

CONTENTS

S.NO	NAME OF THE TITLE	PAGE NO
1	SYNOPSIS	
2	COMPANY PROFILE	
3	INTRODUCTION	
4	MODULAR DESCRIPTION	
5	SYSTEM REQUIREMENTS 4.1 Software requirements 4.2 Minimum Hardware Requirements 4.3 System Analysis 4.4 System Design	
6	SYSTEM TESTING AND IMPLEMENTATION 5.1 System Testing 5.2 Implementation	
7	ABOUT THE SOFTWARE 6.1 Software Architecture 6.2 HTML 6.3 Java Servlets 6.4 JDBC 6.5 ODBC 6.6 MS Access	
8	DATA FLOW DIAGRAMS	
9	DATABASE DESIGN	
10	SOURCE CODE	
11	SCREEN LAYOUTS	
12	CONCLUSION	
13	BIBLIOGRAPHY	

COMPANY PROFILE

HIDUSTAN SOFTWARE LIMITED

Hindustan software limited established in 1995, has been actively involved in development of customized software for various clients since its inception and also has its presence strongly felt in the areas of Training & consulting. Market Leadership through customer satisfaction, a commitment to excellence & high growth rate, these have characterized HSL in its rapid climb in Information Technology Industry.

DEVELOPMENT

Our software development team strives to develop innovative software that meets customer needs. A team of developers with exclusive educational background and excellent computing skills exert their maximum efforts and endeavor to satisfy the specialized requirement of our esteemed clients. Our expertise and vast exposure qualifies us in developing software's in Information systems, Web Based Applications, Internet Solution, Web page Design and Hosting. Our team consists of expertise in Java, VC++, VB, Oracle and all Internet Application Development Tools.

TRAINING

Hindustan has trained more than 1000 professionals so far in the area of JAVA and VC++. All the students have launched in a very good career path both in India & abroad. The curriculum matches latest trends and make the students to compete in the IT industry.

SKILLSET AVAILABLE

C, C++, VC++, JAVA, ASP, XML, JSP, EJB-Web Logic, VISUAL BASIC -MS-ACCESS, RMI, COM, CORBA, SCRIPTS, EMBEDDED SYSTEM, ETC.,

CONSULTING

The training division gives us an opportunity to meet variety of candidates with various skill sets. Our database consists of details regarding more than 1000 professionals. So we are fit to assist all sort of IT Human resource consultancies and IT companies in India and abroad.

Excellence through "TEAM WORK" The Philosophy of Total Quality Management is inculcated in every HSL employee through intensive training programs. Our investment on a highly skilled and motivated manpower constitutes towards establishing HSL as a truly quality conscious company, continually striving to bring the finest solutions to the discerning customers.

SYNOPSIS

Human Resource Management System is aimed to integrate the activities of Human Resource Department of Hindustan Software Limited (HSL).

The Human Resource Management System maintains the following core activities and core processes of HSL.

1. Personal Information Management
2. Personal Training Management
3. Recruitment Process Management
4. Project Management

The information collected through the above activities will be maintained in a centralized server and could be accessed through the Internet. The company has decided to create a corporate intranetwork to connect all offices and their network would also be utilized in case of implementation. The information collected through this management and process related activities are maintained as follows.

Personal Information Management

This Management process is to maintain the details of employees who are working at HSL as well as employees who are working in various client concerns. The information will be maintained by HR Managers of all HSL and its units. Employee details that would be sent abroad and other concerns other than clients of HSL would also be managed.

Personal Training Management

Employee will be given regular training on need in the latest advanced areas. The training management maintains a folio on the various topics on which the training is provided. It is also maintains the details of the employee who underwent/is undergoing /will be undergone training. The training has been classified into three categories.

- In-house training

- Offshore training

The training management also maintains information about the guest lectures as well as lecturers.

Recruitment Process Management

It creates a folio relating to Recruitment Process. There are three different methods to recruit employees. They are

- Through direct recruitment

- Through campus interviews and

- Absorbing project trainees

Direct Recruitment

Direct Recruitment is done on the net. Resumes of candidates are received as either directly or by mail/e-mail/fax/courier. If they found eligible for interview they would be mailed indicating a user name, password and date of interview. The candidate has to log on into the HSL server and he will be given a set of questions (mostly of objective type). He has to answer those questions and the HR Manager will process his papers. The process may include further interviews, direct appointment.

Campus Interviews

Employees are appointed by conducting interviews at college campus. The profile of the students who are interviewed is maintained. It also maintains the details of the college. The selected candidates will first be given training in their respective areas and then they are put up in any one of its developing unit.

Absorbing Project Trainees

The Project Trainees could also be hired by the HSL concern, if their project performance is well.

Project Management

The details of all projects done by HSL are maintained. The projects are classified into two categories.

- ☐ Offshore Projects
- ☐ Onsite Projects

In case of onsite project employees will be sent to the client concern. Offshore projects are developed in any one of the developing units. All project-related information is maintained.

It also maintains the details of the clients who have consulted for their project work. It includes the client organization and nature of consultancy.

The information belonging to the students who request to work as project trainees are also maintained. That includes the profile of the student as well as the project to which he is sent. It also maintains the student's college information.

MODULAR DESCRIPTION

The Human Resource Management System comprises of a login screen and 2 modules

There are basically 2 modules

- ☐ User module
- ☐ Administration module

User module exits for

- ☐ Recruitment process
- ☐ Company details

Recruitment process

- ☐ Direct interview
- ☐ Campus interview
- ☐ Absorbing trainee

Administration module exits for

- ☐ Project maintenance
- ☐ Company details
- ☐ Employee details
- ☐ Recruitment process

Reports generated for

- ☐ Employee details
- ☐ Project training
- ☐ Recruitment report
- ☐ Online exam report
- ☐ Company details

SYSTEM REQUIREMENTS

Software Requirements

OPERATING SYSTEM	: WINDOWS'98
FRONT END	: JAVA SERVLET, HTML 4.0
BACK END	: MS ACCESS
SERVER	: JAVA WEB SERVER 2.0

Hardware Requirements

CPU TYPE	: Pentium III
MEMORY	: 128 MB RAM
HARD DISK	: 20 GB HDD
KEY BOARD	: 110 KEYS

Drive Summary

FLOPPY DRIVE	: 1.44 MB FDD
CDROM	: 52X
MONITER	: 15"COLOR MONITOR

SYSTEM ANALYSIS

Introduction to System Analysis

System analysis is a process of gathering interpreting facts, diagnosing problems, and using facts to improve the system. The objective of the system analysis is to understand the important facts of current system by studying it in detail. To accomplish this objective the following have to be done.

- Learn the details of the system as well as procedures currently in practice.
- Develop insight into future demands of the organization on its growth; hike in Competition, evolving new financial structures, introduction of new technology, and changes in the customer needs.
- Documentation details of the current system for discussion and review by others.
- Evaluate effectiveness and efficiency of the current system and procedure taking into account the impact of anticipating future demands.
- Recommend any revisions and enhancements to the current system, indicating how they are justified. If appropriate, an entire new system may be purposed.
- Document the new system features at a level of details that allows others to understand its components and manage the new system developed.

SYSTEM DESIGN

The design of a system produces the details that state how a system meet the requirements identified during system analysis. System specialists often refer to this stage as logical design, in contrast to the process of developing program software, which is referred to as physical design.

Data Flow Diagrams have been used in the design of the system. Data Flow Diagram is a graphical tool used to describe and analyze the movement of data. The transformation of data from input to output, through processes may be described logically using these Data Flow Diagrams.

The DFD shown to the user must represent only the major functions being performed by the system. This is called Top Level DFD. If this process is complex enough, it can be broken further into different levels. This process can be continued till the process is simple. This is called the leveling of DFD's.

SYSTEM TESTING

Theoretically, a new designed system should have all the pieces in working order, but in reality, each piece works independently. Now is the time to put all pieces into one system and test it to determine whether it meets the user's requirements. The purpose of the system is to consider all the likely variations to which it will be subjected and then push the system to its limits. It is tedious but necessary step in system development. One needs to be familiar with the following basic terms.

- **UNIT TESTING:** Unit Testing is testing changes made in an existing or a new program.
- **SEQUENTIAL OR SERIES TESTING:** Sequential or Series Testing is checking the logic of one or more programs in the candidate system, where the output of one program will affect the processing done by another program.
- **SYSTEM TESTING:** System Testing is executing a program to check logic changes made in it and with the intention of finding errors making the program fail.
- **ACCEPTANCE TESTING:** Acceptance Testing is running the system with live data by the actual user of the system.

Testing is vital to the success of the system. System testing makes a logical assumption that if all the parts of the system are correct, the goal will be successfully achieved. Inadequate testing or no-testing leads to errors that may not appear until months later. Another reason for system testing is its utility as a user-oriented vehicle before implementation. The best program and the user have communication barriers due to different backgrounds. The system tester (designer, programmer, or user) who has developed some computer mastery can bridge this barrier.

(i) Unit Testing:

This focuses on the smallest unit of software design. The module using the details design description as a guide; important control paths are tested to uncover errors within the boundary of the module.

Unit test consideration:

The module interface is tested to ensure that information properly flows into and out of the program unit under test. The local data structures are examined to ensure that the data stored temporarily maintains its integrity during all steps in an algorithm execution. Boundary conditions are tested to ensure that the module operates properly at boundaries established to limit or restrict processing. The test of data flow across a module interface is required before any other test. If data do not enter and exit properly, all other tests are moot.

Unit Procedure:

- Unit test is normally considered adjunct to the coding style.
- After source level code has been developed, reviewed and verified for correct syntax, unit test case design begins. Each test case should be coupled with a set of expected results.
- Normally, a driver is a "main program" that accepts test case data, passes such data to the module to be tested and prints the relevant results. Stubs serve to replace modules that are subroutines called by the module to be tested. A Stub or 'dummy stub program' uses the subroutine module's interface to do minimal data manipulation and returns.

Unit testing is simplified when a module with high cohesion is designed. When a module addresses only one function, the number of test cases is reduced and errors can be more easily predicted and uncovered.

(ii) Integration Testing:

Integration is a systematic technique for constructing the program structure, while at the same time conducting tests to uncover errors associated with interfacing. The objective is to make unit-tested modules and build a program structure that has been dictated by design. Incremental integration is the program that is the program that is constructed and tested in small segments where errors are easier to isolate and correct.

Top down Integration:

Top-down integration is an incremental approach to the construction of program structure. Modules are integrated by moving downward through the control hierarchy, beginning with the main control module. Module subroutine to the main control module is incorporated into the structure either in a depth-first manner is engaged for this system. Breadth-first incorporates all modules directly subroutine at each level, moving across the structure horizontally.

The integration process is performed in a series of five steps:

- a. The main control module is used as a test driver and stubs are substituted for all modules directly subroutine to the main control module.
- b. Depending on the integration approach selected (depth or breadth first) subroutine stub are replaced one at a time with actual modules.
- c. Tests are conducted as each module is integrated.
- d. On the completion of each set of test, another stub is replaced with the real module.
- e. Registration testing is conducted to ensure that new errors have not been introduced.

(iii) Validation testing:

At the end of integration testing, the system is completely assembled as a package with interfacing errors corrected after which a final series of software tests namely validation testing begins. Validation succeeds when the software functions in a manner that can be reasonably expected by the user.

Criteria:

Software validation is achieved through black box tests that demonstrates conformity with requirements.

(iv) System Testing:

System testing is actually a series of different tests whose primary purpose is to fully exercise the computer-based system. Although each test has a different purpose, all work should verify that all system elements have been properly integrated and perform allocated functions. Being the most important test, the performance test is covered briefly below:

- a. **Performance Testing:** For real-time systems, software that provides required function but does not conform to performance requirement is unacceptable. Performance testing is designed to test the run-time performance of software within the context of an integrated system. Performance testing occurs throughout all steps in the testing process. Even at the unit level, the performance of an individual module may be accessed as tests are conducted. However, it is not until all system elements are fully integrated that the true performance of a system can be ascertained.

Debugging:

Debugging is not testing not occurs as a consequence of testing, that is when a test case uncovers an error, debugging is the process that results in the removal of the error.

Normally three categories for debugging approaches are proposed:

- a. Brute force
- b. Back-tracking**
- c. Cause-elimination

a. Brute force:

This is the most common and least efficient method for isolating the cost of a software error. Brute-force debugging method is usually applied when all else fails. Using a "let the computer finding the error" philosophy, memory-dumps are taken, runtime traces are invoked and the program is loaded with WRITE (in this case message box) statements. In the information that is produced, a clue is found leading to the cause of the error.

b. Back-tracking:

This is fairly common debugging approach that can be used successfully in small programs. Beginning at the site where a symptom has been uncovered, the source code is traced backwards (manually) until the site of the cost is found.

c. Cause-Elimination:

This approach is manifested by induction/deduction and introduces the concept of 'binary partition'. A 'cause hypothesis' is devised and the error related data are used to prove or disprove the hypothesis. Alternatively, a list of all possible causes is developed, and tests are conducted to eliminate each. If initial tests indicate that a particular cause hypothesis shows promise, that data are refined in an attempt to isolate the path.

