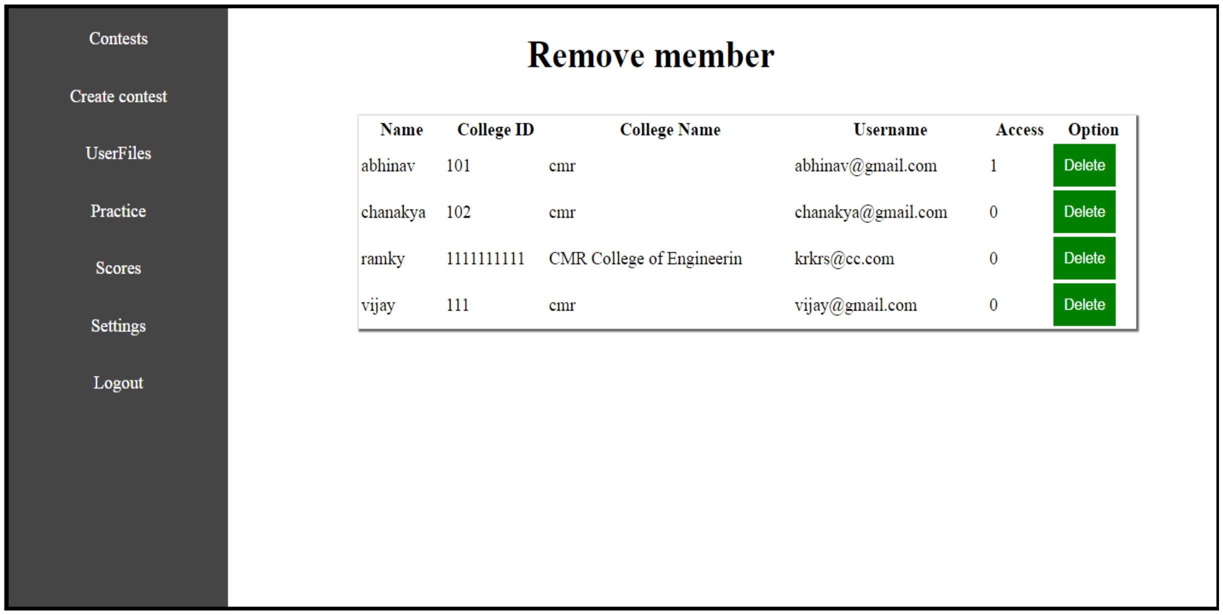
**Description: Delete Contest Screen**



Add Member

*access=1 for admin else 0*

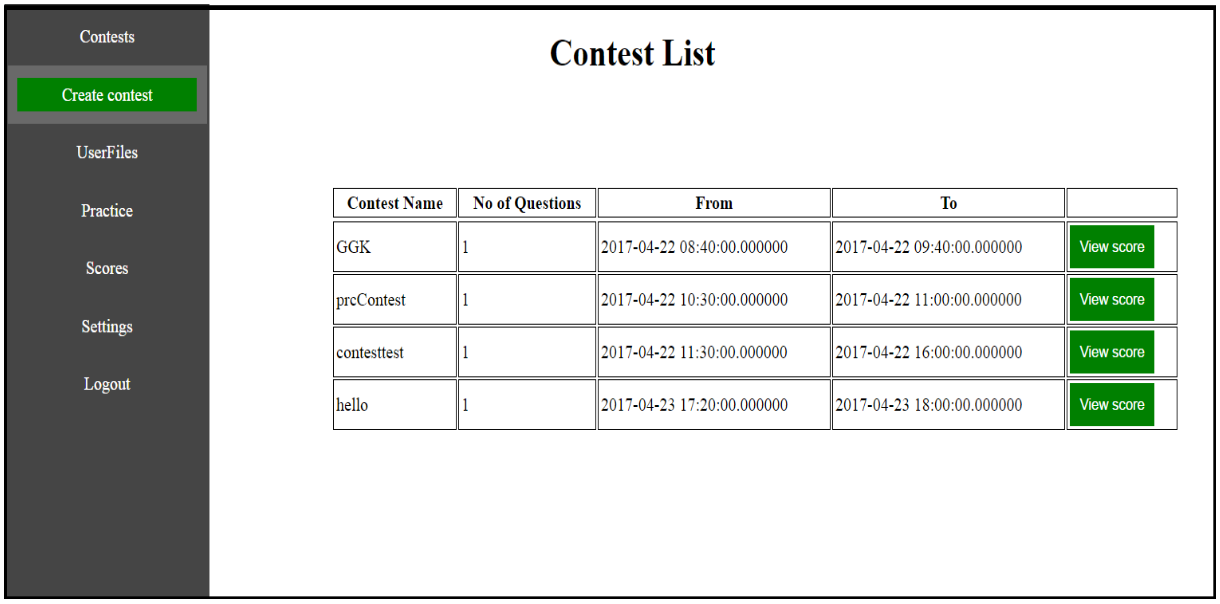
**Description: Add Member form**



Remove member

| Name | College ID | College Name | Username | Access | Option |
|---|---|---|---|---|---|
| abhinav | 101 | cmr | abhinav@gmail.com | 1 | Delete |
| chanakya | 102 | cmr | chanakya@gmail.com | 0 | Delete |
| ramky | 1111111111 | CMR College of Engineerin | krkrs@cc.com | 0 | Delete |
| vijay | 111 | cmr | vijay@gmail.com | 0 | Delete |

**Description: Remove Member Screen**



Contest List

| Contest Name | No of Questions | From | To | |
|---|---|---|---|---|
| GGK | 1 | 2017-04-22 08:40:00.000000 | 2017-04-22 09:40:00.000000 | View score |
| prcContest | 1 | 2017-04-22 10:30:00.000000 | 2017-04-22 11:00:00.000000 | View score |
| contesttest | 1 | 2017-04-22 11:30:00.000000 | 2017-04-22 16:00:00.000000 | View score |
| hello | 1 | 2017-04-23 17:20:00.000000 | 2017-04-23 18:00:00.000000 | View score |

**Description: Score Table Screen**

# 9. CONCLUSION

## 9.1 Conclusion

The Saral Programming Interface presented in this document. This provides a key solution for online compilation of code. This application enables students to compile their programs without having to configuring their machine on various languages for program compilation. This also allows students to do the programming online, anywhere, anytime. The provided detailed diagnostics and log helps students to find compilation and execution errors much easily. This application can be used in the future to implement online IDE platform which would allow students to create and maintain their project online with capability of version control. It also generates detailed statistics of student’s compilations that can help Administrator to improve their teaching methodologies. Administrator can also restrict student’s compilation by specifying limited attempts. The Saral Interface also developed using secure and robust approach to online compilation that overcomes limitations in the design of the other Online compilers.

## 9.2 Outcomes

The current version of the Saral Programming Interface is capable of performing all of the functions outlined in the above design description. This includes the capacity to:

- Provide the teacher and student User interface.
- Runs independently from the local compilers.
- Allow Administrator to create a contest with questions.
- Allow students to attempt the programming problem online by performing successive compilations.
- Have error and other diagnostic messages returned to the student.
- Store the score of each attempt into the Saral database.
- Execute the C, C++, Java, Python, Perl, PHP programs.

Thus, the Saral Programming Interface achieves the requirements of the initial design specification.

The development of the Saral Programming Interface uses MVC (Model-View-Controller) framework. The user interface part of the Saral Interface act as View part which allows Students, Faculty and Administrator to interact with module.

The View part implemented using Port Forwarding which actually acts as communication link between two servers. It creates session and maintains it. One user exit from module it destroys the session. The session is created for compilation and execution of program for each user and to maintain unique identity.

The Model part implements real compilation logic. It receives data, perform operation and send back to Model which then gets displayed on the user interface.

The MVC requires less maintenance and it is easy to understand for others developers to implement further. The development has also considered security

aspect related to online compilation and implemented basic security features such as checking file permission, creating and deleting file structure.

