

MASTERS PROJECT

CPSC-597A  
CPSC-597B



SUMMER - 2007

Prof. Gonhsin Liu

**FINAL REPORT**

**Submitted by**

**SRIKANTH VELAGAPUDI**

**(ID#0725692)**

# WORK FLOW MANAGEMENT

---

## ABSTRACT

The adherent need for an integrated network of a composite environment in an office has led to this Intranet project. The workflow in a company would rather flow from one division to another division in a department bottlenecked by human factors. This bottleneck would affect the net performance of the organization (viz., slow information flow across the company, ineffective explanation of resource etc). These constraints can be reduced using an automated flow of work. Across the organization this flow is maneuvered using the project named "Work Flow Management". The workflow automation is basically an Intranet based project. The evolution of workflow management consists of the automation of business procedures or "workflows" during which documents, information or tasks are passed from one participant to another in a way that is governed by rules or procedures. Workflow software products, like other software technologies have evolved from diverse origins. While some offerings have been developed as pure workflow software, many have evolved from image management systems, document management systems, relational or object database systems, and electronic mail systems. Vendors who have developed pure workflow offerings have invented terms and interfaces, while vendors who have evolved products from other technologies have often adapted terminology and interfaces. Each approach offers a variety of strengths from which a user can choose. Adding a standard based approach allows a user to combine these strengths in one infrastructure. The key benefits of workflow are improved efficiency, better process control, improved customer service, flexibility and business process improvement. The system also reduces the resources needed for processing the data. It has few disadvantages besides many advantages. The person may or may not check the message he has received. Some times the server may fail by which work cannot be assigned to the right person.

## INTRODUCTION

One of the tougher steps in the paper-processing world is waiting for a supporting document to arrive. For example, an application may be received but a medical or credit report must be ordered or received before the application can be considered. In a small organization, the application may be held on the desk of the person who will make the decision. In larger organizations, the problem is harder. Since the recipient cannot remember every case, or there may multiple pupil who could make the decision. The documents need to filed and each arriving document that could satisfy a requirement needs to be checked that something is not waiting for it.

The work flow management systems allow organizations to define and control various activities associated with a business process. In addition many management systems also allow a business the opportunity to measure and analyze the execution of the process so that continuous improvements can be made. Such improvements may be short term (re-allocation of tasks to better balance the workload at any point of time) or long term (example: redefining portions of the work flow process to avoid bottlenecks in the future). Most work flow systems also integrate with other systems used by the organization - document management systems, databases, email, office automation products, geographic information systems, production application, etc. This integration provides structure to a process that employee a number of otherwise independent systems. It can also provide a method (such as project folder) for organizing documents from diverse sources.

## Establish goals:

- There may be a variety of reasons to implement work management systems. Improvement of internal processes is the traditional first purpose. This might be to increase the capacity (the amount of business processed) without increasing the costs. It might be to improve the equality of service. Or it might be to reduce costs in an industry that is not growing rapidly. It is important to know what you are trying to achieve.
- Managing processes between organizations is another potential use of work flow - issuing and teaching the orders for products and services from a second company perhaps integrated with processes within the first company. Some of the new facilities for e-commerce have been established and integrated with work management tools.

Workflow software enables modern businesses automate work flow on day- to-day operations. It is an IT technology that uses electronic systems to manage and monitor business processes. It allows the flow of work between individuals and/or departments to be defined and tracked. Although documents are often used as a medium for transporting information in a work flow software systems, it is mostly associated with document management, content management and knowledge management where the work flow system is used to track the process of creating and reviewing and distributing documents. Workflow software helps business automate a range of business tasks and electronically route the right information to the right people at the right time. With the help of this system, users are notified of pending work and help managers route approvals through the system quickly. Workflow software supports workflow, which is a combination of states and transitions that make up a process. Workflow can also be set up to track issues, purchases and expenses.

With the automated work flow management systems

- Work does not get misplaced or stalled – expeditors are rarely required to recover from errors or mismanagement of the work.
- The managers can focus on staff and business issues like individual performance, optimal procedures and special cases rather than the routine assignment of tasks. The army of clerks is no longer required to deliver and track the work.
- The procedures are formally documented and followed exactly, ensuring that the work is performed in the way planned by management, meeting all business and regulatory requirements.
- The best person (machine) is assigned to do each case, the most important cases are assigned first. Users do not waste time choosing which item to work on perhaps procrastinating on important but difficult cases.
- Parallel processing where two or more tasks are performed concurrently is far more practical than in a traditional, manual workflow.

With the best person doing the most important work following the correct procedures, not only is the business conducted more effectively, but also costs are lowered and the service to the customers is generally better. With the work equitably distributed and confidence that are always working “RIGHT” thing, users are happier. Therefore, workflow is good for company customers and users.

## Existing System:

One of the major challenges many of the companies and organizations are facing is the total time consumed by the business program. In the existing system, the workflow will be from one division to another division in a department directly or indirectly. Assigning the work directly can clarify the doubts, if there are any and some of their views can also be shared which helps in completing the work easily. But the disadvantage with the existing system is that, when the work is to be distributed among the people and if any of the persons is not available then the work is to be assigned when he is present. Since the process is done manually, it is time consuming. As the work assigned in an organization is not proposed in a particular way, it ultimately leads to a very slow and less efficient process.

## Proposed System:

The proposed system is used to overcome the drawbacks of the existing system. In the proposed system, the workflow is computerized. By automating the work flows the cost savings and the efficiency of the organization has been increased. In this proposed system, even though the person is not available, the work can be assigned within no time, without waiting for him. This allows a person to know his work or to ask any queries without moving from his place. It also reduces the resources needed for processing the work.

The project entitled "Work Flow Management" deals with the way of assigning work to the employees to increase the efficiency and performance of the organization and helps to reach a respect position in the global world. The project mainly includes four modules to control the flow. These modules help us in analyzing the data systematically and also very quickly.

These modules are made depending on the functionality of an organization. They are :

- Administrator.
- AGM.

- Manager/Coordinator.
- Employee.

### **Administrator:**

The administrator has the authority overall employees. All the rights of addition, deletion, updation of employee details, etc, are with him. He can perform the following functions:

- Add the employee.
- Modify the employee.
- Delete the employee.
- Report generation.
  - Daily report.
  - Weekly report.
  - Monthly report.

### **Add the employee:**

All the employees working in the organization will be listed here. If any new employee has entered organization, he will be added to the database.

All the details about the employee will be stored. This one asks for the details of the employee such as employee name, his designation, under whom the employee is working, employee ID, etc. Then, it asks for the confirmation. Then this will be stored in the database.

### **Modify the employee:**

If the employee was promoted (from one department to other department and if any new employee has been appointed in that place) then all the details must be updated. To update these details, we need to modify all the existing details with the new details. This helps us to update or modify the details of the employee working in an organization.

## Delete the employee:

If any employee has been transferred or removed, all the details that belong to that employee must be deleted. That is, it helps to us to set the working status of the employee as 'not working', in the database.

## Report generation:

The main aim of developing this project is to check the status of the work whenever required even in the absence of an employee. The progress of the organization is to be checked regularly to overcome all the drawbacks and such that there is a chance for further development. Thus, report generation helps us in generating the reports that are required. These reports can be maintained in any one of the three ways:

Daily reports.

Weekly reports.

Monthly reports.

### Daily reports:

Here, the work progress will be maintained in a daily fashion. Every day the work details will be stored as reports. The details such as work assigned for the employee for that day, the amount of work completed, any new work assigned and so on.

### Weekly reports:

Here, all the reports will be stored once in a week. That is, all the work that is completed in that particular week will be maintained as a single report and will be stored. This consists of all the work done and the work left in the week. It also shows the details of employees and their performance in a week.

### Monthly reports:

This sub module shows the progress of the work in a one-month period. This report provides us with the works that have been given to the employees and the status of the work and also estimates the performance of the employees.

AGM:

Assign the work:

The work will be assigned to different employees in various fields. The work will be sent to the respective employee id. The employee can view the work only if he knows his password. This also consists of the details regarding the date on which work was assigned and the time of its completion.

Work Status:

AGM will view the details of the work (i.e., overall status of work) and the names of the employees to whom the work was assigned. This also helps us to understand the type of work assigned to the people under the organization.

Search:

This module helps to retrieve the data required by AGM. This includes the date and also the status of the work, which provides us to easily access the required data.

Date

Status

Date:

This gives the details of the work assigned to the employees on a particular date. This helps us to keep in view of the details of the work on all days.

Status:

This will display the details of the work and the progress of the work. This helps in knowing the position of the work whether it partly done, started or completed.

## **Manager/coordinator:**

### Assign the work:

The work will be assigned to different employees working under manager. The work will be sent to the respected employee id. The employee can view the work only if he knows his password. The module also consists of the details regarding the date on which work was assigned and the time of its completion.

### View the work:

This helps the manager to view the different kinds of work assigned to him by his superior. This also includes the details of work like date of assignment, date of its completion, etc.

### Status Update:

After the completion of his work, he needs to update the status of work. His superior checks this updated status.

### Search:

This helps to retrieve the data required by the manager. This includes the date and also the status of the work, which provides him to easily access the required data.

Date

Status

### Date:

This gives the details of the work assigned to the employees on a particular date. This helps us to keep in view of the details of the work on all days.

### Status:

This will display the details of the work and the progress of the work. This helps in knowing the position of the work whether it partly done, started or completed.

Employee:

View the work:

This helps the employee to view the different kinds of work assigned to him by his superior. This also includes the details of work like date of assignment, date of its completion, etc.

Status Update:

After the completion of his work, he needs to update the status of work. His superior checks this updated status.

# JAVA OVERVIEW

## History of Java:

Java is a general purpose, object-oriented programming language developed by Sun Microsystems of USA in 1991. Originally called Oak by James Gosling. Java was designed to solve the problem of connecting many household machines together. Then it was redesigned to work with cable TV.

When the World Wide Web became popular in 1994, Sun realized that java was the perfect programming language for the web.

The most striking feature of the language is that it is a *platform-neutral* language. Java is the first programming language that is not tied to any particular hardware or operating system. Programs developed in Java can be executed anywhere on any system.

## Java Strengths:

Java is an excellent programming language. For most programming it's better than older programming languages like C or C++.

**Productivity :** The top reason Java has become popular is because of the increased productivity of Java programmers. Java programmers have about better productivity of C/C++ programmers.

**GUI :** Java a good, portable library for the graphical user Interface (GUI). Most of the programming languages supply only a text mode interface.

**Internet:** Java lets you easily use the Internet. Most languages were designed before the Internet was born!

**Portability** : Java programs can run on many different machines (Intel, Sparc, PowerPC, ...) and many different operating systems (Windows, Unix, Macintosh,...) you can move a C program if it is written very carefully, but it is usually difficult or impossible. In contrast, it is easy to move most Java programs.