Each of the debugging approaches can be supplemented with debugging tools. A wide variety of debugging compilers, dynamic debugging aids (tracers), automatic test case generators, memory dumps and cross-reference maps can be applied. However, tools are not a substitute for careful evaluation, based on a complete software design document and clear source code.

SYSTEM IMPLEMENTATION

After proper testing and validation, the question arises whether the system can be implemented or not. Implementation includes all those activities that take place to convert from the old system to the new. The new system may be totally new, replacing an existing module or automated system, or it may be major modification to an existing system. In either case proper implementation is essential to provide a reliable system to meet organization requirements.

All planing has now, be completed and the transformation to a fully operational system can commence. The first job will be writing, debugging documenting of all computer programs and their integration into a total system. The master and transaction files are decided, and this general processing of the system is established. Programming is complete when the programs conformed to the detailed specification.

When the system is ready for implementation, emphasis switches to communicate with the finance department staff. Open discussion with the staff is important form the beginning of the project. Staff can be expected to the concerned about the effect of the automation on their jobs and the fear of redundancy or loss of status must be allayed immediately. During the implementation phase it is important that all staff concerned be apprised of the objectives of overall operation of the system. They will need shinning on how computerization will change their duties and need to understand how their role relates to the system as a whole. An organization-training program is advisable; this can include demonstrations, newsletters, seminars etc.

The department should allocate a member of staff, who understands the system and the equipment, and should be made responsible for the smooth operation of the system. An administrator should coordinate the users to the system.

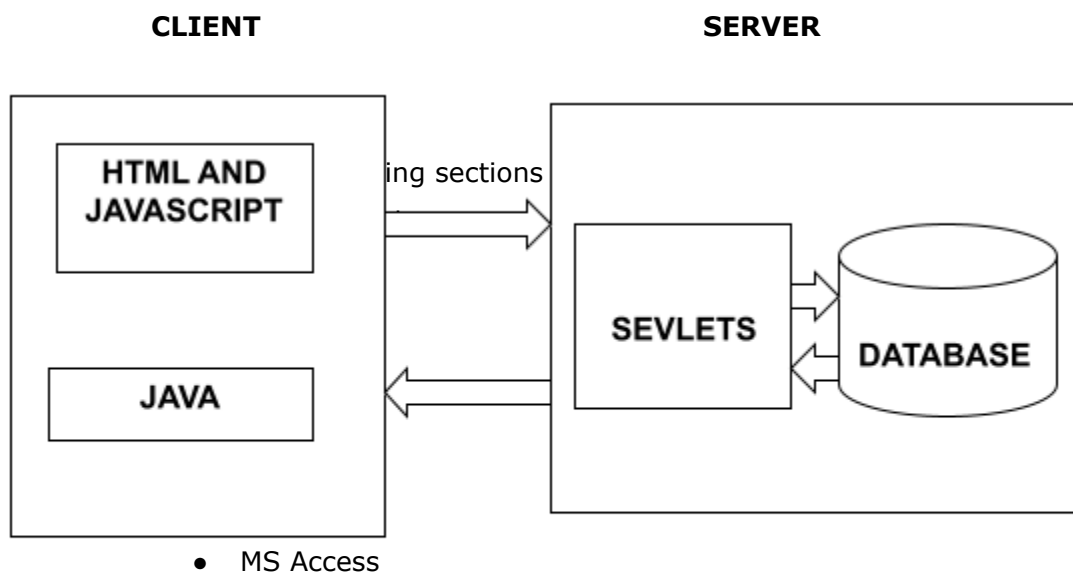
Users should be informed about new aspects of the system that will, affect them. The features of the system explained with the adequate documentation. New services such as security, on-line application from the back-ups must be advertise on the staff when the time is ripe.

Existing documents such as employee loan details should be entered into the new system. Since these files are very large, conversion of these may continue long after the system based on current files has been implemented. Hence we need to assign responsibility for each activity.

The system may come into full operation via number of possible routes. Complete change over at one point time is conceptually the most tidy. But this approach requires careful planning and coordination, particularly during the changeover. A phased approach, possible implementing the system of the section relating to one operation or procedure first and processing to more novel or complex subsystems in the fullness of time. These likely to be less traumatic. A phased approach gives the staff time to adjust to the new system. But depends on being able to split the system, without reliance on it. Thus approach is sensible when the consequences of failure are disastrous, but will require extra staff time. The fourth angle, is pilot operation permits any problems to be tackled on a smaller scale operation. Pilot operation generally means the implementation of the complete system, but at one location or branch only.

SOFTWARE ARCHITECTURE

The following diagram illustrates the links between various components involved in this system:



JAVA

WHAT IS JAVA?

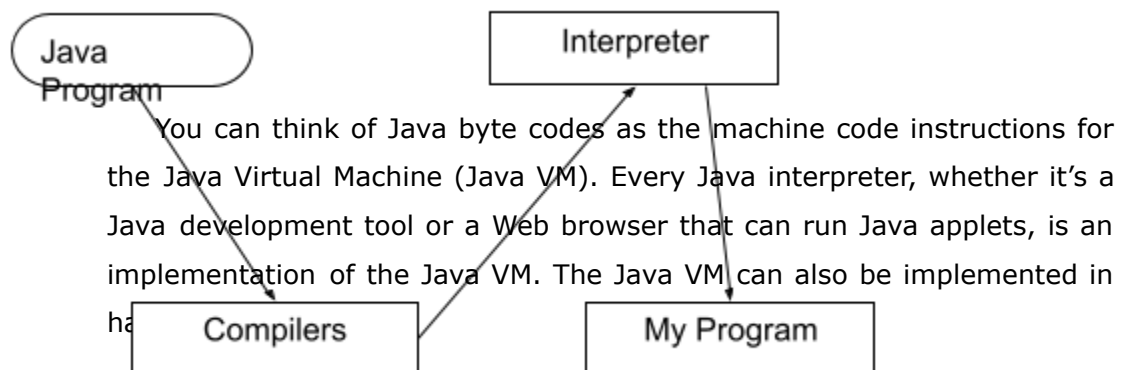
Java ha two things: a programming language and a platform.

Java is a high-level programming language that is all of the following

Simple	Architecture-neutral
Object-oriented	Portable
Distributed	High-performance
Interpreted	multithreaded
Robust	Dynamic
Secure	

Java is also unusual in that each Java program is both compiled and interpreted. With a compiler, you translate a Java program into an intermediate language called Java bytecodes. The platform-independent code instruction is passed and run on the computer.

Compilation happens just once; interpretation occurs each time the program is executed. The figure illustrates how this works.



Java byte codes help make "write once, run anywhere" possible. You can compile your Java program into byte codes on my platform that has a Java compiler. The byte codes can then be run any implementation of the Java VM. For example, the same Java program can run Windows NT, Solaris, and Macintosh.

JAVA PLATFORM

A platform is the hardware or software environment in which a program runs. The Java platform differs from most other platforms in that it's a software-only platform that runs on top of other, hardware-based platforms. Most other platforms are described as a combination of hardware and operating system.

The Java platform has two components:

- 1) The Java Virtual Machine (Java VM)
- 2) The Java Application Programming Interface (Java API)

You have already been introduced to the Java VM. It's the base for the Java platform and is ported onto various hardware-based platforms.

The Java API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets.

The Java API is grouped into libraries (packages) of related components. The next sections, *What Can Java Do?* highlights each area of functionality provided by the packages in the Java API.

The following figure depicts a Java program, such as an application or applet, that's running on the Java platform. A special kind of application known as a server serves and supports clients on a network. Examples of the servers include Web Servers, proxy servers, and mail servers, print servers, and boot servers. Another specialized program is a Servlet. Servlets are similar to applets in that they are runtime extensions of the application. Instead of working in

browsers, though, Servlets run with in Java Web Servers, configuring of tailoring the server.

How does the Java API support all of these kinds of programs? With packages of software components, that provides a wide range of functionality. The API is the API included in every full implementation of the platform.

The core API gives you the following features:

- 1) The Essentials: Objects, Strings, threads, numbers, input and output, datastructures, system properties, date and time, and so on.
- 2) Applets: The set of conventions used by Java applets.
- 3) Networking URL's TCP and UDP sockets and IP addresses.
- 4) Internationalization: Help for writing programs that can be localized for users.

Worldwide programs can automatically adept to specific locates and be displayed in the appropriate language.

JAVA PROGRAM

- Java API
- Java Virtual Machine

- Java Program
- Hard Ware

API and Virtual Machine insulates the Java program from hardware dependencies. As a platform-independent environment, Java can be a bit slower than native code. However, smart compilers, well-tuned interpreters, and Just-in-time-byte-code compilers can bring Java's performance close to the native code without threatening portability.

WHAT CAN JAVA DO?

However, Java is not just for writing cut, entertaining applets for the World Wide Web (WWW). Java is a general purpose, high-level programming language and a powerful software platform. Using the fineries Java API, you can write many types of programs.

The most common types of program are probably applets and application, where a Java application is a standalone program that runs directly on the Java platform.

Security:

Both low-level and high-level, including electronic signatures, public/private key management, accesses control, and certificate.

JavaScript OVERVIEW

INTRODUCTION

JavaScript is an interpreted, object-based scripting language. Although it has fewer capabilities than full-fledged object-oriented languages like C++ and Java, JavaScript is more than sufficiently powerful for its intended purposes. JavaScript is not a cut-down version of any other language; it is not a simplification of anything. It is, however, limited. You cannot write standalone applications in it, and it has little capability for reading or writing files. Moreover, JavaScript script can run only in the presence of an interpreter, in either a web server or a web browser.

JavaScript is a loosely typed language. That means you do not have to declare those data types of variables explicitly.

It is a scripting language used to develop Internet applications mostly for client-side validations. It has the features of windows, frames, buttons, forms, images, list boxes, applets and so on.

MAIN FEATURES:

Most of the browsers support JavaScript. It has all the functionality's of a high-level language. It supports OOPS.

JavaScript IN VB SCRIPT:

Microsoft Visual Basic Scripting Edition, the newest member of the Visual Basic family of programming languages, brings active scripting to a wide variety of environments, including web client scripting to a wide variety of environments, including web client scripting in Microsoft Internet Explorer and Web server scripting in Microsoft Internet Information Server.

VBScript talks with host applications using ActiveX scripting. With ActiveX, scripting, browsers and host applications do not require special integration code for each scripting component. ActiveX scripting enables a host to compile scripts, obtain and call entry points, and manage the namespace available to the developer.

HYPERTEXT MARKUP LANGUAGE (HTML)

INTRODUCTION

HTML is a major language of the Internet's World Wide Web. Web sites and web pages are written in HTML. HTML files are plain-text files so they can be edited on any type of Computer IBM, Mac, UNIX, Intel etc. The World Wide Web is a collection of linked documents or pages on millions of computers spread over the entire internet. HTML which defines their appearance and layout and more importantly creates the links to other documents.

A set of instructions embedded in a document is called MarkUp Language. These instructions describe what the document text means and how it should look in a display. The language also tells you how to make a document with other document on your local system, the World Wide Web and other Internet resources such as FTP.

Brief History of HTML:

HTML was originally developed by time Berners lee which at CERN, and popularized by the mosaic browser developed at NCASA. During of the course of the 1990's it has blossomed with the explosive growth of the Web. During this time, HTML has been extended in a number of ways. The Web depends on Web

page authors and vendors sharing the same conventions for HTML. This has motivated the joint work on specifications for HTML. HTML 2.0 was developed under the aegis of the Internet Engineering Task Force (IETF) to codify common practice in late 1994. HTML has been developed with the version that all manners of devices should be able to use information on the Web. Computers with high or low and bandwidth.

Document Structure Elements:

HTML documents are composed of four parts:

- A HTML document begins with a line declaring which version of HTML is being used to create the document.
- A HTML document that describes the documents as a HTML document.
- A declarative header section which is enclosed in the <HEAD> element
- The main body of the document that contains the actual document content. The body can be contained within either the <BODY> elements.

The "head" section is opened and closed by <HEAD> and </HEAD> tags. Information about the document such as the title, indexing information and ownership. The "body" section is opened and closed by <BODY> and </BODY> tags. The text and images of the document itself to be displayed by the Web browser.