## 9.3 Limitations

There are some limitations in current version of Saral Programming Interface they are:

- Students needs to do all functionality using single class and single file only.
- Administrator do not have option of comparing each compile request of student’s response.
- Students do not have option of compile and execute as different action.
- Current version is tested with the limited resource and single user environment only.

## 9.4 Future work

There are many opportunities to further enhance the Saral Programming Interface. This includes:

- Allowing students to create multiple class files on the same screen.
- Automatic evaluating student's code and grading based on optimisation.

As well the conceptual design underpinning the Saral Interface could be used to create projects online. This allows multiple students to work in single group on a single project from anywhere, anytime without downloading and installing IDE tools to their machine. The student can see changes done by others online and also restore them back if necessary.

# 10. BIBLIOGRAPHY

## Appendix – A– Source Code

1. **EditorBackend.php**

```
<?php
session_start();
$dom = simplexml_load_file("temp\\".$_SESSION['contestcode'].'.xml');

 if (isset($_POST['Run']))
    {

foreach ( $dom->problem as $value) {
echo "<div id='questionInfo'>";
if($value->attributes()->problemCode == $_SESSION['questioncode'])
{
        $link="temp\\".$_SESSION['contestcode'].$_SESSION['questioncode'];
        $lang=$_POST['language'];
        if($lang=='C')
        {
                                $MyFile = fopen("CodeFile.c", "w") or die("Unable to open
file!");
                                fwrite($MyFile,$_POST['Code']);
                                fclose($MyFile);
                                $MyFile = fopen("temp\\InputFile.txt", "w") or die("Unable to
open file!");
                                fwrite($MyFile, $_POST['Args']);
                                fclose($MyFile);
                                if(!empty($_POST['Args'])){

                                 if (system('gcc -o CodeFile CodeFile.c
2>temp\\ErrorFile.txt')!==false)
                                {
```

```
                        exec('CodeFile.exe
1>temp\\OutputFile.txt<temp\\InputFile.txt');

                    }
                    unlink("temp\InputFile.txt");
                    }
                    else
                    {
                        if (system('gcc -o CodeFile CodeFile.c
2>temp\\ErrorFile.txt')!==false)
                        {
                            exec('CodeFile.exe
1>temp\\OutputFile.txt<'.$link.'sampleinput.txt');
                        }
                    }
                    unlink("CodeFile.exe");
                    unlink("CodeFile.c");
                }
                elseif ($lang=="C++")
                {
            $MyFile = fopen("CodeFile.cpp", "w") or die("Unable to open file!");
            fwrite($MyFile,$_POST['Code']);
            fclose($MyFile);
            $MyFile = fopen("temp\\InputFile.txt", "w") or die("Unable to open file!");
            fwrite($MyFile, $_POST['Args']);
            fclose($MyFile);
            if(!empty($_POST['Args'])){
              if (system('g++ -o CodeFile CodeFile.cpp 2>temp\\ErrorFile.txt')!==false)
              {
                exec('CodeFile.exe 1>temp\\OutputFile.txt<temp\\InputFile.txt');
              }
              unlink("temp\\InputFile.txt");
              }
```

```
    else {
        if (system('g++ -o CodeFile CodeFile.cpp 2>temp\\ErrorFile.txt')!==false)
        {
              exec('CodeFile.exe 1>temp\\OutputFile.txt<'.$link.'sampleinput.txt');
        }
    }
    unlink("CodeFile.cpp");
    unlink("CodeFile.exe");
        }
elseif ($lang=="java")
{
$MyFile = fopen("main.java", "w") or die("Unable to open file!");
    fwrite($MyFile,$_POST['Code']);
    fclose($MyFile);
    $MyFile = fopen("temp\\InputFile.txt", "w") or die("Unable to open file!");
    fwrite($MyFile, $_POST['Args']);
    fclose($MyFile);
    $s=filesize('temp\\InputFile.txt');
    if($s){
        if (system('javac main.java 2>temp\\ErrorFile.txt')!==false)
        {
            exec('java main 1>temp\OutputFile.txt<temp\\InputFile.txt');
        }
        unlink("temp\\InputFile.txt");
    }
    else {
        if (system('javac main.java 2>temp\\ErrorFile.txt')!==false)
        {
            exec('java main 1>temp\OutputFile.txt<'.$link.'sampleinput.txt');


        }
    }
        unlink("main.java");
    unlink("main.class");
}
elseif ($lang=="python") {
    $MyFile = fopen("main.py", "w") or die("Unable to open file!");
    fwrite($MyFile,$_POST['Code']);
    fclose($MyFile);
    $MyFile = fopen("temp\\InputFile.txt", "w") or die("Unable to open file!");
    fwrite($MyFile, $_POST['Args']);
    fclose($MyFile);
    $s=filesize('temp\\InputFile.txt');