**Reliability:** Java programs are more reliable because Java doesn't have very dangerous things like C/C++'S pointer arithmetic. Java also checks array bounds and other error-prone operations. Memory management is much safer because Java does automatic garbage collection.

**Libraries:** Java has a very large number of Packages, which extend the language. Therefore it is unnecessary to call the operating system directly.

**OOP:** Object-Oriented programming features (Inheritance, Encapsulation and Polymorphism) make many programs, especially large programs, easier to write.

**Large programs:** Java supports large programming projects with Object-Oriented programming, Packages, and Components (Java Beans).

## JAVA Features:

Sun Microsystems describes Java with the following attributes.

**Compiled and interpreted:** Usually a computer language is either compiled or interpreted. Java combines both these approaches thus making Java a two-stage system. First, Java compiler translates source code into what is known as Byte-code instructions. Byte-code is not machine instructions and therefore, in the second stage, Java interpreter generates machine code that can be directly executed by the machine that is running the Java program.

**Platform-independent and portable:** The most significant contribution of Java over other languages is its portability. Java programs can be easily moved from one computer system to another, anywhere and anytime. Changes and upgrades in operating systems, processors and system resources will not force any changes in java programs.

Java ensures portability in two ways. First, Java compiler generates byte-code instructions that can be implemented on any machine. Secondly, the size of the primitive data types is machine-independent.

**Object-oriented:** Java is a true object-oriented language. Almost everything in Java is an *object*. All program code and data reside within objects and classes. Java comes with an extensive set of classes, arranged in Packages, that we can use in our programs by inheritance.

### **Main concepts of OOP:**

**Abstraction :** It is a basic element of object-oriented programming. Complexity can be managed through abstraction. “Abstraction refers to the act of representing essential features without including the background details or explanations”. Classes are concept of abstraction.

**Encapsulation:** The wrapping up of data and methods into a single unit called class is known as encapsulation. The data is not accessible to the outside world and only those methods, which are wrapped in class, can access it. This method of insulation is called “data hiding”.

**Inheritance:** It is the process by which objects of one class acquire the properties of objects of another class. Inheritance supports the concept of Hierarchical classification.

In OOP, inheritance provides the idea of “Reusability”.

**Polymorphism:** Polymorphism means ability to take more than one form. That is, an operation may exhibit different behavior in different instances. Polymorphism plays an important role in allowing objects having different internal structures to share the same external interface.

**Robust and Secure:** Java is a robust language. It has strict compile time and run time checking for data types. Java also incorporates the concept of Exception

handling which captures series errors and eliminates any risk of crashing the system.

Security becomes an important issue for a language that is used for programming on Internet. The absence of pointers in Java ensures that programs cannot gain access to memory locations without proper authorization.

**Distributed:** Java is designed as a distributed language for creating applications on networks. It has ability to share both data and programs. Java applications can open and access remote objects on Internet as easily as they can do in a local system. This enables multiple programmers at multiple remote locations to collaborate and work together on a single project.

**Simple, Small and Familiar:** Java is small and simple language. Many features of C and C++ that are either redundant or sources of unreliable code are not part of Java. Familiarity is another striking feature of Java. To make the language look familiar to the existing programmers, it was modeled on C and C++ languages. Therefore Java is a simplified version of C++.

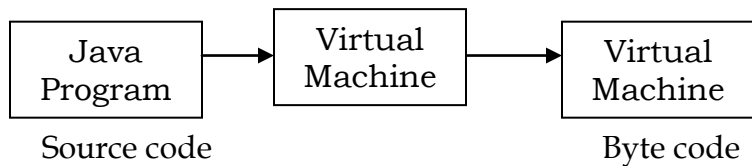
**Multithreaded and Interactive:** Multithreaded means handling multiple tasks simultaneously. Java supports multithreaded programs. This means that we need not wait for the application to finish one task before beginning another. This feature greatly improves interactive performance of graphical applications.

**High Performance:** Java performance is impressive for an interpreted language, mainly due to the use of intermediate byte-code. The incorporation of multithreading enhances the overall execution speed of java programs.

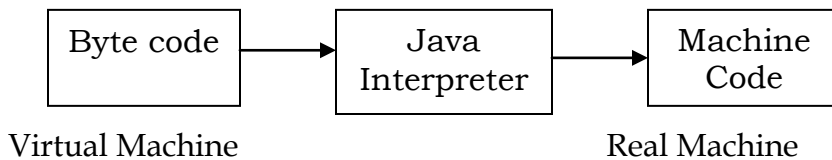
**Dynamic and Extensible:** Java is a dynamic language. Java is capable of dynamically linking in new class libraries, methods and objects. Java can also determine the type of class through a query, making it possible to either dynamically link or abort the program, depending on the response.

## Java virtual machine (JVM):

All language compilers translate source code into machine code for a specific computer. Java compiler does the same thing. Java compiler produces an intermediate code known as byte-code for a machine that doesn't exist. This machine is called *Java virtual machine* and it exists only inside the computer memory. The process of compiling a java program into byte-code is also referred as *virtual machine code*.



Java Virtual Machine code is not machine specific. Java Interpreter generates the machine specific code by acting as an intermediary between the virtual machine and the real machine.



## Java Weaknesses:

Java is arguably the best overall programming languages, but there are problems with it.

### Lack of few features:

Java lacks a few features that some C++ programmers find useful, e.g., enums, macros, operator overloading, generic types (templates), etc. Many of the "features" of C++ that are not in Java have been removed to make the language better (e.g., pointers and Multiple Inheritance).

## **JDBC**

### The JAVA Programming Language and JDBC

A JAVA program, written according to specification, can run on any JAVA technology - enabled platform without recompilation. The JAVA programming language is completely specified and, by definition a JAVA technology - enabled platform must support a known a core of libraries. One such library is the java's package or JDBC, which you can think of as a portable version ODBC, and is itself a major standard. Using the JAVA programming language in a conjunction with JDBC provides truly portable solution to writing database applications.

### **JDBC DRIVER**

JDBC Driver is a class that implements the JDBC Driver interfaces and understands how to convert a program (and typically SQL) requests for a particular database. Clearly, the driver is what makes it all work.

### JDBC Application Programming Interface

The JDBC API is broken into 2 parts: the core API and the JDBC optional package. In general, the JDBC core API adds a few more classes to its previous version, but is primarily concerned with performance, class enhancements and functionality, and the new SQL3 (also known as SQL-99) data types.

The functionality in the core API includes scrollable result sets, batch updates, programmatic inserts, deletes and updates, performance hints, character streams for streams of internationalized Unicode characters, full precision for java. Math decimal value and support for time zone in date, time and timestamp values.

## Connecting to the Database

There are always 2 steps to making a database connection using the driver manager:

### 1. Load the JDBC Driver:

You must load a Driver that enables the JDBC to communicate with a Data source. Here is the standard method for dynamically loading a driver:

```
class.forName(DriverClassName);
```

A Standard JDBC compliant Driver should also create a new instance of the driver class with this code. Unfortunately, in practice this doesn't work for all the cases. For this reason, the following code is used:

```
class.forName(DriverClassName).newInstance();
```

### 2. Connect to a Data Source:

The driver supplies methods to make a connection, but requires a specific type of URL, which uses the JDBC protocol. The generalized form is

```
Jdbc:<subprotocol>:<subname>
```

Using the driver manager class, you request a connection using the passed URL and the Driver Manager selects the appropriate driver.

Here is the standard form of the connection request:

```
Connection con=DriverManager.getConnection(URL,username>Password);
```

This form is best for probability even in cases where user name and password are empty string (" ") due to a database default or, say text files acting as ODBC data sources, which cannot make use of such attributes.

### Creating a Table:

While the connection class has a number of capabilities in order to use DDL or Data Manipulation Language (DML) SQL statements, a statement object required. So, the next step is to ask the connection for a statement object:

```
Statement stmt=con.createStatement();
```

## **Connecting a JAVA Program to a Database:**

A connection object represents and controls a connection to a database. While everything in JDBC depends on the capabilities of the database and the JDBC driver, in general you can have multiple connections to the same database and/or connections to multiple databases. The driver manager class handles driver registration and provides methods for obtaining a connection.

One of the first step in obtaining a connection is often the most important: how to setup that database URL?.

Jdbc:<subprotocol>:<subname> , with the <subprotocol>: identifying the machine or server and <subname> essentially identifying the database .In practice ,the content depends on the specific driver and can be bewildering ,ranking along with classpath problems in producing "no suitable drivers" errors. This is fairly straight forward, primarily because the client and the server run on the same machine.

Most DBMS engines that support remote (and even local) connections do so using a TCP/IP port. Like any other socket program, the DBMS engine is free to decide what port it wants to use. While TCP/IP is generally the norm, other communication protocols may be used. Db2, for example, can also use APPC(Advanced Program to Program Communication) on several platforms.

When applications attempt to connect to a network or Internet server, identification/location information must be provided. The general JDBC way is to use //host:port/subname, where host is an IP address or DNS(Domain Name Service) or other locatable name.

## **Statements, Resultsets and Interacting with a Database:**

A statement object is a container or transport mechanism to send/execute (normally) SQL statements and retrieve any results via its associated connection. In areas controlled by the Connection Interface, there are three types of statements, including Prepared Statements and Callable Statements, both of which are sub interfaces of statement. You don't need to create new instance of statement, but

instead, request the associated Connection to create one:

```
Statement stmt=con.createStatement();
```

The execute series are the most often used of Statement's methods:

- **executeQuery()** is used to execute SQL statements that result a single **ResultSet**.
- **executeQuery()** is used for statements that return a **ResultSet** ,basically a SELECT statement.
- **executeUpdate()** is used to execute SQL statements that modify a table or values of columns in a table and return the number of rows affected(which is zero in the case of DDL statements).
- **executeUpdate()** returns an int containing the row count for INSERT,UPDATE or DELETE statements ,or zero for SQL statements that do not return anything ,like DDL statements.
- **execute()** can be used to execute any type of SQL statement ,but is intend for those that can return multiple results or values.

To allow the most flexibility to work with various databases and data sources, JDBC places no restriction on the kinds of SQL statements that statement can send.

A statement only keeps one **ResultSet** open at a time and often reuses the same **ResultSet** for new data. You should be sure to get all the data required from the **ResultSet** before executing another query via its associated statement.

A statement should automatically close() the **ResultSet** on execution and on Statement.close(),but you may want to close the **ResultSet** yourself as soon as its data is no longer needed.

#### Data navigation:

**ResultSet.next()** returns a Boolean: 'True', if there a next row and 'False ' ,if not(the end of data).Conceptually, a pointer or cursor is positioned just before the first row when the **ResultSet** is obtained .Invoking **next()** moves to the first row ,then the second row and so on.

```
if(result.next())
```

The if-statement collects the data. After that, a loop while(result.next()) is used, to allow the program to continue to the end of the data.

### Data Extraction:

Once positioned at a row, the application can get the data on a column-by-column basis using the appropriate **ResultSet.getXXX** method.