Model HTML document:


```

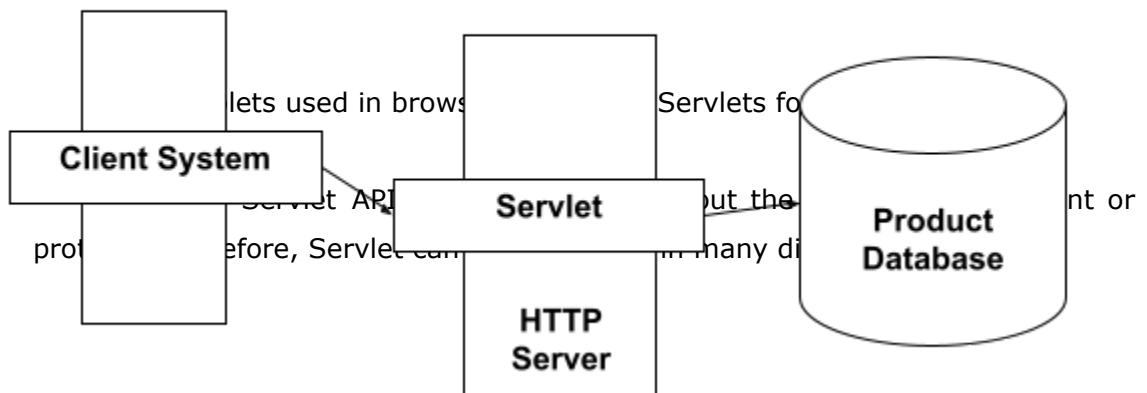
<HTML>
<HEAD>
<TITLE>
    <SAMPLE HTML Document>
</TITLE>
</HEAD>
<BODY>.....</BODY>
</HTML>

```

SERVLETS OVERVIEW

INTRODUCTION

Servlets extend the request-response-oriented servers, such as Java-enabled Web servers. For example, a Servlet can retrieve data from an HTML form and applying the business logic used to update Product database.



Uses of Other Servlets:

There are many applications for Servlets:

- A Servlet can handle multiple requests concurrently, it allows Servlets to support systems and concurrently requests.
- Servlets can be used to balance load among several servers that mirror the same content, and to partition a single logical service over several servers, thus Servlets can forward requests.

Web Servers:

A Web server receives the request it then springs into action. Depending on the type of request, the webserver might look for a web page or it might execute program on the server, usually, as discussed earlier, CGI script or an advanced server-side program.

The Servlet API:

The two packages contains the code to build the Servlets: `javax.servlet` and `javax.servlet.http`

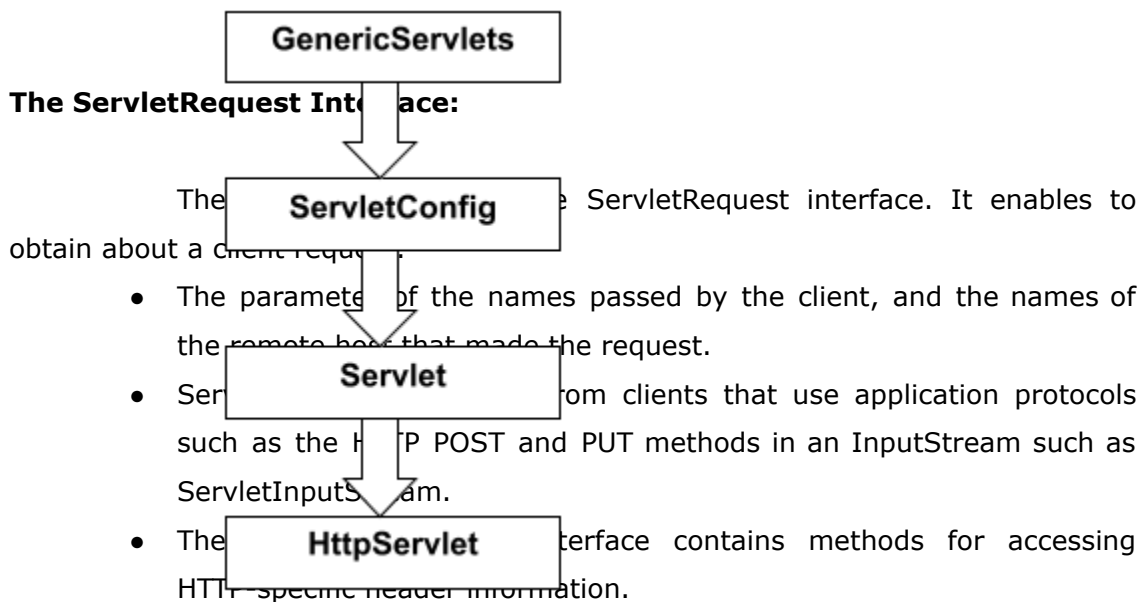
The `javax.servlet` package:

The `javax.servlet` package contains a number of interfaces and classes that establish the framework in which Servlets operate. The `ServletRequest` and `ServletResponse` are also very important.

- The `ServletRequest` interface is used to read data from a client request.
- The `ServletResponse` interface is used to write data to a client response.

The Servlet Interface:

All Servlets must implement the Servlet interface. Generic Servlet must implements the Servlet and Servletconfig interface. HttpServlet extends GenericServlet. It is commonly used to Servlets that receives and process HTTP requests.



The ServletResponse Interface:

The ServletResponse interface is implemented by the server. It enables to obtain about a client response:

- It allows the Servlet to set the content length.
- Servlet can send the reply data which writer through n output stream such as ServletOutputStream.
- The HttpServletResponse interface contains allow the Servlet to manipulate HTTP-specific header information.

Hypertext Transfer Protocol (HTTP):

The Hypertext Transfer Protocol (HTTP) is used as an application-level protocol for distributed, collaborative, hypermedia information systems. A feature of HTTP is the typing and negotiation of data representation, allowing systems to be built independently of the data being transferred. The HTTP protocol is a request/response protocol.

The javax.servlet.http Package:

The javax.servlet.http package contains several interfaces and classes that are commonly used by Servlet developers.

Http Request and Responses:

The Http Requests and Responses contains in two arguments:

- An HttpServletRequest is used to enable Servlets to read data from an HTTP request.
- An HttpServletResponse is used to enable Servlets to write data to an HTTP response.

HttpServletRequest Object:

An HttpServletRequest Object provides to access the data from the client:

- The getParameter method returns the value of a named parameter and similarly the getParameterValues method returns an array of values for the named parameter.
- The getReader method returns the BufferedReader to use to read the data.
- The getInputStream method returns the ServletInputStream to use to read the data.

HttpServletResponse Objects:

An HttpServletResponse object provides the HTTP method. To access the data from the user:

- The getWriter method returns a Writer.
- The getOutputStream method returns the ServletOutputStream.

HTTP Header Data:

The HTTP header data means to access Writer or OutputStream. For example, the GET and POST for HTTP requests to which the service method includes:

- The handling for GET requests returns to doGet, Conditional GET and HEAD requests.
- The handling for POST requests returns to doPost.
- The handling for PUT requests returns to doPut.
- The handling for DELETE requests returns to doDelete.

Servlets Features:

Servlets are efficient when compared to any other server-side programs. They allow persistence of data to be maintained. More important is the fact they are portable, robust security features.

Servlets Better than CGI:

Servlets offer advantages over CGI in the areas of performance, portability and security. Each of these advantages will be discussed in turn:

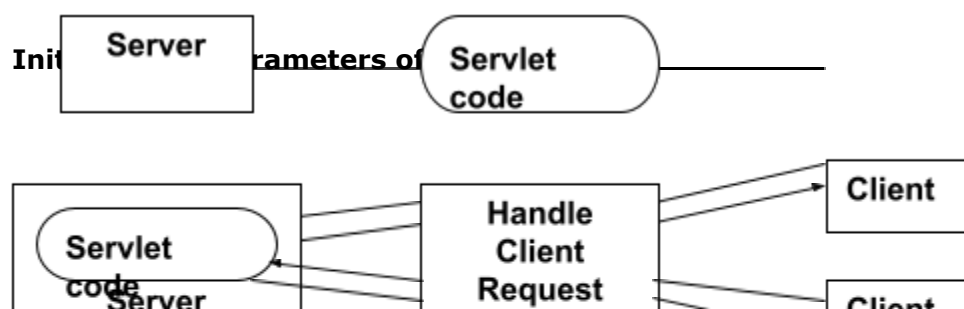
1. Performance is perhaps the most visible difference between Servlets and CGI. Since most Servlets run in the same process space as the server and are loaded only once, they are able to respond much more quickly and efficiently to client requests. In contrast, CGI must create a new process to service each new request. The overhead involved with creating a new process incurs a significant performance penalty. Unlike Servlets, CGI cannot share a single database connection across multiple requests.
2. Portability is another strong advantage for Servlets. Unlike many CGI applications, Servlets can be run on different servers and platforms without modifications. This characteristic can be extremely important when building enterprise-wide distributed applications.
3. Servlets are much more secure than CGI. Though CGI scripts can be written in Java, they are often written in more error-prone language such as C, since C programs can inadvertently or maliciously access invalid memory locations, CGI programs are less secure. Of course, these restrictions can be lifted according to the security policies set by the Java Security Manager.

Life Cycle of Servlet:

Each Servlet has the same life cycle:

- Server loads and initializes the servlets calls the `init ()` method.
- The server handles zero or more client requests calls the `service ()` method.
- The server removes the servlets calls the `destroy ()` method.

(Some servers do this step only when they shutdown **Destroy ()**)



Initialize the Servlet before client requests are handled and before the Servlet is destroyed. The Servlet runs the init method. The Servlet calls the init method when the server loads the Servlet, and similarly will not call the init method again unless the server is reloading the Servlet. After initialization, the Servlet is able to handle client requests. Initialization can be accessed in two ways:

- The init method declared by the Servlet interface receives ServletConfig object as its argument. This object provides methods that enable you to read the initialization parameters.
- The getServletConfig method declared by the Servlet interface returns a ServletConfig object.

Destroying Servlets:

Destroys the Servlet runs the servlet's destroy method. The method is run once; similarly the server will not run the destroy method, when the server calls the destroy method, another thread might be running a service request. Servlets run until the server destroys them.

Communication of Servlets:

To communicate Servlets sometimes need to access network resources. HTML pages, objects shared among Servlets at the same server and other Servlets:

- The Request Dispatcher Object with other Server Resources (JSDK 2.1) such as other Servlets with HTML pages and so on.
- The resource is an object in the JAVA programming language such as Sharing Resources among Servlets (JSDK2.1).
- The other servlet's Servlet object and calling its public such as Calling Servlet from Servlets (JSDK2.0).

Utilities For Running Servlets:

The JSDK2.0 has a utility called Servletrunner. If you can test it with the utility included in the JSDK, while JSDK2.1 comes with a small utility server.

Properties of Servlets:

The Servlet requires initialization parameters, if you set this data before starting the JSDK process that runs your Servlet.

JAVA DATABASE CONNECTIVITY (JDBC)

JDBC AND ODBC IN JAVA:

Most popular and widely accepted database connectivity called Open Database Connectivity (ODBC) is used to access the relational databases. It offers the ability to connect to almost all the databases on almost all platforms. Java applications can also use this ODBC to communicate with a database. Then we need JDBC why? There are several reasons:

- ODBC API was completely written in C language and it makes an extensive use of pointers. Calls from Java to native C code have a number of drawbacks in the security, implementation, robustness and automatic portability of applications.
- ODBC is hard to learn. It mixes simple and advanced features together, and it has complex options even for simple queries.
- ODBC drivers must be installed on client's machine.

Architecture of JDBC:

JDBC Architecture contains three layers:

JDBC Application

JDBC Drivers

1. ~~Application Layer: Java program wants~~ to get a connection to a database. It needs ~~the database to display on the screen or to modify the existing data or to insert the data into the table.~~
JDBC Drivers
2. Driver Manager: The layer is the backbone of the JDBC architecture. When it receives a connection-request form.
3. The JDBC Application Layer: It tries to find the appropriate driver by iterating through all the available drivers, which are currently registered with Device Manager. After finding out the right driver, it connects the application to appropriate database.
4. JDBC Driver layers: This layer accepts the SQL calls from the application and converts them into native calls to the database and vice-versa. A JDBC Driver is responsible for ensuring that an application has consistent and uniform m access to any database.

When a request received by the application, the JDBC driver passes the request to the ODBC driver, the ODBC driver communicates with the database, sends the request, and gets the results. The results will be passed to the JDBC driver and in turn to the application. So, the JDBC driver has no knowledge about the actual database, it knows how to pass the application request o the ODBC and get the results from the ODBC.

The JDBC and ODBC interact with each other, how? The reason is both the JDBC API and ODBC are built on an interface called "Call Level Interface" (CLI). Because of this reason, the JDBC driver translates the request to an ODBC call. The ODBC then converts the request again and presents it to the database. The results of the request are then fed back through the same channel in reverse.