    if(filesize("ErrorFile.txt")){
    $handle = fopen("temp\\ErrorFile.txt", "r");
            if ($handle) {
                while (($line = fgets($handle)) !== false) {
                    echo strip_tags($line);
                    echo "<br/>";
                }
                fclose($handle
            }
            unlink("ErrorFile.txt");
            }
            if(filesize("temp\\OutputFile.txt")){
            $handle = fopen("temp\\OutputFile.txt", "r");
                if ($handle) {
                    while (($line = fgets($handle)) !== false) {
                        echo "<br/>";
                        echo strip_tags($line);
                    }
                    fclose($handle);


                }
                 unlink("temp\\OutputFile.txt");
            }
```

```
        }
    }
    }
     function filecomparision($a, $b){
        $v = rtrim(file_get_contents($b),"\x0D\x0A");
        $MyFile = fopen($b, "w") or die("Unable to open file!");
        fwrite($MyFile,$v);
        fclose($MyFile);
        if(filesize($a) !== filesize($b))
             return false;
        $ah = fopen($a, 'rb');
        $bh = fopen($b, 'rb');
      $result = true;
        while(!feof($ah))
        {
            if(fread($ah, 8192) != fread($bh, 8192))
            {
            $result = false;
            break;
            }
        }
        fclose($ah);
        fclose($bh);
        return $result;
    }
    if(isset($_POST['Submit']))
    {
        $score=0;
        $dom = simplexml_load_file("temp\\".$_SESSION['contestcode'].'.xml');
        $link="temp\\".$_SESSION['contestcode'].$_SESSION['questioncode'];
        $lang=$_POST['language'];
        foreach ( $dom->problem as $value) {
            if($value->attributes()->problemCode == $_SESSION['questioncode']){
```

```
        foreach ($value->TestCase as $key) {
         $filename=(string)$key->InputFile;
if($lang=='C++')
{
  $MyFile = fopen("CodeFile.cpp", "w") or die("Unable to open file!");
    fwrite($MyFile,$_POST['Code']);
    fclose($MyFile);
    if (system('g++ -o CodeFile CodeFile.cpp 2>temp\\ErrorFile.txt')!==false)
    {
        exec('CodeFile.exe 1>temp\\OutputFile.txt<'.$filename);
    }
    unlink("CodeFile.cpp");
    unlink("CodeFile.exe");
}
if($lang=='C') {
    $MyFile = fopen("CodeFile.c", "w") or die("Unable to open file!");
    fwrite($MyFile,$_POST['Code']);
    fclose($MyFile);
     if (system('gcc -o CodeFile CodeFile.c 2>temp\\ErrorFile.txt')!==false)
    {
        exec('CodeFile.exe 1>temp\\OutputFile.txt<'.$filename);
    }
    unlink("CodeFile.c");
    unlink("CodeFile.exe");
  }
  if ($lang=="java") {
   $MyFile = fopen("main.java", "w") or die("Unable to open file!");
    fwrite($MyFile,$_POST['Code']);
    fclose($MyFile);

     if (system('javac main.java 2>temp\\ErrorFile.txt')!==false)
    {
        exec('java main 1>temp\\OutputFile.txt<'.$filename);
        }
        unlink("main.java");
        unlink("main.class");
   }
       if(filesize('temp\\OutputFile.txt')){
            if(filecomparision(strval($key->OutputFile),'temp\\OutputFile.txt'
                 $score=$score+((int)($key->TestCasePoints));                    }
              unlink('temp\\OutputFile.txt');
       }
   }
   }
   }
   f(filesize("temp\\ErrorFile.txt")){
   $handle = fopen("temp\\ErrorFile.txt", "r");
           if ($handle) {
              while (($line = fgets($handle)) !== false) {
                 echo strip_tags($line);
                 echo "<br/>";
              }
              fclose($handle
           }
           unlink("temp\\ErrorFile.txt");
           }
            $servername = "localhost";
        $username = "root";
        $password = "";
        $connection = mysqli_connect($servername, $username, $password,"editor");
              $result=mysqli_query($connection,"select * from $_SESSION[contestcode]
where username='$_SESSION[userName]' and
programcode='$_SESSION[questioncode]'");
               $res=mysqli_query($connection,"select * from
$_SESSION[contestcode]".'leaderboard'." where
username='$_SESSION[userName]'");
```

```
                $row = mysqli_fetch_array($result,MYSQLI_ASSOC);
                $row1 = mysqli_fetch_array($res,MYSQLI_ASSOC);
                $lscore=$row1['score'];
                $pscore=$row['score'];
                if($pscore<$score){
                    $pscore=$score;
                    $lscore=$lscore-$pscore;
                    $lscore=$lscore+$score;}
                    mysqli_query($connection,"update $_SESSION[contestcode] set
score='$pscore' where username='$_SESSION[userName]' and
programcode='$_SESSION[questioncode]'");
                    mysqli_query($connection,"update
$_SESSION[contestcode]".'leaderboard'." set score='$lscore' where
username='$_SESSION[userName]'");
                    echo "<br/><b>Your score for this problem is </b>".$score;
 }
 ?>
```