### Sample Servlet Collecting And Retrieving Data From The Database

```
Import java.sql.*;
Class stdList
{
    Public Static Void Main(String args[]) throws SQLException
    {
        DriverManager.registerDriver(new Oracle.jdbc.driver.OracleDriver());
        Connection conn =
DriverManager.getConnection("jdbc:thin@hc.ap.nic.in:1521:test","scott","tiger");
        Statement stmt = conn.createStatement();
        ResultSet rset =stmt.executeQuery("Select adv_name from advir where
adv_code=(\"Select law1 from main where mtype=? and mno = ? and myear = ?\")");
        While(rset.next())
        {
            System.out.println(rset.getString(1));
            System.out.println(" ");
            System.out.println(rset.getString(2));
            System.out.println(" ");
            System.out.println(rset.getString(3));
        }
    }
}
```

## Session Tracking:

The servlet API provides several methods and classes specifically designed to handle session tracking on behalf of servlets. Every user of site is associated with a `javax.servlet.http. HTTP session object` that servlets can use to store or retrieve information about that user. A servlet uses its request objects `getSession()` method to retrieve the current HTTP session object:

Syntax: `public HttpSession HttpSession request.`

`getSession(boolean create)`

This method returns the current session associated with the user making the request. If the user has no current value session, this method creates one if `create` is true or returns null if `create` is false. To ensure the session is properly maintained, this method must be called at least once before any output is returned to the response.

### **getAttribute() method:**

**Syntax:**

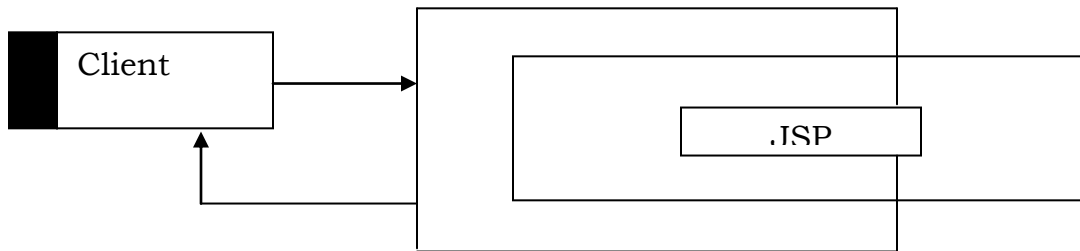
`Void getAttribute(String name)`

Retrieves the value of the specified object is stored in the servlet context.

# Java Server Pages

Java Server Pages(JSP) technology provides an easy way to create dynamic web pages and simplify the task of building web applications that work with a wide variety of web servers, application servers, browsers and development tools. This article provides an overview of JSP from a developers perspective and includes examples of JSP in action.

JSP technology allows web developers and designers to easily develop and maintain dynamic web pages that leverage existing business systems. As part of the java technology family JSP enables independent. JSP separates user interfaces from content generation, enabling designers to change the overall page layout without altering the underlying dynamic content. You normally give your file a.jsp extension, and typically install it in any place.



You could place a normal web page although what you write often looks more like a regular HTML file than a servlet, behind the screens, the JSP page just gets converted to a normal servlet with the static HTML simply being printed to the output stream associated with the servlet's service method. This is normally done the first time the page is requested, and developers can request the page themselves when first installing it if they want to be sure that the first real user does not get a momentary delay when the JSP page is translated to a servlet and the servlet is compiled and loaded. Note also that many web servers let you define aliases that so that a URL that appears to reference an HTML file really points to a servlet or JSP page. Aside from the regular HTML, there are three main types of JSP constructors that you embed in a page

- Script elements
- Directives
- Actions

Scripting elements let you specify java code that will become part of the resultant servlet, directives let you control the overall structure of the servlet and actions let you specify existing components that should be used and otherwise control the behavior of the JSP engine. A JSP page is simply an HTML web page that contains additional bits of code that execute application logic to generate dynamic content. This application logic to generate dynamic content. This application logic may involve JavaBeans JDBC objects, Enterprise Java Beans(EJB) and Remote Method Invocation(RMI) objects all which can be easily accessed from a JSP page. For example, a JSP page method call to a JDBC object that access a database; when the page is displayed in a users browser, it will contain both the static HTML content and dynamic information retrieved from the database.

The separation of user interface and program logic in a JSP page allows for a very convenient delegation of tasks between web content authors and developers. It also allows developers to create flexible code that can easily updated and reused. Because JSP pages are automatically compiled as needed, web authors can make changes to presentation code without recompiling application logic. This makes JSP more flexible method of generating dynamic web contents that Java Servlets, whose functionality Java Server Pages extend.

### **JSP Scripting Elements:**

JSP scripting elements let you insert Java code into the Servlet that will be generated from the current JSP page. There are three forms

1. Expressions of the form `<%= expression%>` that are evaluated and inserted into the output.
2. Scriptlets of the form `<%code%>` that are inserted into the servlets service method.

3. Declarations of the form `<%code%>` that are inserted into the body of the servlet class outside of any existing methods.

### **JSP Expressions:**

A JSP Expression is used to insert Java values directly into the output. It has the following form:

```
<%=Java Expression%>
```

The Java Expression is evaluated, converted to a string, and inserted in the page. This evaluation is performed at run time (when the page is requested) and thus has full access to information about the request. For example, the following shows the date/time that the page was requested:

```
Current time: <%=new Java.Util.Date() %>
```

A JSP declaration lets you define methods or fields that get inserted into the main body of the Servlet Class (Outside of the service method processing the request). It has the following form:

```
<% !Java code%>
```

### **JSP Directives :**

A JSP Directive affects the overall structure of the Servlet Class. It usually has the following form:

```
<@directive attribute= "value" %>
```

However, you can also combine multiple attribute settings for a single directive as follows:

```
<@directive attribute1= "value1" attribute2= "value2" attributeN= "valueN" %>
```

There are two main types of Directive: Page which lets you do things import classes, customize the Servlet superclass and the like; and Include which lets you insert a file into the Servlet class at the time the JSP file is translated into a Servlet. The specification also mentions taglib directive, which is not supported in JSP version 1.0 but is intended to let JSP authors define their own tags. It is expected that this will be the main contribution of JSP 1.1.

**Predefined Variables :**

To simplify code in JSP expressions and scriptlets, you are supplied with eight automatically defined variables, sometimes called implicit objects. The available variables are request, response, out, session, application, config, page content and page. They are :

- Request
- Response
- Out
- Session
- Application
- Config
- Actions

## Hypertext Markup Language

HTML is an application of ISO standard 8879, SGML (Standard Generalized Markup Language) but specialized to hypertext and adapted to the web. HTML is a markup language, i.e., a language describing how documents ought to be formatted. This is enabled by a set of elements, which makeup the document and informs the browser about the action to be taken when a certain element is specified. It's a versatile language and can be used on any platform or desktop.

The HTML tags model the appearance of a web page. Usage of graphics, fonts, different colors, etc can enhance the presentation of the document. It also has a provision for the creation of hypertext links or hyperlinks to other documents or portions of the same document.

HTML supports two types of tags i.e., separating tags and surrounding tags. Separating tags are placed between the text elements to which they apply. Ex: <BR> is used to insert a line break. Surrounding tags consists of

**Hyperlinks:** They play a major role in HTML. Hyperlink is a link which when clicked takes you to a screen where more information about that link will be available. Hyperlink can even more point to a remote document, which resides in a different directory. When the movement of the mouse looks like a palm with a finger pointing upwards, it implies that it is a hyperlink. Syntax is <a href="filename">Text to be displayed</a> where <href> is a hypertext references browser has to take you soon after the user slides on the hyperlink. Text to be displayed is the text that appears on the screen as the hotspot

**Forms:** Forms are the most interesting ones that have been added in the recent times to web publishing. With the advent of these features, the web changed from being a publishing medium with hyperlinks to fully interactive environment with the potential for being something entirely new. The following are the form elements, which are described one by one

```
<form method="post" action="do something">
```

---

```
</form>
```

You can have multiple forms within a document, but these forms cannot be nested.

**Method:** An attribute of the form tag, indicating the method with which the form input is given to the script that processes the form. Possible values are Get, Post.

**Action:** An attribute of the form tag indicating the script to process the form input. Contains a relatively path or URL to the script.

**Type:** An attribute of the form tag, indicating the script to process the form input. Possible values are submitted, reset, text, radio, checkbox and hidden.

Submit creates a button, which resets default values of the form if any.

Reset creates a button, which creates default values of the button.

Text creates a single-line text field.

Checkboxes creates a form element that is not visible in the screen but has a name and value that can be presented onto the script that processes the form input.

**HTTP:** Hypertext transfer protocol is a stateless TCP/IP based protocol used for communicating in the world wide web. HTTP defines the precise manner in which web clients communicate with web servers. HTTP/1.0 is the most common version in use today.

# ORACLE OVERVIEW

## **Database Management System:**

A collection of programs that enables you to store, modify and extract information from a database. There are many different types of DBMS's ranging from small systems that run on personal computers to huge computers that run on mainframes.

The set of rules for constructing queries are known as a *query language*. Different DBMS support different Query languages although there is a semi-standardized query language called SQL (*Structured query language*). Sophisticated languages for managing database system are called fourth-generation languages.

## **Oracle:**

Oracle Corporation is largest software company whose primary business are database products. Historically, oracle has tagged high-end workstation and main computer as a server platform to its database system. Its relation database was the first to support the SQL language, which has since been industry standard. Along with Sun Microsystems oracle has been one of the leading champion of network computers.

## **RDBMS**

Relation Database Management system is a type of database management system (DBMS) that stores data in the form of related tables. Relational database are powerfully because they require only few assumptions about how data is related or how it will be extracted from the database. As a result the same database can viewed in many different ways.

An important feature of relational system is that a single system can spread across several tables. This differ from flat-file database, in each database is self-contained in a single table.

## Normalization

1. In relational database design, the process of organizing data to minimize redundancy. Normalization usually involves dividing a database into two or more tables and defining relations among tables. The objective is to isolate data. So, addition, deletion and modification of a field can be made in just one table and they propagated through rest of the database via defined relations.

There are three main normal forms each with increasing level of normalization.

- First Normal Form (1NF) Each field in table contains different information.
- Second Normal Form (2NF) Each field in table that is not determine of the contents of another field must it self be function of the other fields in the table.
- Third Normal Form (3NF) No duplicate information is permitted.

There are additional normalization levels, such as Boyce Codd Normal Form (BCNF), fourth normal form (4NF) and fifth normal form (5NF). While normalization makes databases more efficient to maintain, they can also make them more complex because data is separated into so many different tables.

- In Data processing, a process applied to all data in a set that produces a specific statistical property. For example, each expenditure to produce a percentage.
- In programming, changing the format of a floating-point number so the left-most digit in the mantissa is not a zero.