Structured Query Language (SQL):

SQL (Pronounced Sequel) is the programming language that defines and manipulates the database. SQL databases are relational databases; this means simply the data is store in a set of simple relations. A database can have one or more table. You can define and manipulate data in a table with SQL commands. You use the data definition language (DDL) commands to creating and altering databases and tables.

You can update, delete or retrieve data in a table with data manipulation commands (DML). DML commands include commands to alter and fetch data.

The most common SQL commands include commands is the SELECT command, which allows you to retrieve data from the database.

In addition to SQL commands, the oracle server has a procedural language called PL/SQL. PL/SQL enables the programmer to program SQL statement. It allows you to control the flow of a SQL program, to use variables, and to write error-handling procedures

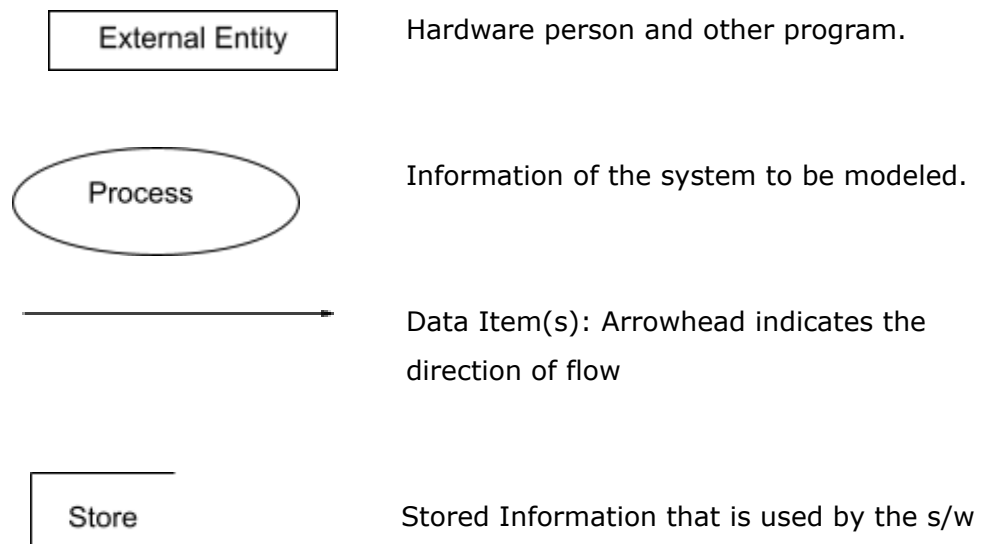
Data Flow Diagrams

Definition:

A data flow diagram is a graphical technique that depicts information flow and the transforms that applied as data move from input to output. The Data flow diagram used to represent a system or software at any level of abstraction. In fact DFDs may be portioned into levels.

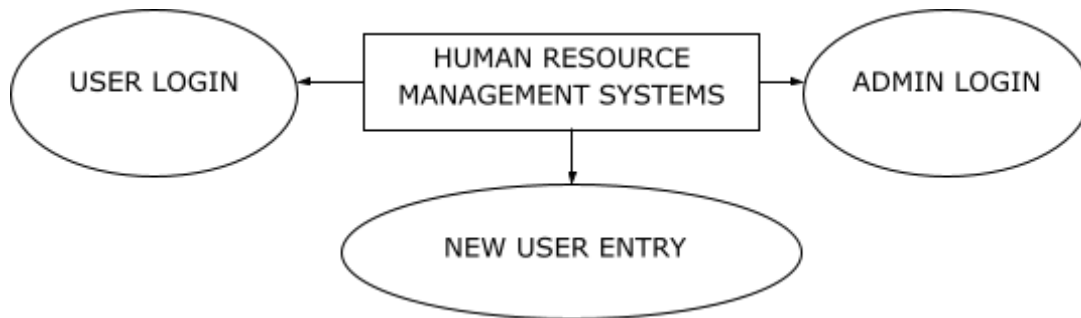
A level of DFD, also called a context model, represents the entire software elements as a single bubble with input and output by arrow. A level of DFD is portioned into several bubbles with inter connecting arrows. Each of the process represented at level one is sub function of the over all depicted in the context model.

The DFD Notations:



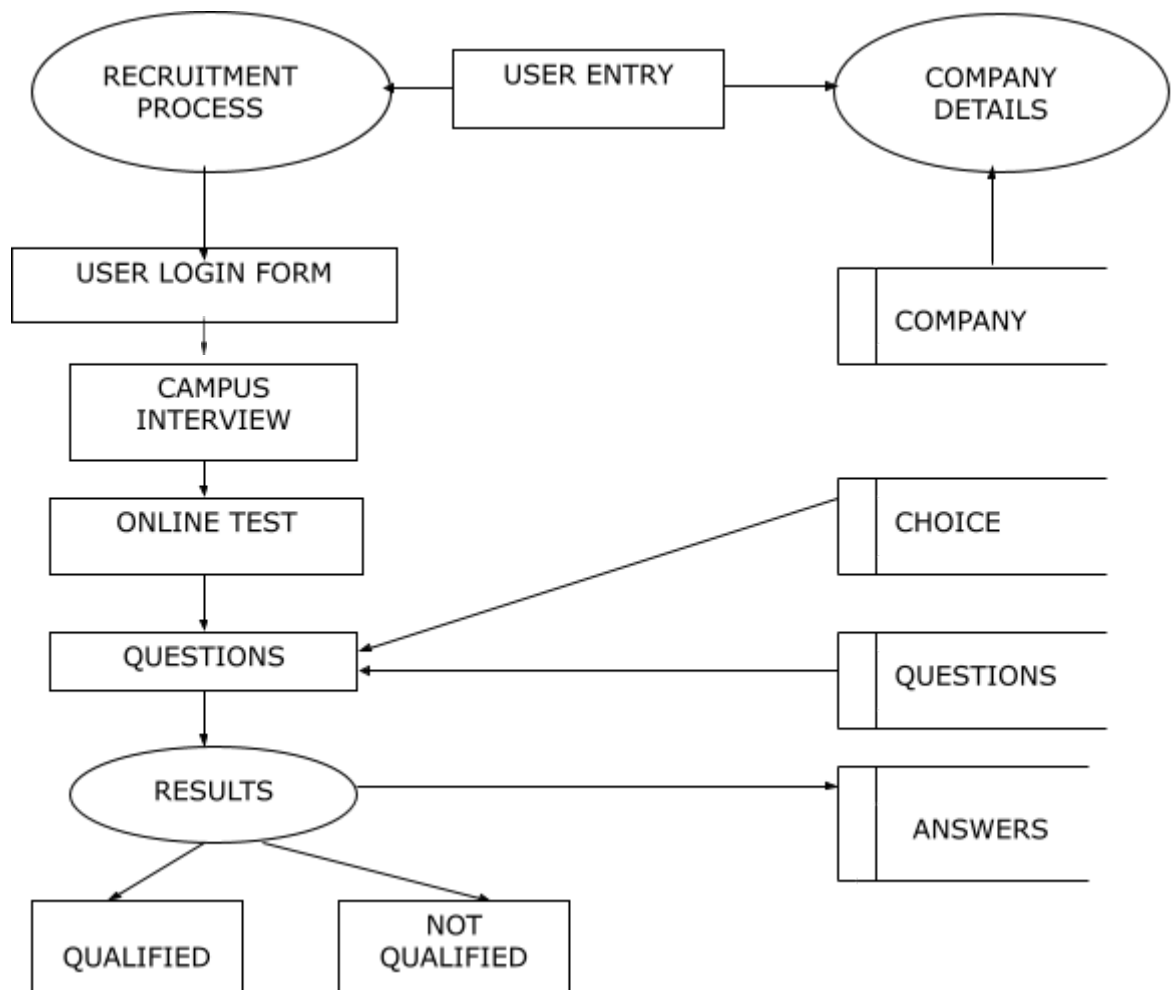
Level 0:

DATA FLOW DIAGRAM FOR OVERALL SYSTEM



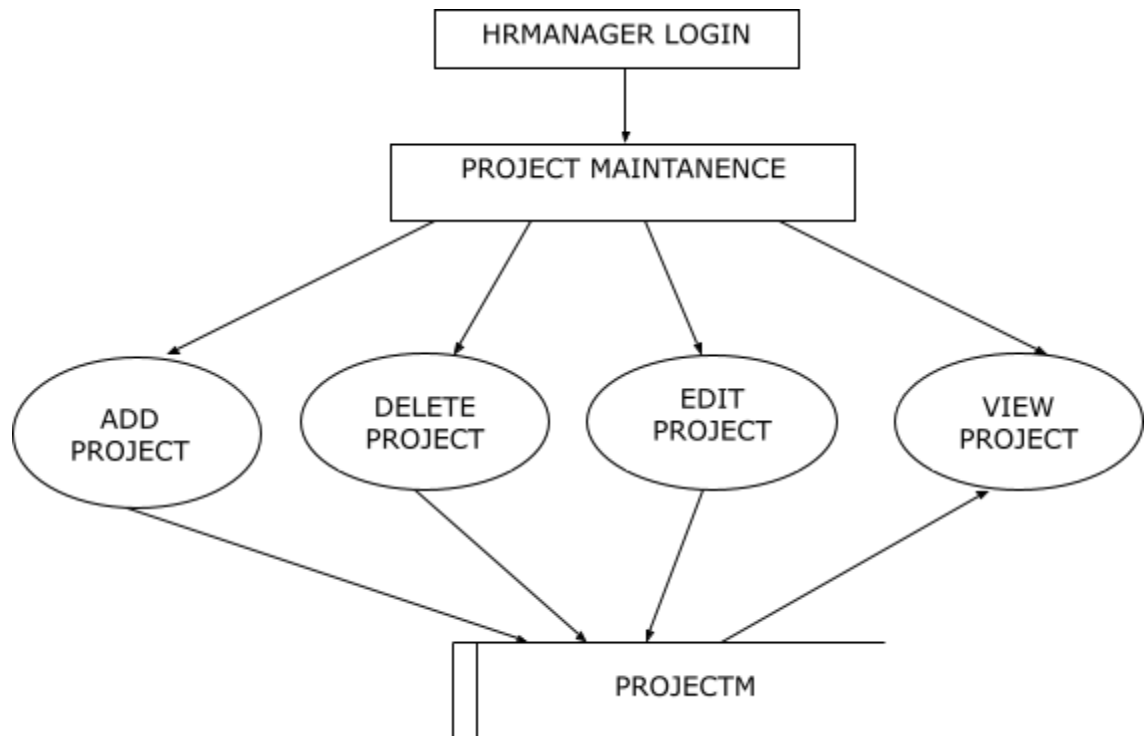
Level 1:

USERS MODULE FLOW PROCESS

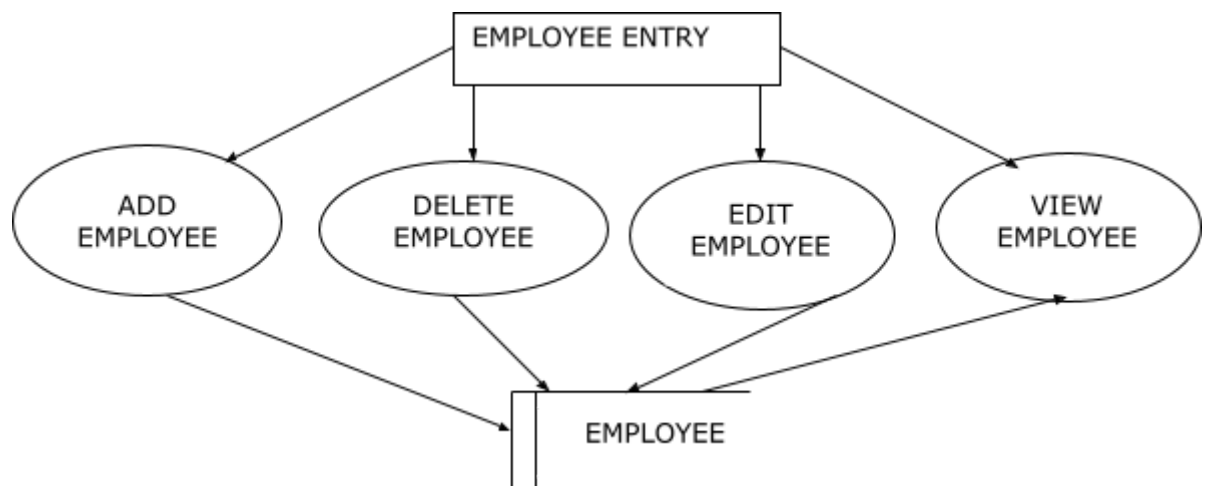


ADMINISTRATIVE MODULE FLOW PROCESS

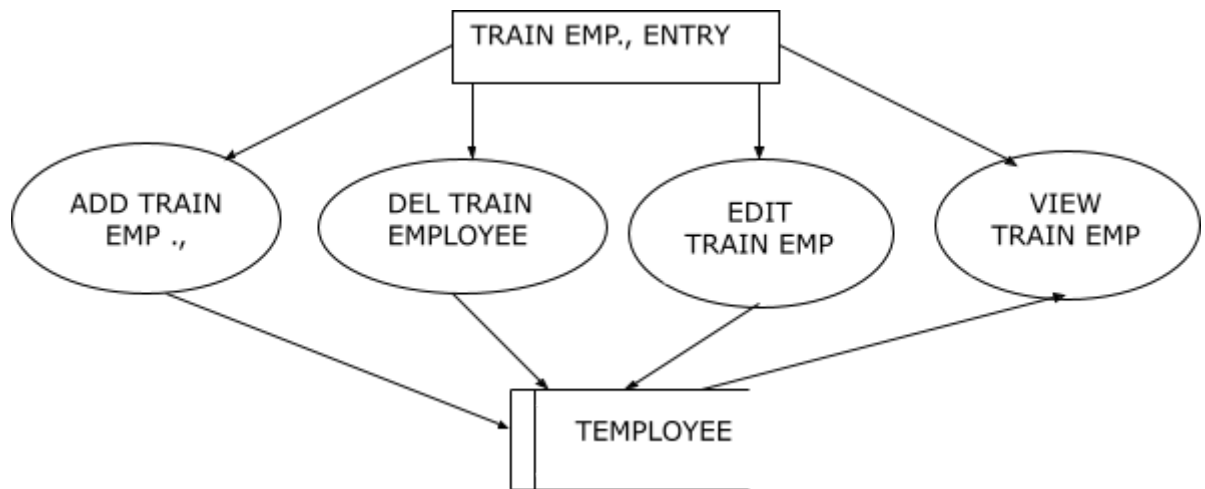
PROJECT MAINTENANCE FLOW PROCESS



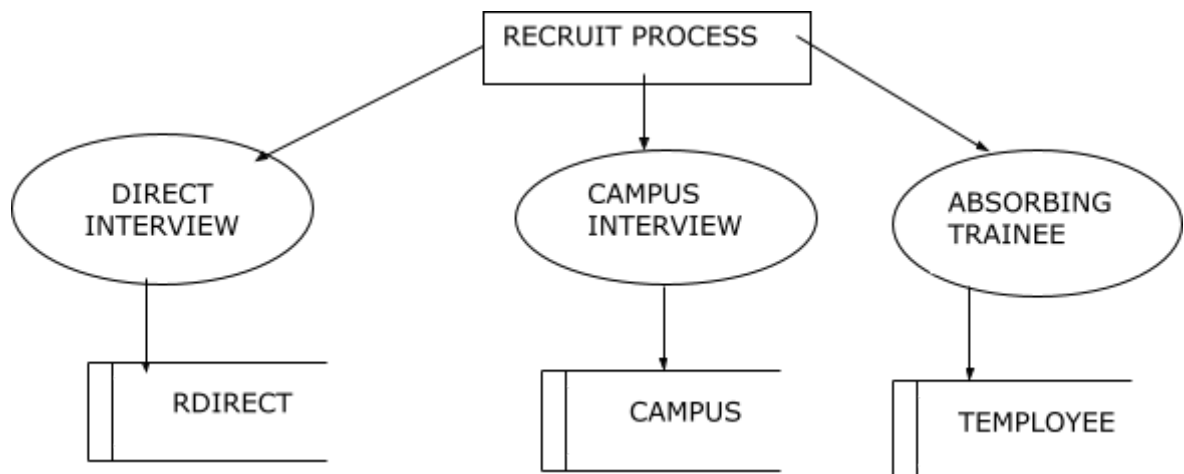
EMPLOYEE DETAILS FLOW PROCESS



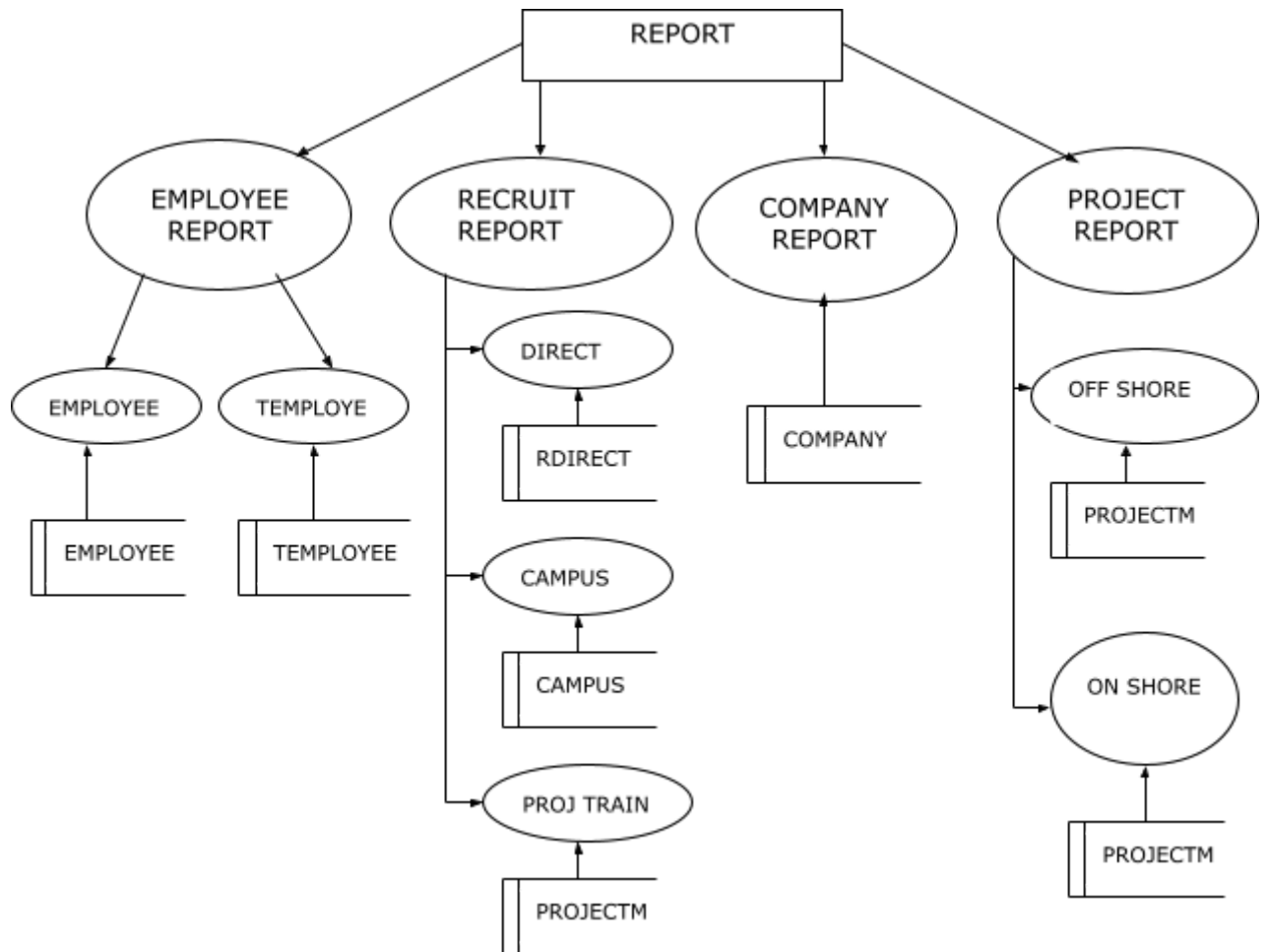
TRAINING EMPLOYEE'S FLOW PROCESS



RECRUITMENT FLOW PROCESS



REPORTS FLOW PROCESS



DATABASE DESIGN

Admin Table :

Field Name	Data Type
Adminname	Text
Adminpwd	Text

Answer Table :

Field Name	Data Type
No	Number
a1	Text
a2	Text
a3	Text
a4	Text
a5	Text
a6	Text
a7	Text

Campus Table :

Field name	Data type
Recode	Text
Name	Text
Cname	Text
Caddress	Text
Doi	Text
Perfor	Text
Parea	Text
Aot	Text
Refby	Text

Choice Table :

Field name	Data type
Qno	Number
c1	Memo
c2	Memo
c3	Memo
c4	Memo

Company Table :

Field name	Data type
Cname	Text

Address	Text
Country	Text
City	Text
Area	Text
Phone	Number
Email	Text
url	Text
Mname	Text
Yos	Text

Employee table :

Field name	Data type
Ecode	Number
Ename	Text
Email	Text
Quali	Text
Ccexp	Text
Pcexp	Text
Bgroup	Text
Dob	Text
Jcode	Number
Jnature	Text
Mstatus	Text
Ref	Text
Paddress	Text
Peradd	Text

Projectm :

Field name	Data type
Pcode	Number
Pcat	Text
Pplace	Text
Cname	Text
Cnature	Text
Pname	Text
Parea	Text
Psdate	Text
Pedate	Text

Questions :

Field name	Data type
Sno	Number
q1	Memo
q2	Memo
q3	Memo
q4	Memo
q5	Memo

q6	Memo
q7	Memo

Rdirect :

Field name	Data type
Rcode	Number
Name	Text
Passwd	Text
Marks	Number

Rproject :

Field name	Data type
reccode	Text
Name	Text
Ptitle	Text
Plang	Text
Pplace	Text
Pfdate	Text
Ptdate	Text
Doi	Text
Email	Text
Address	Text
Quali	Text

Employee :

Field name	Data type
Ecode	Number
Ename	Text
Email	Text
Jcode	Number
Tcategory	Text
Tplace	Text
Tsdate	Text
Tedate	Text
Tperson	Text

Userreg :

Field name	Data type
Userid	Text
Passwd	Text
Sex	Text
Dob	Text
Address	Text
City	Text
Country	Text

Quali	Text
Skills	Text
Phone	Number
email1	Text
email2	Text

SOURCE CODE

Program No	Program Name	Program Description
1	New User Sign	This program helps to get the Personnel Information Details (i.e.) getting the information of Customer's name, his password, sex, contact address etc.
2	User Login	Selects the name and password from the stored details of the database using SQL query.
3	Online Test	All Users must Attend the Online Test and who qualified in this test, which one selected for the direct interview.
4	Results	We know about the Online test's Results
5	Company Details	Here view about all project Company's Details
6	Employee Details	Here enter the working employee's details. i.e., empname, Address etc.,
7	Training Employee's Details	Here enter the Training employee's details. i.e., train area, train place etc.,
8	Recruitment Process	Here Recruit all students and publics. Here all of them should attend the online test.
9	Recruit Reports	Here should view about all the recruit reports. I.e., direct interview, campus inter view, absorbing trainee.
10	Employee Report	Let the option to proceed further details about employee and training employees
11	Project Report	Let the option to proceed further details about the onshore or the off shore projects
12	Campus Interview	Who one had selected in online test that person should attend the Campus interview. Here those people details are stored.