## Appendix– B– References

[1] Douglas Kramer, 1996, The Java Platform: A white paper, 2009,
http://java.sun.com/docs/white/platform/javaplatformTOC.doc.html

[2] E.Allen, R.cartwright and B. Stoler, n.d., *DrJava*
http://drjava.sourceforge.net

[3] Jost Bökemeier, n. d., Php/Java Bridge,
http://php-java-bridge.sourceforge.net/pjb/index.php

[4] P. Niemeyer, n.d. BeanShell,
http://www.beansheell.org/

[5] Roedy Green, February, 2006, JavaCompiler: Java Glossary,
http://mindprod.com/jgloss/javacompiler.html#BENEFITS

[6] Sun Microsystems Inc, 2009c, Java SE 6,
http://java.sun.com/javase/6/

[7] Von der Ahé Peter, Leading-Edge Java: the Java Compiler *API*,
http://www.artima.com/lejava/articles/compiler_api.html

[8] Terence Parr, Johannes Luber, The Difference Between Compilers and Interpreters

[9] Heyne, R. (1984). "Basic-Compreter für U880" [BASIC compreter for U880 (Z80)].

[10] AST intermediate representations, Lambda the Ultimate forum

[11] Armin Rigo, Maciej Fijałkowski, Carl Friedrich Bolz. Welcome to PyPy,
http://pypy.org/compat.html

[12] Frank Wierzbicki, Jython NEWS,
https://hg.python.org/jython/file/412a8f9445f7/NEWS

[13] BEggers, billchiles, DinoViehland. IronPython,
http://ironpython.codeplex.com/releases/view/81726

[14] pythonnet / pythonnet.

https://travis-ci.org/pythonnet/pythonnet

[15] Denfromufa, TonyRoberts, pythonnet-480xs.
https://ci.appveyor.com/project/TonyRoberts/pythonnet-480xs

[16] Benjamin. Python 2.7 Release Schedule.

https://www.python.org/dev/peps/pep-0373/#id2