# REQUIREMENT ANALYSIS

## SOFTWARE SPECIFICATIONS:

Operating System : Windows NT/2000  
Language Used : JSP, JDBC  
Version : JSP2.0  
Type Of Server : Weblogic-7.0  
Database : Oracle9i  
Browser : Internet Explorer/ Netscape Navigator

## HARDWARE SPECIFICATIONS:

### SERVER SIDE:

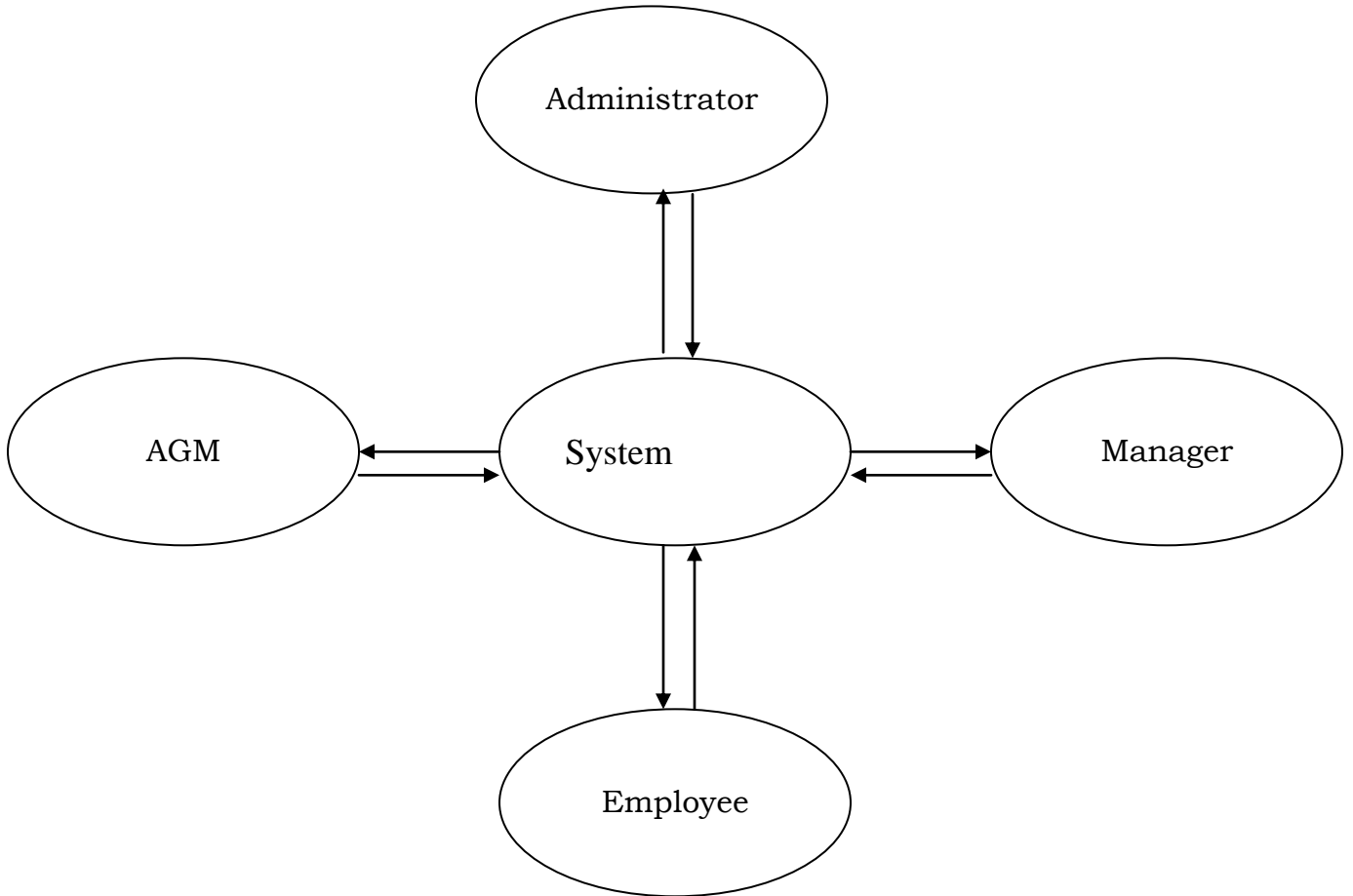
Processor : Pentium3  
RAM : 128(preferable 256)  
Hard-Disk : 2GB  
Keyboard : 104 keys

### CLIENT SIDE:

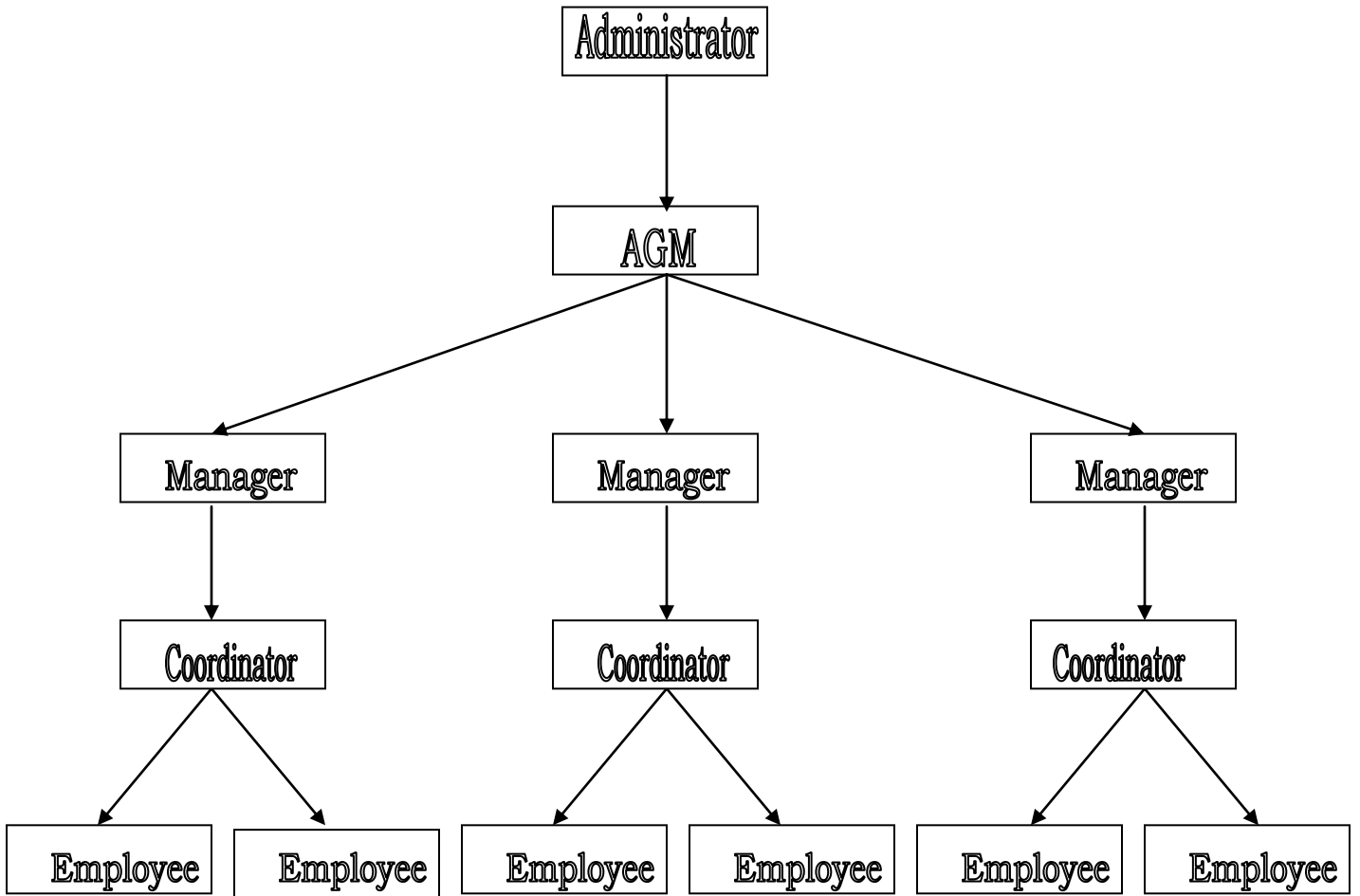
Processor : Pentium3  
RAM : 256  
Hard-Disk : 40GB  
Keyboard : 104 keys