1.New users Login Java Servlet:

```
import javax.servlet.http.*;
import javax.servlet.*;
import java.sql.*;
import java.io.*;

public class usersreg extends HttpServlet
{
    Connection c;
    public void doGet(HttpServletRequest re,HttpServletResponse rs) throws
ServletException,IOException
    {
        rs.setContentType("text/html");
        PrintWriter out=rs.getWriter();
        String name=re.getParameter("t1");
        String passwd=re.getParameter("t2");
        String sex=re.getParameter("t3");
        String date=re.getParameter("date");
        String mon=re.getParameter("mon");
        String year=re.getParameter("year");
        String address=re.getParameter("t5");
        String city=re.getParameter("t6");
        String country=re.getParameter("t7");
        String pin=re.getParameter("pin");
        long pincode=Long.parseLong(pin);
        String quali=re.getParameter("t8");
        String skill=re.getParameter("t9");
        String phone1=re.getParameter("t10");
        long phone=Long.parseLong(phone1);
        String e1=re.getParameter("t11");
        String e2=re.getParameter("t12");
        String dob=date+"/"+mon+"/"+year;

try
    {
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        c=DriverManager.getConnection("jdbc:odbc:hrms");
```

```

Statement s=c.createStatement();
ResultSet r;
r=s.executeQuery("select username from userreg");
int temp=0;
while(r.next())
{
    if(name.equals(r.getString(1)))
    {
        temp=1;
        break;
    }
}
if(temp==1)
{
    out.println("<html><body bgcolor=skyblue><center>");
    out.println("<br><br><br><pre align=center>");
    out.println("<h3>Sorry,UserName Already Exist, Please Try  
Again</h3><br><br>");
    out.println("<a href='http://localhost:8080/servlet/newuserlogin'>Back</a>");
    out.println("</center></body></html>");
}
else
{
    String str="insert into userreg  
values('"+name+"','"+passwd+"','"+sex+"','"+dob+"','"+address+"','"+city+"','"+country+"','"+pincode+"','"+quali+"','"+skill+"','"+phone+"','"+e1+"','"+e2+"')";
    //out.println(str);
    s.executeUpdate(str);
    out.println("<html><body bgcolor=skyblue><center>");
    out.println("<br><br><br><pre align=center>");
    out.println("<h3>YOUR DETAILS ARE STORED</h3><br><br>");
    out.println("<a href='http://localhost:8080/servlet/usersmodule'>Proceed  
Further</a>");
    out.println("</center></body></html>");
}
}
catch(SQLException sqle)
{
    out.println(sqle);
}
catch(ClassNotFoundException cnfe)
{
    out.println(cnfe);
}
}
}

```

```
import javax.servlet.http.*;
import javax.servlet.*;
import java.sql.*;
import java.io.*;
public class userslogin extends HttpServlet
{
    public void doGet(HttpServletRequest re,HttpServletResponse rs) throws
ServletException,IOException
    {
        rs.setContentType("text/html");
        PrintWriter out=rs.getWriter();
        out.println("<html>");
        out.println("<head>");
        out.println("<title></title>");
        out.println("<style>a{color:black}</style></head>");
        out.println("<body bgcolor=skyblue text=black>");
            out.println("<form          name=per          method=post
action='http://localhost:8080/servlet/userslogin'>");
        out.println("<font face=Verdana>");
        out.println("<h2 align=center>");
        out.println(" Users Login Module");
        out.println("</h2><br><br><table align=center><tr>");
            out.println("<td><font          size=-1          face=Verdana><b>User
name</b></font></td>");
        out.println("<td><input type=text name=uname></td></tr><tr>");
        out.println("<td><font size=-1 face=Verdana><b>Password</b></font></td>");
            out.println("<td><input          type=password          name=upass>
</td></tr></table><br><br>");
        out.println("<center><input type=submit name=sub1 value=Submit>");
        out.println("<input type=reset name=can value=Cancel></center><br>");
            out.println("<center>&nbsp;&nbsp;&nbsp;<a
href='http://localhost:8080/servlet/usersmodule'><font
size=-1><b>Back</b></font></a>");
        out.println(" &nbsp;&nbsp;&nbsp;<a
href='http://localhost:8080/servlet/newuserlogin1'><font          size=-1><b>New
User</b></font></a>");
        out.println("</center>");
    }
}
```

```

        out.println("</form></body></html>");
    }
    public void doPost(HttpServletRequest re,HttpServletResponse rs) throws
ServletException,IOException
    {
        rs.setContentType("text/html");
        PrintWriter out=rs.getWriter();
        HttpSession hp=re.getSession(true);
        String name=re.getParameter("uname");

```

```

String pass=re.getParameter("upass");
        hp.putValue("username",name);
        hp.putValue("password",pass);
        Connection con;
        try
        {
            Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            con=DriverManager.getConnection("jdbc:odbc:hrms");
            Statement s=con.createStatement();
            ResultSet r=s.executeQuery("select *from userreg");
            int temp=0;
            while(r.next())
            {
                if(name.equals(r.getString(1)) && pass.equals(r.getString(2)))
                {
                    temp=1;
                }
            }
            if(temp==1)
            {
                out.println("<HTML>");
                out.println("<HEAD><center><META NAME='GENERATOR' Content='Microsoft
Developer Studio'>");
                out.println("<META HTTP-EQUIV='Content-Type' content='text/html'
charset=iso-8859-1>");
                out.println("<TITLE>Document
Title</TITLE><style>a{color:blue}</style></HEAD>");
                out.println("<BODY bgcolor=skyblue><br><br><p align=center>");
                out.println("<font face=Verdana size=-1> <b>Hello,<u><font
color=red></font></u></b></font>");
                out.println("<br><br><br>");
                out.println("<h3><a href='http://localhost:8080/servlet/testing' style='color:
rgb(0,0,255)'>Click here for Online Test</a></h3>");

```

```

        out.println("<br><br><br><br><br>");
        out.println("<a
href='http://localhost:8080/servlet/usersmodule'>Back</a>&nbsp;&nbsp;&nbsp;");
        out.println("<a href='http://localhost:8080/servlet/home'>Home
Page</a></center></BODY></HTML>");
    }

```

```

else
{
    out.println("<body bgcolor=skyblue><center><br><br><br>");
    out.println("<h2 align=center>");
    out.println("YOU ARE NOT A VALID USER");
    out.println("</h2><br><br><a
href='http://localhost:8080/servlet/usermodule'><font
size=-1><b>Back</b></center>");
}

}
catch(SQLException sqle)
{
    out.println(sqle);
}
catch(ClassNotFoundException cnfe)
{
    out.println(cnfe);
}
}
}
}

```


4. Online Test in Java Servlet:

```
import javax.servlet.http.*;
import javax.servlet.*;
import java.sql.*;
import java.io.*;
public class testing extends HttpServlet
{
    Connection con;
    public void doGet(HttpServletRequest re,HttpServletResponse rs) throws
    IOException,ServletException
    {
        rs.setContentType("text/html");
        PrintWriter out=rs.getWriter();
        Statement s=null,s1=null;
        ResultSet r=null,r2=null;
        double a1;
        a1=java.lang.Math.random();
        a1=a1*1000;
        int a2=(int)a1;
        int a3=a2%4;
        a3=a3+1;
        Integer a4=new Integer(a3);
        HttpSession hs=re.getSession(true);
        hs.putValue("key",a4);
        try
        {
            Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            con=DriverManager.getConnection("jdbc:odbc:hrms");
            s=con.createStatement();
            s1=con.createStatement();
            r=s.executeQuery("select * from questions where [sno]="+a3);
            r2=s1.executeQuery("select * from choice where [qno]="+a3);
            /* while(r2.next())
            {
                out.println(r2.getString(2));
                out.println(r2.getString(3));
                out.println(r2.getString(4));
```

```

        out.println(r2.getString(5));
    }

    while(r.next())
    {
        out.println(r.getString(2));
        out.println(r.getString(3));
        out.println(r.getString(4));
        out.println(r.getString(5));
        out.println(r.getString(6));
        out.println(r.getString(7));
        out.println(r.getString(8));

    }*/
    r.next();
    r2.next();
    out.println("<html><body
bgcolor=skyblue><br><h2><center>QUESTIONS</center></h2>");
    out.println("<form method=post
action='http://localhost:8080/servlet/results'>");

    String c1,c2,c3,c4;
    c1=r2.getString(2);
    //out.println(c1);
    c2=r2.getString(3);
    c3=r2.getString(4);
    c4=r2.getString(5);
    out.println("<br><br><h3>1."+r.getString(2)+"</h3>");
    out.println("<input type=radio name=q1 value='"+c1+"'>"+c1);
    out.println("<input type=radio name=q1 value='"+c2+"'>"+c2);
    out.println("<input type=radio name=q1 value='"+c3+"'>"+c3);
    out.println("<input type=radio name=q1 value='"+c4+"'>"+c4);
    r2.next();

    c1=r2.getString(2);
    c2=r2.getString(3);
    c3=r2.getString(4);
    c4=r2.getString(5);
    out.println("<br><br><h3>2."+r.getString(3)+"</h3>");
    out.println("<input type=radio name=q2 value='"+c1+"'>"+c1);
    out.println("<input type=radio name=q2 value='"+c2+"'>"+c2);
    out.println("<input type=radio name=q2 value='"+c3+"'>"+c3);
    out.println("<input type=radio name=q2 value='"+c4+"'>"+c4);
    r2.next();

    c1=r2.getString(2);
    c2=r2.getString(3);
    c3=r2.getString(4);
    c4=r2.getString(5);
    out.println("<br><br><h3>3."+r.getString(4)+"</h3>");
    out.println("<input type=radio name=q3 value='"+c1+"'>"+c1);
    out.println("<input type=radio name=q3 value='"+c2+"'>"+c2);
    out.println("<input type=radio name=q3 value='"+c3+"'>"+c3);

```

```

        out.println("<input type=radio name=q3 value=\"" + c4 + "\">" + c4);
        r2.next();
        c1=r2.getString(2);
        c2=r2.getString(3);
        c3=r2.getString(4);
        c4=r2.getString(5);
        out.println("<br><br><h3>4." + r.getString(5) + "</h3>");
        out.println("<input type=radio name=q4 value=\"" + c1 + "\">" + c1);
        out.println("<input type=radio name=q4 value=\"" + c2 + "\">" + c2);
        out.println("<input type=radio name=q4 value=\"" + c3 + "\">" + c3);
        out.println("<input type=radio name=q4 value=\"" + c4 + "\">" + c4);
        r2.next();
        c1=r2.getString(2);
        c2=r2.getString(3);
        c3=r2.getString(4);
        c4=r2.getString(5);
        out.println("<br><br><h3>5." + r.getString(6) + "</h3>");
        out.println("<input type=radio name=q5 value=\"" + c1 + "\">" + c1);
        out.println("<input type=radio name=q5 value=\"" + c2 + "\">" + c2);
        out.println("<input type=radio name=q5 value=\"" + c3 + "\">" + c3);
        out.println("<input type=radio name=q5 value=\"" + c4 + "\">" + c4);
        r2.next();
        c1=r2.getString(2);
        c2=r2.getString(3);
        c3=r2.getString(4);
        c4=r2.getString(5);
        out.println("<br><br><h3>6." + r.getString(7) + "</h3>");
        out.println("<input type=radio name=q6 value=\"" + c1 + "\">" + c1);
        out.println("<input type=radio name=q6 value=\"" + c2 + "\">" + c2);
        out.println("<input type=radio name=q6 value=\"" + c3 + "\">" + c3);
        out.println("<input type=radio name=q6 value=\"" + c4 + "\">" + c4);
        r2.next();
        c1=r2.getString(2);
        c2=r2.getString(3);
        c3=r2.getString(4);
        c4=r2.getString(5);
        out.println("<br><br><h3>7." + r.getString(8) + "</h3>");
        out.println("<input type=radio name=q7 value=\"" + c1 + "\">" + c1);
        out.println("<input type=radio name=q7 value=\"" + c2 + "\">" + c2);
        out.println("<input type=radio name=q7 value=\"" + c3 + "\">" + c3);
        out.println("<input type=radio name=q7 value=\"" + c4 + "\">" + c4);
        out.println("<br><br><br><center><input type=submit  

value=send></center>");
        out.println("</form></body></html>");
    }
    catch(SQLException sqle)
    {
        out.println(sqle);
    }
    catch(ClassNotFoundException cnfe)
    {
        out.println(cnfe);
    }
}

```

```

try
{
    s.close();s1.close();r2.close();r.close();
}
catch(SQLException sqle1)
{
    out.println(sqle1);
}
}
}
}

```

4. Results in Java Servlet:

```

import javax.servlet.http.*;
import javax.servlet.*;
import java.sql.*;
import java.io.*;
public class results extends HttpServlet
{
    Connection c=null;
    int rcode1;
    public void doPost(HttpServletRequest re,HttpServletResponse rs) throws
IOException,ServletException
    {
        rs.setContentType("text/html");
        PrintWriter out=rs.getWriter();
        int mark=0,temp=0,rcode;
        try
        {
            Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            c=DriverManager.getConnection("jdbc:odbc:hrms");
            Statement s=c.createStatement();
            Statement s1=c.createStatement();
            Statement st1=c.createStatement();
            Statement st=c.createStatement();
            HttpSession hp=re.getSession(true);
            Integer in=(Integer)hp.getValue("key");
            int no=in.intValue();
            // String a="SELECT answers.[no], answers.a1, answers.a2, answers.a3,
answers.a4, answers.a5, answers.a6, answers.a7 FROM answers WHERE
(((answers.[no])="+no+"))";
            String a="select *from answers where [no]="+no;
            //out.println(a);
            ResultSet r=st.executeQuery(a);
            //out.println(r.next());
            /* while(r.next())
            {
                out.println(r.getString(2));
                out.println(r.getString(3));
            }
        }
    }
}

```

```

        out.println(r.getString(4));
        out.println(r.getString(5));
        out.println(r.getString(6));
        out.println(r.getString(7));
        out.println(r.getString(8));
    }*/
    HttpSession hs=re.getSession(true);
    String a1=re.getParameter("q1");
    //out.println(a1);
    String a2=re.getParameter("q2");
    //out.println(a2);
    String a3=re.getParameter("q3");
    //out.println(a3);
    String a4=re.getParameter("q4");
    //out.println(a4);
    String a5=re.getParameter("q5");
    //out.println(a5);
    String a6=re.getParameter("q6");
    //out.println(a6);
    String a7=re.getParameter("q7");
    //out.println(a7);
    while(r.next())
    {
        if(a1.equalsIgnoreCase(r.getString(2)))
            mark=mark+1;
        if(a2.equalsIgnoreCase(r.getString(3)))
            mark=mark+1;
        if(a3.equalsIgnoreCase(r.getString(4)))
            mark=mark+1;
        if(a4.equalsIgnoreCase(r.getString(5)))
            mark=mark+1;
        if(a5.equalsIgnoreCase(r.getString(6)))
            mark=mark+1;
        if(a6.equalsIgnoreCase(r.getString(7)))
            mark=mark+1;
        if(a7.equalsIgnoreCase(r.getString(8)))
            mark=mark+1;
    }
    // out.println(mark);
    if(mark>5)
    {
        String name=(String)hs.getValue("username");
        String pass=(String)hs.getValue("password");
        ResultSet rs1=st.executeQuery("select * from rdirect");
        while(rs1.next())
        {
            temp=1;
            break;
        }
        if(temp==1)
        {
            rs1=st1.executeQuery("select max(rcode) from rdirect");
            while(rs1.next())

```

```

        {
            rcode1=rs1.getInt(1);
            rcode1=rcode1+1;
        }
    }
    else
    {
        rcode1=1;
    }
    s1.executeUpdate("insert into rdirect
values("+rcode1+", '"+name+"', '"+pass+"', '"+mark+"")");
    out.println("<html>");
    out.println("<center><body
bgcolor=skyblue><center><br><br><br>");
    out.println("<h3>CONGRATS! YOU ARE
SELECTED</h3></body></center>");
    out.println("<br><br>Please contact us in the following
address<br>");
    out.println("<br>Hindustan Software Limited<br>");
    out.println("No.120 Wallajah Road,<br>");
    out.println("Triplicane, Chennai-2");
    out.println("<br><br><a
href='http://localhost:8080/servlet/home'><h4>BACK</h4></a></center>"
);
    }
    else
    {
        out.println("<html>");
        out.println("<center><body
bgcolor=skyblue><center><br><br><br>");
        //out.println("<h2 align=center>");
        out.println("<h3>SORRY YOU ARE NOT SELECTED. BETTER LUCK NEXT
TIME</h3></body></center>");
        out.println("<br><br><a
href='http://localhost:8080/servlet/home'><h4>BACK</h4></a></center>"
);
    }

    }
    catch(SQLException sqle)
    {
        out.println(sqle);
    }
    catch(ClassNotFoundException cnfe)
    {
        out.println(cnfe);
    }
}
}
}

```

5. Company Details in Java Servlet:

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.sql.*;

public class compview extends HttpServlet {

    Connection c4;
    Statement st;
    ResultSet rs;
    String s1="";
    String ss1="";

    public void init(ServletConfig sc) throws ServletException
    {
        try
        {
            Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            c4=DriverManager.getConnection("jdbc:odbc:hrms","","");
        }
        catch(Exception ee4){}
    }

    public void doGet(HttpServletRequest req,HttpServletResponse res)
        throws ServletException,IOException {

        PrintWriter pw=res.getWriter();

        //s1=req.getParameter("s1");
        int temp=0;
        try{

            st=c4.createStatement();

            ss1="select * from company";
```

```

        rs=st.executeQuery(ss1);
        pw.println("<html><body
bgcolor=skyblue><style>a{color:black}</style><center><h3>COMPANY
DETAILS</h3><br><br>");
        pw.println("<table border=2 align=center>");
        pw.println("<tr><th>Company name
</th><th>Address</th><th>country</th><th>city</th><th>area</th><t
h>phone</th><th>email</th><th>url</th><th>Manager
name</th><th>Year of origin</th></tr>");

        while(rs.next())
        {

            pw.println("<tr><td>" +rs.getString(1) +"</td><td>"
+rs.getString(2) +"</td><td>" +rs.getString(3) +"</td><td>"
+rs.getString(4) +"</td><td>" +rs.getString(5) +"</td><td>"
+rs.getInt(6) +"</td><td>" +rs.getString(7) +"</td><td>"
+rs.getString(8) +"</td><td>" +rs.getString(9) +"</td><td>"
+rs.getString(10) +"</td></tr>");
            temp=1;
        }

        pw.println("</table>");
        pw.println("<br><br>");

        pw.println("<br><br>");
        pw.println("<a
href='http://localhost:8080/servlet/companye'>BACK</a></center>");
        if(temp==0)
            pw.println("NO DATA FOUND");

    }catch(SQLException ee1) {pw.println(ee1);}

}
}

```


7.Employee details in Java Servlet:

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.sql.*;

public class empview extends HttpServlet {

    Connection c4;
    Statement st;
    ResultSet rs;
    String s1="";
    String ss1="";

    public void init(ServletConfig sc) throws ServletException
    {
        try
        {
            Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            c4=DriverManager.getConnection("jdbc:odbc:hrms","","");
        }
        catch(Exception ee4){}
    }

    public void doGet(HttpServletRequest req,HttpServletResponse res)
        throws ServletException,IOException {

        PrintWriter pw=res.getWriter();

        //s1=req.getParameter("s1");

        try{

            st=c4.createStatement();

            ss1="select * from employee";
```

```

rs=st.executeQuery(ss1);

pw.println("<html><body
bgcolor=skyblue><h4><style>a{color:black}</style></h4><br><center><h2>E
mployee Details</h2></center><br>");
pw.println("<table border=2 align=center>");
pw.println("<tr><th>Employee No</th><th>E
name</th><th>Email</th><th>Quali</th><th>Ccexp</th><th>Pcexp</th><th>
Bgroup</th><th>DOB</th><th>JobCode
</th><th>JobNature</th><th>Mstatus</th><th>Reference</th><th>Paddress</
th><th>Peradd</th></tr>");
int temp=0;
while(rs.next())
{
    temp=1;
    pw.println("<tr><td>" +rs.getInt(1) +"</td><td>"
+rs.getString(2)+"</td><td>" +rs.getString(3) +"</td><td>" +rs.getString(4)
+"</td><td>" +rs.getString(5) +"</td><td>" +rs.getString(6) +"</td><td>"
+rs.getString(7) +"</td><td>" +rs.getString(8) +"</td><td>" +rs.getInt(9)
+"</td><td>" +rs.getString(10) +"</td><td>" +rs.getString(11) +"</td><td>"
+rs.getString(12) +"</td><td>" +rs.getString(13) +"</td><td>"
+rs.getString(14) +"</td></tr>");
}

pw.println("</table>");
pw.println("<br><br>");
pw.println("<br><br><center><a
href='http://localhost:8080/servlet/employ'>BACK</a></center><br><br>");
if(temp==0)
    pw.println("NO DATA FOUND");

}catch(SQLException ee1) {pw.println(ee1);}

}
}

```

8. Training Employee Details:

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.sql.*;
public class tempview extends HttpServlet {

    Connection c4;
    Statement st;
    ResultSet rs;
    String s1="";
    String ss1="";

    public void init(ServletConfig sc) throws ServletException
    {
        try
        {
            Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            c4=DriverManager.getConnection("jdbc:odbc:hrms","","");
        }
        catch(Exception ee4){}
    }

    public void doGet(HttpServletRequest req,HttpServletResponse res)
        throws ServletException,IOException {

        PrintWriter pw=res.getWriter();

        //s1=req.getParameter("s1");

        try{

            st=c4.createStatement();

            ss1="select * from tempview";
```

```

rs=st.executeQuery(ss1);

pw.println("<html><body bgcolor=skyblue><br><center><h2>Trainee
Details</h2></center><br>");
pw.println("<table border=1 align=center>");
pw.println("<tr><th>Employee No</th><th>Employee
name</th><th>Email</th><th>JobCode</th><th>Training
Category</th><th>Training
Place</th><th>TSDate</th><th>TEDate</th><th>Train Person</th></tr>");
int temp=0;
while(rs.next())

{
    temp=1;
    pw.println("<tr><td>" +rs.getInt(1) + "</td><td>"
+rs.getString(2)+"</td><td>" +rs.getString(3) + "</td><td>" +rs.getInt(4)
+ "</td><td>" +rs.getString(5) + "</td><td>" +rs.getString(6) + "</td><td>"
+rs.getString(7) + "</td><td>" +rs.getString(8)
+ "</td><td>" +rs.getString(9)+"</td></tr>");
}

pw.println("</table>");
pw.println("<br><br>");
pw.println("<br><br><center><a
href='http://localhost:8080/servlet/tempemploy'>BACK</a></center><br><br>");
if(temp==0)
    pw.println("NO DATA FOUND");

}catch(SQLException ee1) {pw.println(ee1);}

}
}

```

9.Recruitment Process in Java Servlet:

```
import javax.servlet.http.*;
import javax.servlet.*;
import java.sql.*;
import java.io.*;

public class rectproject extends HttpServlet
{
    Connection c;
    public void doPost(HttpServletRequest re,HttpServletResponse rs) throws
ServletException,IOException
    {
        rs.setContentType("text/html");
        PrintWriter out=rs.getWriter();
        String rcode=re.getParameter("t1");
        String name=re.getParameter("t2");
        String ptitle=re.getParameter("t3");
        String plang=re.getParameter("t4");
        String pplace=re.getParameter("t5");
        String sdate=re.getParameter("sdate");
        String smon=re.getParameter("smon");
        String syear=re.getParameter("syear");
        String edate=re.getParameter("edate");
        String emon=re.getParameter("emon");
        String eyear=re.getParameter("eyear");
        String idate=re.getParameter("idate");
        String imon=re.getParameter("imon");
        String iyear=re.getParameter("iyear");
        String email=re.getParameter("t9");
        String address=re.getParameter("t10");
        String quali=re.getParameter("t11");
```

```

String pfd=sdate+"/"+smon+"/"+syear;
String ptd=edate+"/"+emon+"/"+eyear;
String doi=idate+"/"+imon+"/"+iyear;

try
{
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
    c=DriverManager.getConnection("jdbc:odbc:hrms");
    Statement s=c.createStatement();
    ResultSet r;
    r=s.executeQuery("select reccode from rproject");
    int temp=0;
    while(r.next())
    {
        if(rcode.equals(r.getString(1)))
        {
            temp=1;
            break;
        }
    }
    if(temp==1)
    {
        out.println("<html><body bgcolor=skyblue><center>");
        out.println("<br><br><br><h3>Recruitment Code Already Exist, Please Try Again</h3>");
        out.println("<br><br><br><a href='http://localhost:8080/servlet/recruitment'>Back</a>");
        out.println("</center></body></html>");
    }
    else
    {
        String str="insert into rproject values('"+rcode+"','"+name+"','"+ptitle+"','"+plang+"','"+pplace+"','"+pfd+"','"+ptd+"','"+doi+"','"+email+"','"+address+"','"+quali+"')";
        //out.println(str);

        s.executeUpdate(str);
        out.println("<html><body bgcolor=skyblue><center>");
        out.println("<br><br><br><h3>DETAILS ARE STORED</h3>");
        // out.println("<a href='http://localhost:8080/usermo.html'>Proceed Further</a>");
        out.println("<br><br><br><a href='http://localhost:8080/servlet/rectproject1'>Back</a>");
        out.println("</center></body></html>");

    }
}
catch(SQLException sqle)
{
    out.println(sqle);
}
catch(ClassNotFoundException cnfe)

```

```

    {
        out.println(cnfe);
    }
}
}

```

10.Employee Report in Java Servlet:

```

import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.sql.*;

public class empreport extends HttpServlet {

    Connection c4;
    Statement st;
    ResultSet rs;
    String s1="";
    String ss1="";

    public void init(ServletConfig sc) throws ServletException
    {
        try
        {
            Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            c4=DriverManager.getConnection("jdbc:odbc:hrms","","");
        }
        catch(Exception ee4){}
    }

    public void doGet(HttpServletRequest req,HttpServletResponse res) throws
ServletException,IOException
    {
        res.setContentType("text/html");
        PrintWriter pw=res.getWriter();
        pw.println("<html><body bgcolor=skyblue>");
        pw.println("<form method=post
action='http://localhost:8080/servlet/empreport'>");
        pw.println("<center><br><br><br><br><h3><input type=radio name=t
value=emp> Employee Report<br>");
        pw.println("<br><input type=radio name=t value=tra> Trainee Report</h3>");
        pw.println("<br><br><br><input type=submit value=Display></center>");
    }
}

```