# Design

Context Flow Diagram

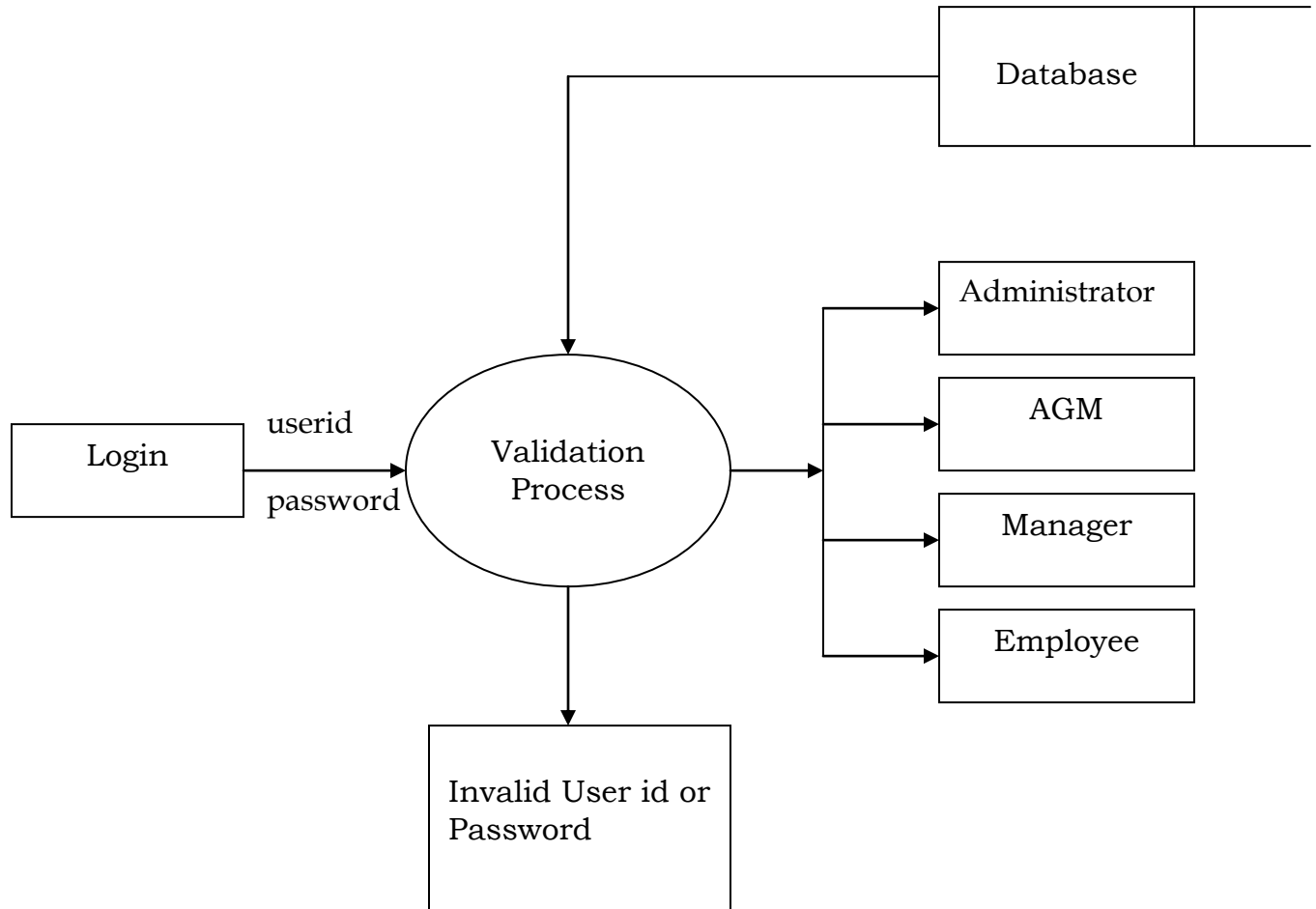


# Hierarchy Diagram



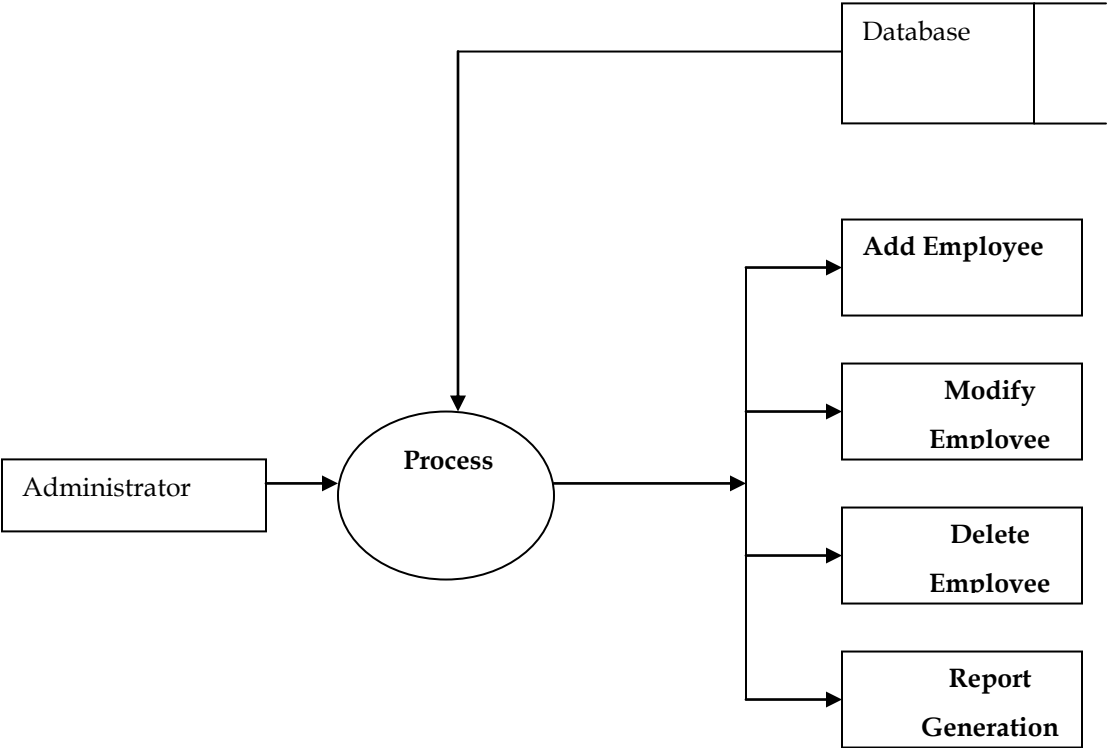
# LEVEL 1 DATA FLOW DIAGRAM

## Login

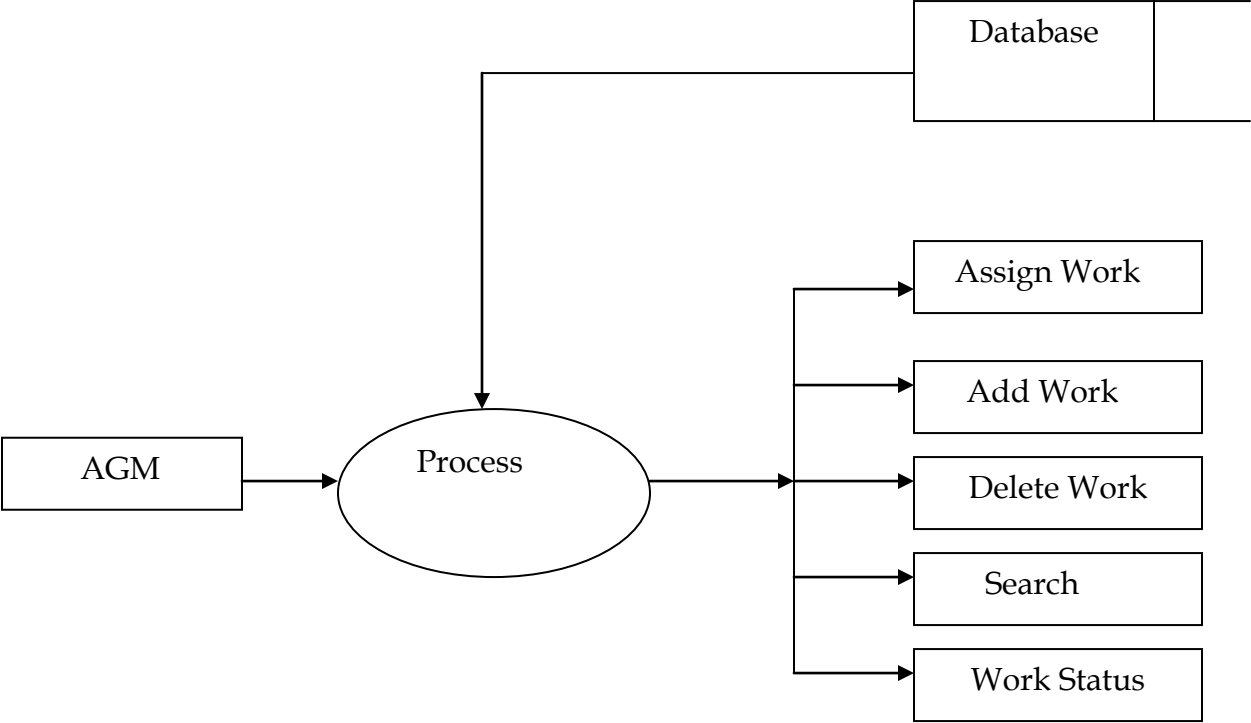


LEVEL 2 DATA FLOW DIAGRAM

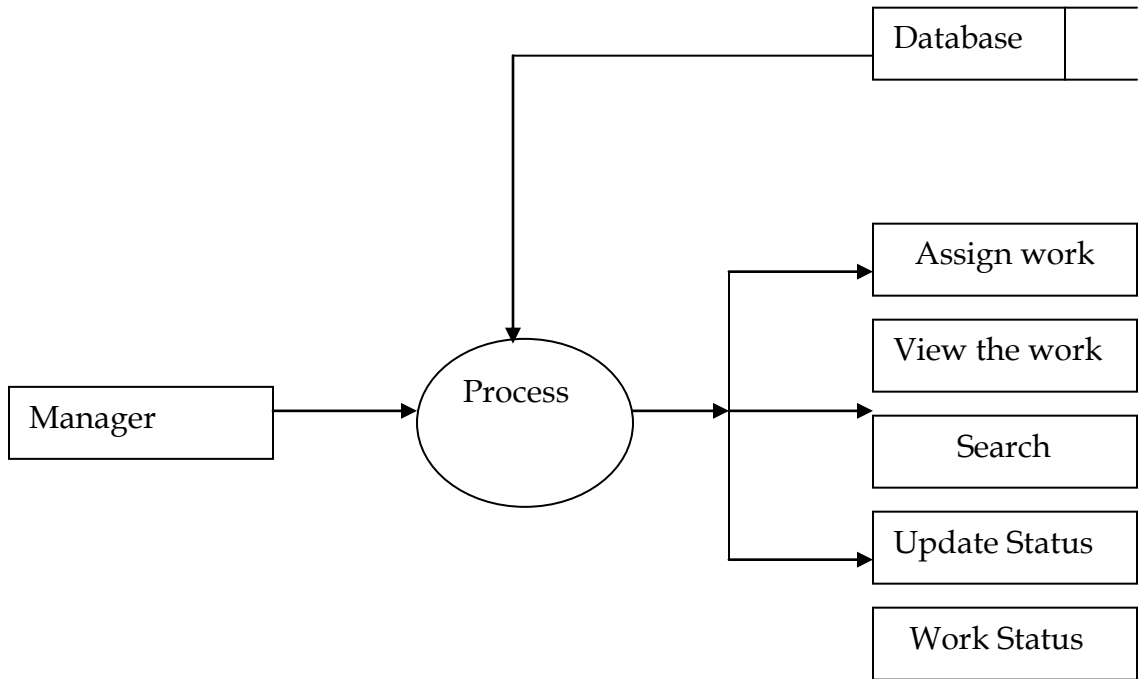
Administrator



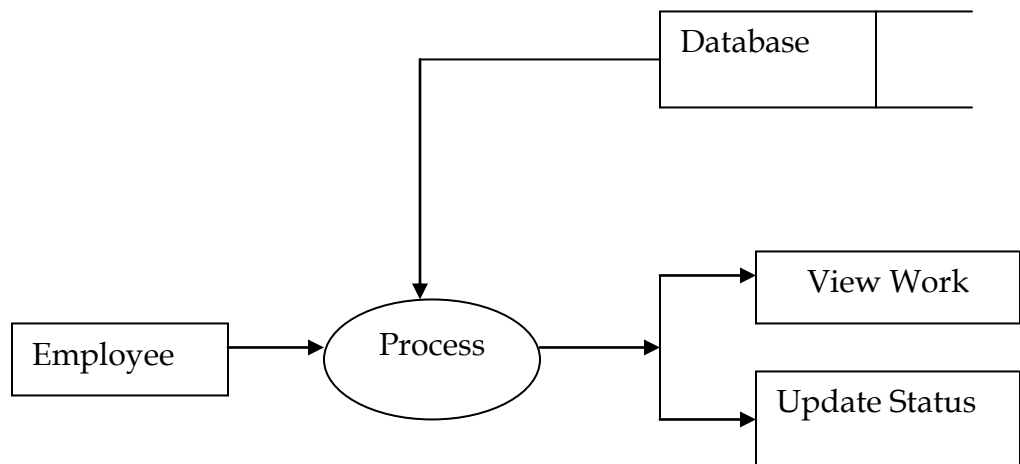
AGM



## Manager/Coordinator

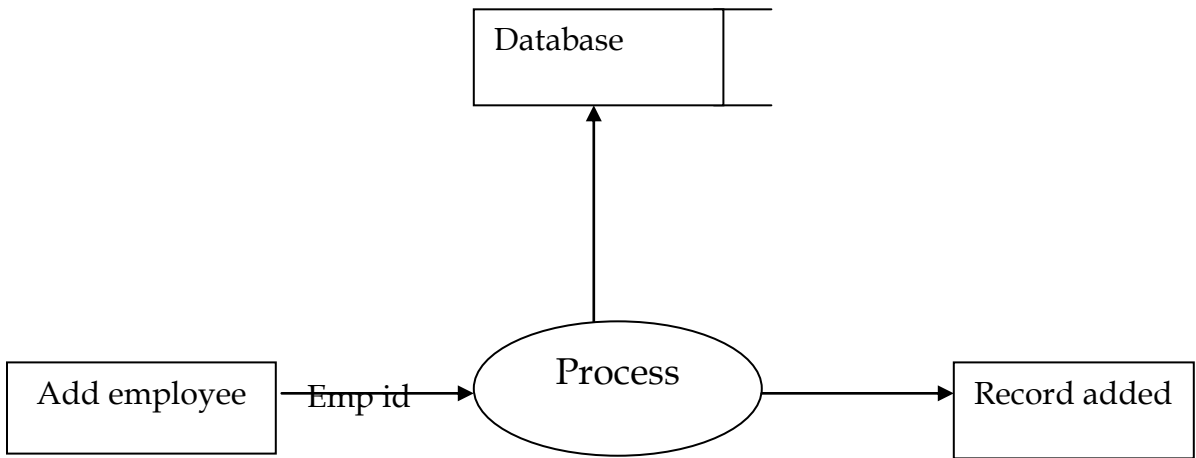


## Employee

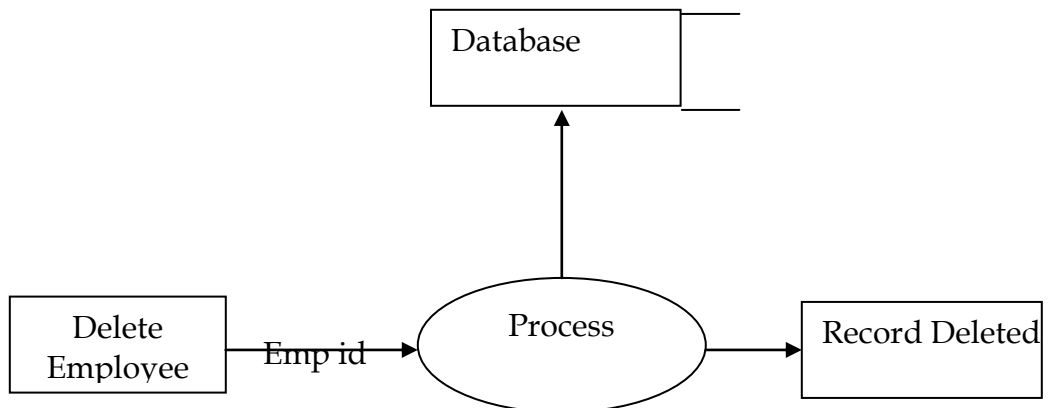


Level 3 DFD's

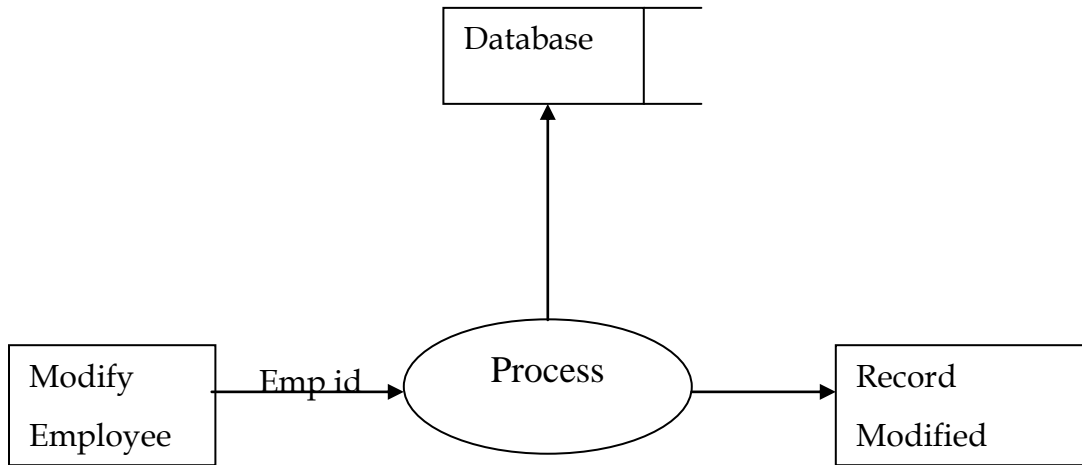
Add Employee



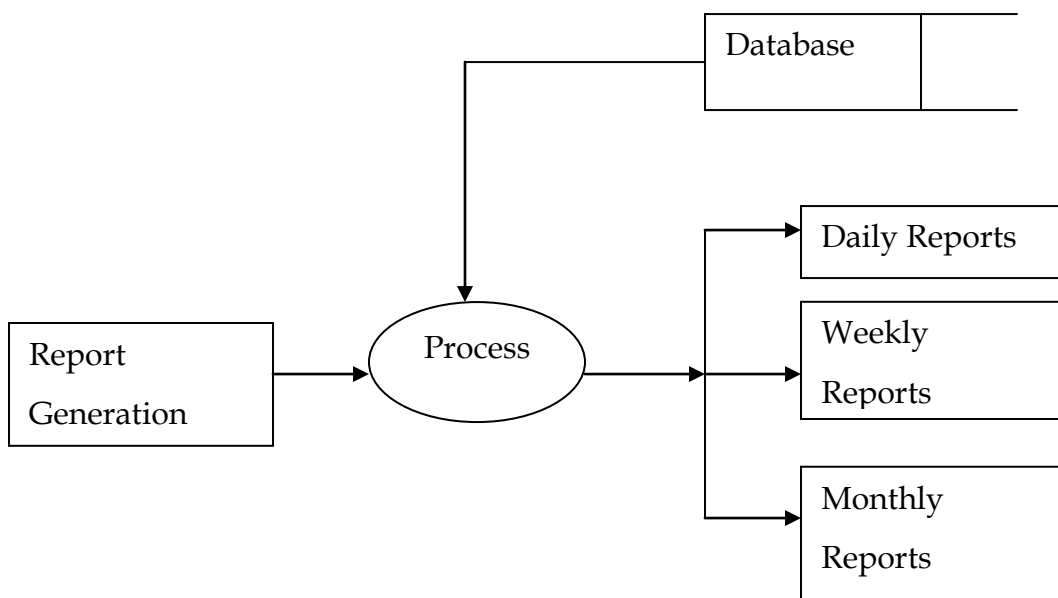
Delete Employee



## Modify Employee

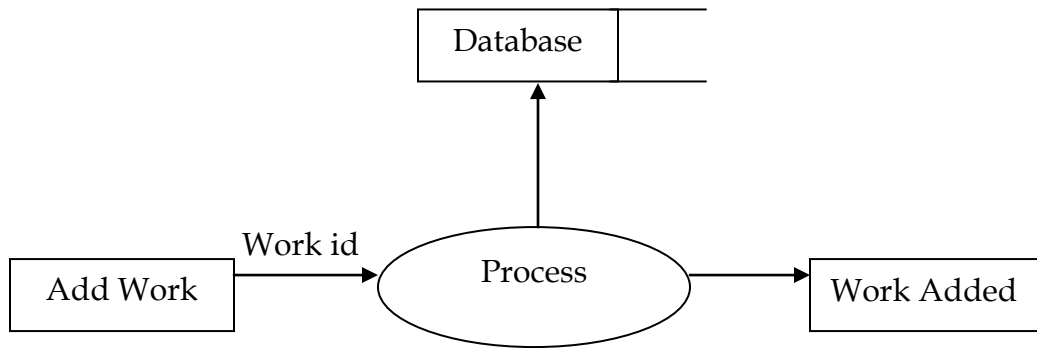


## Report Generation

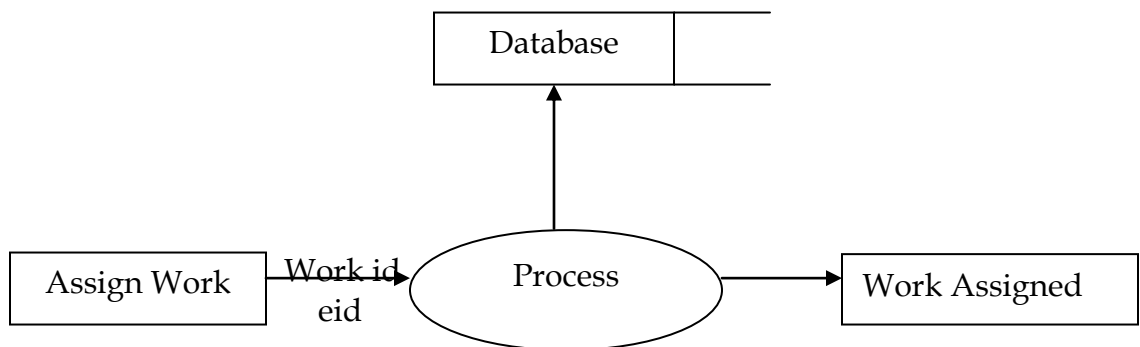


## Level 3 Data Flow Diagrams

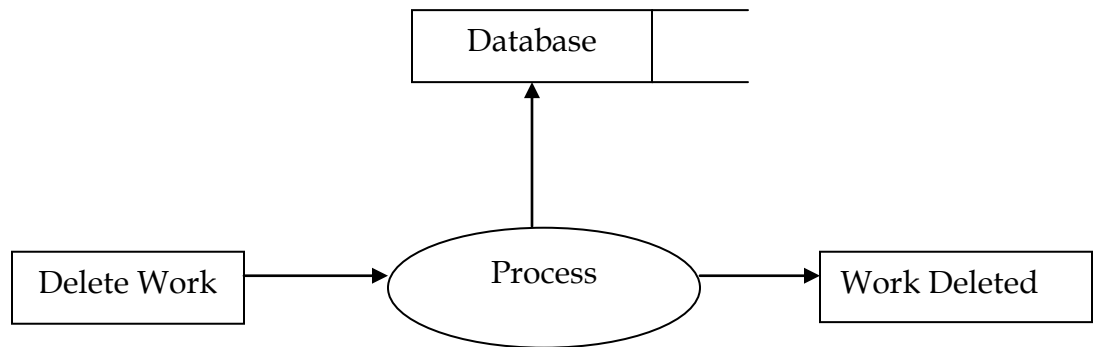
### Add Work



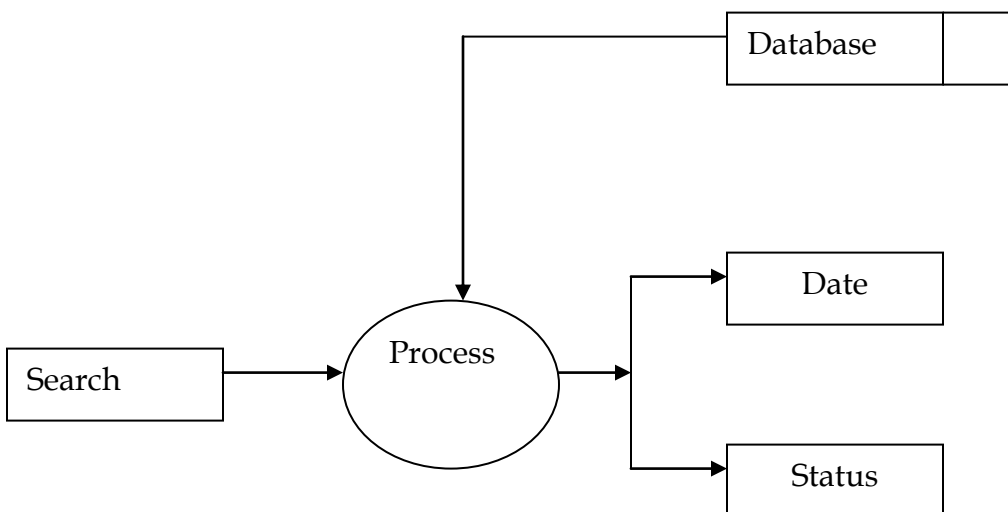
### Assign Work



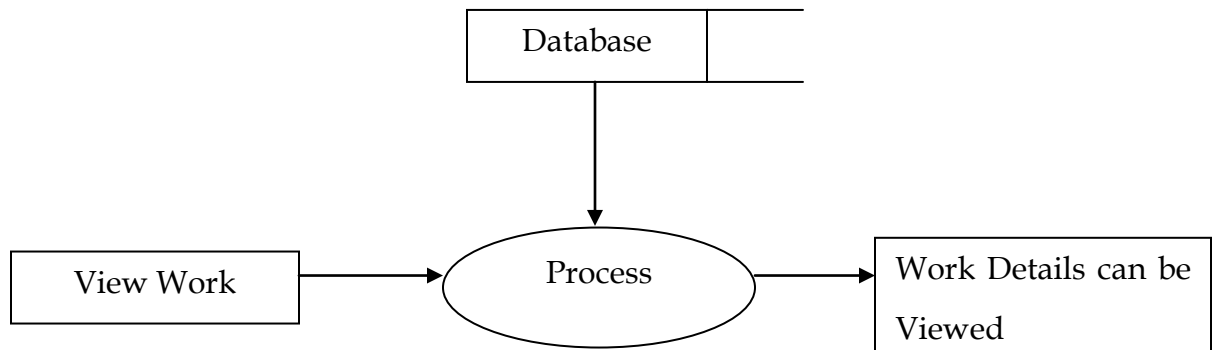
## Delete Work



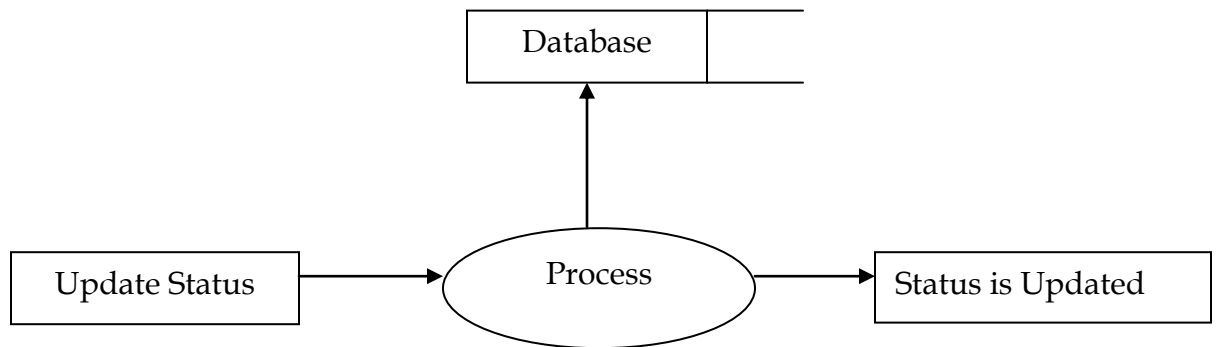
## Search



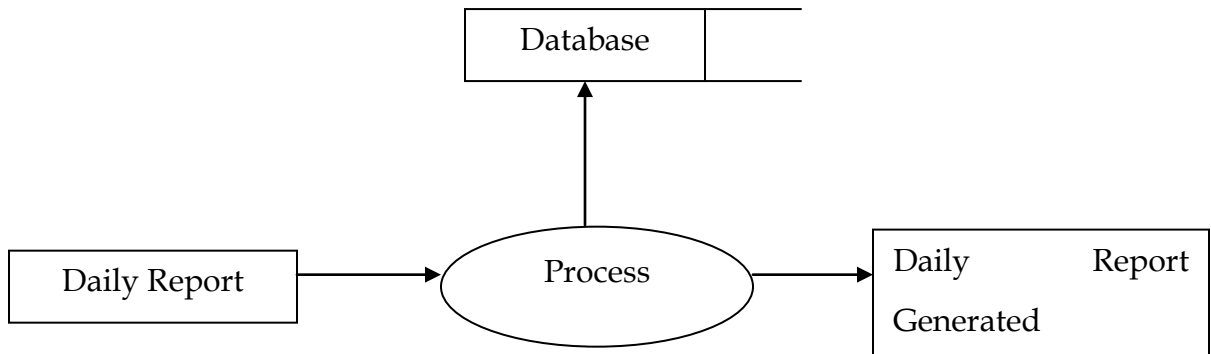
## View Work



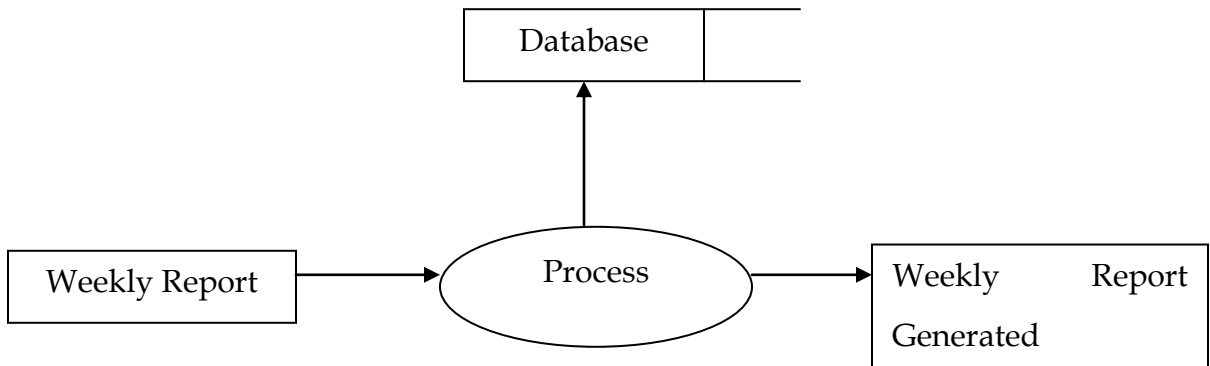
## Update Status



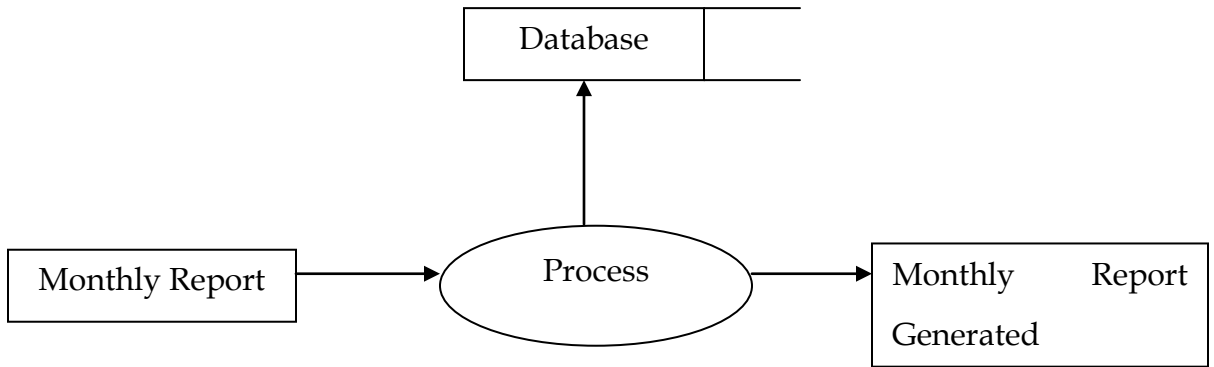
## Daily Reports



## Weekly Reports

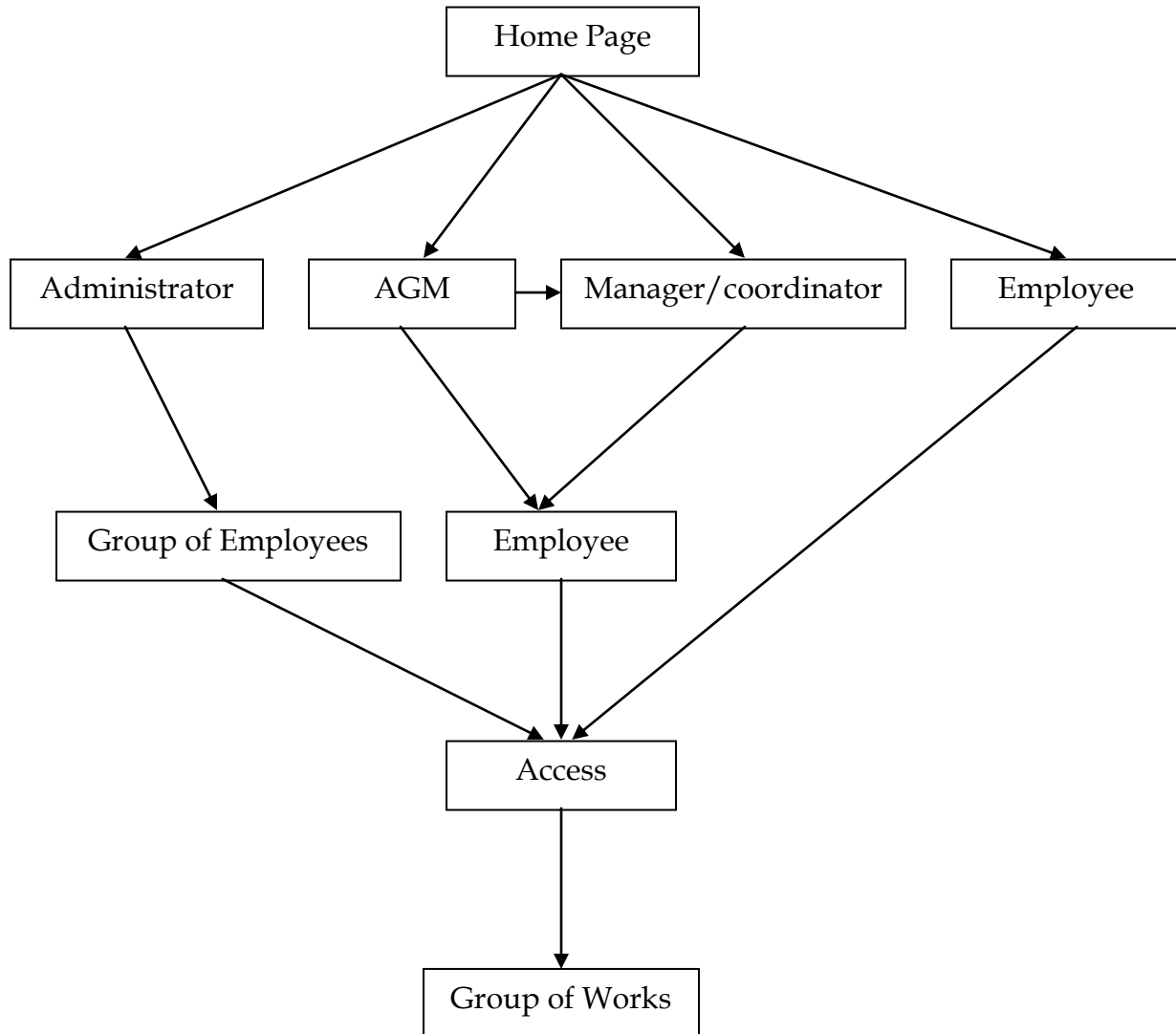


## Monthly Reports

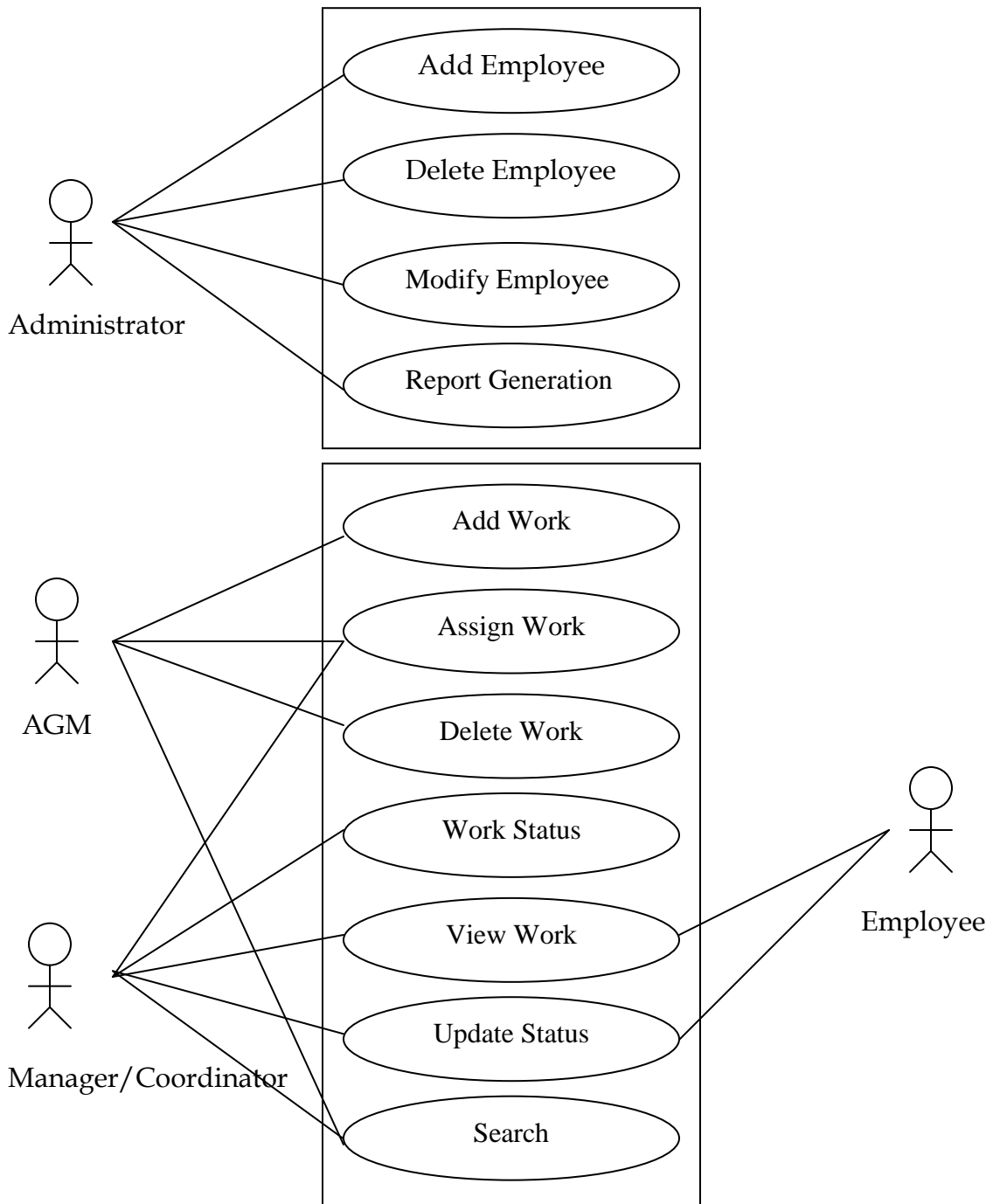


# UML DIAGRAMS

Draft class Diagram:



# Use Case Diagram:



## Database Tables

### e1\_details :

Name	Null?	Type
EMPID	NOT NULL	VARCHAR2(5)
ENAME		VARCHAR2(12)
EMP_ADD		VARCHAR2(25)
EMP_PNO		VARCHAR2(10)
W_STATUS		CHAR(1)
WORKSUNDER		VARCHAR2(15)
DESIGNATION		VARCHAR2(15)

### w1\_details :

Name	Null?	Type
WORKID	NOT NULL	VARCHAR2(5)
WORKNAME		VARCHAR2(15)
DURATION		VARCHAR2(3)
STARTDATE		VARCHAR2(11)
MONTH		NUMBER(2)
DAY		NUMBER(2)
YEAR		NUMBER(4)
S		VARCHAR2(12)

### work1\_status:

Name	Null?	Type
WORKID		VARCHAR2(10)
WORK_DONE_BY		VARCHAR2(10)
WORK_ASSIGNED_BY		VARCHAR2(10)
WORK_STATUS		VARCHAR2(10)

w1\_report:

Name	Null?	Type
WORKID		NUMBER(4)
WORK_DONE_BY		NUMBER(4)
WORKSTATUS		VARCHAR2(15)
DAY		NUMBER(2)
MONTH		NUMBER(2)
YEAR		NUMBER(4)

login\_details:

Name	Null?	Type
EMPID		VARCHAR2(15)
PASSWORD		VARCHAR2(15)
DESIGNATION		VARCHAR2(15)

## Implementation:

### Framepage.html

```
<frameset rows=15%,70%,* border=0>
<frame name=f1 src="headpage.html" scrolling=no>
<frameset cols=35%,10%,* border=0>
<frame name=f2 src="definition.html">
<frame name=f3>
<frame name=f4 src="userpage.html">
</frameset>
<frame name=f5 src="grpmem.html" scrolling=no>
</frameset>
```

### headpage.html

```
<html>
<body>
<font face="Bookantiqua" size=6 color="maroon">
<center>
Work Flow Management
</center>
</font>
</body>
</html>
```

### Definition.html

```
<html>
<body>