```

        pw.println("<br><br><center><a
href='http://localhost:8080/servlet/rectreport'>BACK</a></center><br><br>");
        pw.println("</form></body></html>");
    }

    public void doPost(HttpServletRequest req,HttpServletResponse res) throws
ServletException,IOException
    {

        res.setContentType("text/html");
        PrintWriter pw=res.getWriter();
        String str=req.getParameter("t");
        if(str.equals("emp"))
        {
            try{

                st=c4.createStatement();

                ss1="select * from employee";

                rs=st.executeQuery(ss1);

                pw.println("<html><body bgcolor=skyblue><center><h3>EMPLOYEE
REPORT</h3><br><br>");
                pw.println("<table border=2 align=center>");
                pw.println("<tr><th>Employee No</th><th>E
name</th><th>Email</th><th>Quali</th><th>Ccexp</th><th>Pcexp</th><th>
Bgroup</th><th>DOB</th><th>JobCode
</th><th>JobNature</th><th>Mstatus</th><th>Reference</th><th>Paddress</
th><th>Peradd</th></tr>");
                int temp=0;
                while(rs.next())
                {
                    temp=1;
                    pw.println("<tr><td>" +rs.getInt(1) + "</td><td>"
+rs.getString(2)+"</td><td>" +rs.getString(3) + "</td><td>" +rs.getString(4)
+"</td><td>" +rs.getString(5) + "</td><td>" +rs.getString(6) + "</td><td>"
+rs.getString(7) + "</td><td>" +rs.getString(8) + "</td><td>" +rs.getInt(9)
+"</td><td>" +rs.getString(10) + "</td><td>" +rs.getString(11) + "</td><td>"
+rs.getString(12) + "</td><td>" +rs.getString(13) + "</td><td>"
+rs.getString(14) + "</td></tr>");
                }

                pw.println("</table>");
                pw.println("<br><br>");
                pw.println("<a
href='http://localhost:8080/servlet/empreport'>BACK</a></center></body></htm
l>");
                if(temp==0)
                    pw.println("NO DATA FOUND");

            }catch(SQLException ee1) {pw.println(ee1);}
        }
    }

```



```

    }

    if(str.equals("tra"))
    {
        try{

            st=c4.createStatement();

            ss1="select * from tempolyee";

            rs=st.executeQuery(ss1);

            pw.println("<html><body bgcolor=skyblue><br><center><h2>Trainee
Details</h2></center><br>");
            pw.println("<table border=1 align=center>");
            pw.println("<tr><th>Employee No</th><th>Employee
name</th><th>Email</th><th>JobCode</th><th>Training
Category</th><th>Training
Place</th><th>TSDate</th><th>TEDate</th><th>Train Person</th></tr>");
            int temp=0;
            while(rs.next())
            {
                temp=1;
                pw.println("<tr><td>" +rs.getInt(1) + "</td><td>"
+rs.getString(2)+"</td><td>" +rs.getString(3) + "</td><td>" +rs.getInt(4)
+"</td><td>" +rs.getString(5) + "</td><td>" +rs.getString(6) + "</td><td>"
+rs.getString(7) + "</td><td>" +rs.getString(8)
+"</td><td>" +rs.getString(9)+"</td></tr>");
            }

            pw.println("</table>");
            pw.println("<br><br>");
            pw.println("<br><br><center><a
href='http://localhost:8080/servlet/empreport'>BACK</a></center><br><br>");
            if(temp==0)
                pw.println("NO DATA FOUND");

        }catch(SQLException ee1) {pw.println(ee1);}
    }
}
}

```

11.Project Report in Java Servlet:

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.sql.*;
public class projreport extends HttpServlet {

    Connection c4;
    Statement st;
    ResultSet rs;
    String s1="";
    String ss1="";

    public void init(ServletConfig sc) throws ServletException
    {
        try
        {
            Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            c4=DriverManager.getConnection("jdbc:odbc:hrms","","");
        }
        catch(Exception ee4){}
    }

    public void doGet(HttpServletRequest req,HttpServletResponse res) throws
ServletException,IOException
    {
        res.setContentType("text/html");
        PrintWriter pw=res.getWriter();
        pw.println("<html><body bgcolor=skyblue><center><br><br><br>");
        pw.println("<form method=post
action='http://localhost:8080/servlet/projreport'>");
        pw.println("<h3><input type=radio name=t value=offs> OffShore Projects
Report");
```

```

        pw.println("<br><br><input type=radio name=t value=ons> OnShore Project  
Report Report</h3>");
        pw.println("<br><br><input type=submit value=Display><center>");
        pw.println("<br><br><a  
href='http://localhost:8080/servlet/rectreport'>BACK</a>");
        pw.println("</form></body></html>");
    }

```

```

    public void doPost(HttpServletRequest req,HttpServletResponse res) throws  
ServletException,IOException
    {

```

```

        res.setContentType("text/html");
        PrintWriter pw=res.getWriter();
        String str=req.getParameter("t");
        if(str.equals("offs"))

```

```

    {

```

```

        try{

```

```

            st=c4.createStatement();

```

```

            ss1="select * from projectm where pplace='offshore'";

```

```

            rs=st.executeQuery(ss1);

```

```

            pw.println("<html><body bgcolor=skyblue><center><h3>OFFSHORE  
PROJECT DETAILS</h3><br><br>");
            pw.println("<table border=2 align=center>");
            pw.println("<tr><th>Project No</th><th>Project Type</th><th>Project  
place</th><th>Client name</th><th>Client nature</th><th>Project  
name</th><th>Project area</th><th>Project sdate</th><th>Project  
edate</th></tr>");

```

```

            int temp=0;
            while(rs.next())

```

```

            {

```

```

                temp=1;

```

```

                pw.println("<tr><td>" +rs.getInt(1) + "</td><td>"  
+rs.getString(2)+"</td><td>" +rs.getString(3) + "</td><td>" +rs.getString(4)  
+"</td><td>" +rs.getString(5) + "</td><td>" +rs.getString(6) + "</td><td>"  
+rs.getString(7) + "</td><td>" +rs.getString(8) + "</td><td>" +rs.getString(9)  
+"</td></tr>");

```

```

            }

```

```

            pw.println("</table>");
            pw.println("<br><br>");
            pw.println("<a

```

```

href='http://localhost:8080/servlet/projreport'>BACK</a></center></body></html  
>");

```

```

            if(temp==0)

```

```

                pw.println("NO DATA FOUND");

```

```

    }catch(SQLException ee1) {pw.println(ee1);}
    }
    if(str.equals("ons"))
    {
        try{

            st=c4.createStatement();

            ss1="select * from projectm where pplace='onshore'";

            rs=st.executeQuery(ss1);

            pw.println("<html><body bgcolor=skyblue><center><h3>ONSHORE
PROJECT DETAILS</h3><br><br>");
            pw.println("<table border=2 align=center>");

                pw.println("<tr><th>Project No</th><th>Project Type</th><th>Project
place</th><th>Client name</th><th>Client nature</th><th>Project
name</th><th>Project area</th><th>Project sdate</th><th>Project
edate</th></tr>");
                int temp=0;
                while(rs.next())
                {
                    temp=1;
                    pw.println("<tr><td>" +rs.getInt(1) +"</td><td>"
+rs.getString(2)+"</td><td>" +rs.getString(3) +"</td><td>" +rs.getString(4)
+"</td><td>" +rs.getString(5) +"</td><td>" +rs.getString(6) +"</td><td>"
+rs.getString(7) +"</td><td>" +rs.getString(8) +"</td><td>" +rs.getString(9)
+"</td></tr>");
                }

                pw.println("</table>");
                pw.println("<br><br>");
                pw.println("<a
href='http://localhost:8080/servlet/projreport'>BACK</a></center></body></html
>");
                if(temp==0)
                    pw.println("NO DATA FOUND");

            }catch(SQLException ee1) {pw.println(ee1);}
        }
    }
}

```

12.Campus Interview in Java Servlet:

```
import javax.servlet.http.*;
import javax.servlet.*;
import java.sql.*;
import java.io.*;

public class campinter extends HttpServlet
{
    Connection c;
    public void doPost(HttpServletRequest re,HttpServletResponse rs) throws
ServletException,IOException
    {
        rs.setContentType("text/html");
        PrintWriter out=rs.getWriter();
        String rcode=re.getParameter("t1");
        String name=re.getParameter("t2");
        String cname=re.getParameter("t3");
        String caddress=re.getParameter("t4");
        String date=re.getParameter("date");
        String mon=re.getParameter("mon");
        String year=re.getParameter("year");
        String perfor=re.getParameter("t6");
        String parea=re.getParameter("t7");
        String aot=re.getParameter("t8");
        String refby=re.getParameter("t9");

        String doi=date+"/"+mon+"/"+year;
        try
        {
            Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            c=DriverManager.getConnection("jdbc:odbc:hrms");
            Statement s=c.createStatement();
            ResultSet r;
            r=s.executeQuery("select reccode from campus");
```

```

int temp=0;
while(r.next())
{
    if(rcode.equals(r.getString(1)))
    {
        temp=1;
        break;
    }
}
if(temp==1)
{
    out.println("<html><body bgcolor=skyblue><br><br><br><center>");
    out.println("<h3>Recruitment Code Already Exist, Please Try Again</h3>");
    out.println("<br><br><br><a
href='http://localhost:8080/servlet/recruitment'>Back</a>");
    out.println("</center></body></html>");

}
else
{
    String str="insert into campus
values('"+rcode+"','"+name+"','"+cname+"','"+caddress+"','"+doi+"','"+perfor+"','"+
+parea+"','"+aot+"','"+refby+"')";
    //out.println(str);
    if(perfor.equals("Very Good") || perfor.equals("Good"))
    {
        s.executeUpdate(str);
        out.println("<html><body
bgcolor=skyblue><br><br><br><br><center>");
        out.println("<h3>DETAILS ARE STORED</h3>");
        // out.println("<a href='http://localhost:8080/usermo.html'>Proceed
Further</a>");
        out.println("<br><br><br><a
href='http://localhost:8080/servlet/recruitment'>Back</a>");
        out.println("</center></body></html>");
    }
    else
    {
        rs.sendRedirect("http://localhost:8080/servlet/campinter1");
    }
}
catch(SQLException sqle)
{
    out.println(sqle);
}
catch(ClassNotFoundException cnfe)
{
    out.println(cnfe);
}
}

```

}

CONCLUSION

This HRMS Project should satisfy all the needs of the Human Resource Manager. The Administration work mainly on Interviews conducted by the consultant. And the project module give the information about the project details from the User and service and updating may be done for user's satisfaction. This Project has Online test for the Users. It is used to know the result quickly. So the is very short to know about their results. Who one is selected in online test that user only attend the direct interview quickly. HRManager should insert any values is easy and quickly. The HRM should maintain the Employee Details, Training Employee's Details, Project Details, Recruitment Process i.e., Direct interview, campus interview and absorbing trainee very easy and the time should be concession.

The few years before this Human Resource Management were maintained by manually. So they maintains their data very difficulty. And the work processes are very large and make tensions. So they need an easiest way to solve this problem. And Time is very long to select a Trainee, Employee, Project Trainee and Recruit Process.

The above problems should solve by this Human Resource Management System Project. This Project should fulfill all needs of the HRMS. The world is going fast to finish their Problems and needs. This Project should need for the Real Time Human Resource Management System in any Places.

BIBLIOGRAPHY

1. JAVA2 The Complete Reference - Herbert Schildt
2. An Introduction to HTML - J.Arnold
3. Mastering JavaScript - James Jaworski
4. HTML 3.2 UNLEASHED - John December and
Mark Ginsburg
5. JAVA handbook - TATA InfoTech
6. Software Engineering - Richard Fairley
7. System analysis and design - Elias. M.Award
8. DataBase System - Abraham Silberschartz,
Henry F.Korth and S.Sudarshan

CONTENTS

S.NO	TOPIC	PAGE NO
1.	SYNOPSIS	
2.	COMPANY PROFILE	
3.	ABOUT SOFTWARE	
	3.1. JAVA SERVLETS	
	3.2. JAVA SCRIPT	
	3.3. MS-ACCESS	
4.	SYSTEM CONFIGURATION	
5.	SYSTEM ANALYSIS	
	5.1. EXISTING SYSTEM	
	5.2. PROPOSED SYSTEM	
6.	SYSTEM DESIGN	
	6.1. INPUT DESIGN	
	6.2. OUTPUT DESIGN	
7.	DATABASE DESIGN	
	7.1. DATABASE TABLES	
8.	SYSTEM TESTING & IMPLEMENTATION	
	7.1. SYSTEM TESTING	
	7.2. SYSTEM IMPLEMENTATION	
9.	DATAFLOW DIAGRAMS	
10.	SCREEN REPORTS	

11.	CONCLUSION	
12.	BIBLIOGRAPHY	