<font face="bookantiqua" color="maroon" size=4>
<align=left><br><br><br>
Definition:
<br>
<font color="black">
<i>
Work Flow Management systems allow users to define and control various activities
associated with the business process.
</i>
</font>
</body>
</html>
```

## Userpage.html

```
<html>
<script language="javascript">
function foc()
{
userpage.user.focus();
}
function check()
{
    var x=userpage.user.value;
    var y=userpage.pwd.value;

    if(x=="")
    {
        alert("Username must not be empty")
    }
    else if(y==""){
    alert("Password must not be empty")}
    else{

        userpage.action="LoginCheck.jsp"}

}
</script>
<body bgcolor="lightblue" onload="foc()">
<form name="userpage" onsubmit="check()" onreset="foc()" target=_top>
<table rows=1 cols=1 border=0 align=right bgcolor="lightblue">
<th><b><font face="bookantiqua"
color="maroon"><h4>LoginHere</h4></font></b></th>
<tr><td><pre>
<font face="bookantiqua" color="black"><h5>
User id <input type="text" name="user">

Password <input type="password" name="pwd">

&nbsp; &nbsp; &nbsp; &nbsp;<input type="submit" name="submit" value="submit" >
<input type="reset" name="reset" value="reset">
</h5>
</font>
</pre>
</td></tr>
</table>
```

### grpmem.html:

```
<html>
<body>
<font face="bookantiqua" color="maroon">
Group Members:<br>
<font color="black">
</font>
</body>
</html>
```

### Logincheck.jsp:

```
<html>
<body>
<%@ page import="java.sql.*" %>
<%
    int userid=Integer.parseInt(request.getParameter("user"));
    String password=request.getParameter("pwd");
    String user="" +userid;
    session.setAttribute("userid",user);
    try{
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        Connection con=DriverManager.getConnection("Jdbc:odbc:dsn1","scott","tiger");
        Statement st=con.createStatement();
        ResultSet rs=st.executeQuery("Select * from login_details where
empid="+userid);
        if(!rs.next())
            out.println("hello");
        String str1=rs.getString(2);
        String str2=rs.getString(3);
        if(password.equals(str1))
        {
            if(str2.equals("agm"))
            {
                response.sendRedirect("agmframe.html");
            }

            else if(str2.equals("administrator"))
            {
                response.sendRedirect("admin.html");
            }
            else if(str2.equals("manager") || str2.equals("coordinator"))
            {
```

```

        response.sendRedirect("manager.jsp");
    }
    else if(str2.equals("employee"))
    {
        response.sendRedirect("emp.jsp");
    }
    }
    else
    {
        out.println("Password is Invalid");
        response.sendRedirect("framepage.html");
    }
    }
    catch(Exception e)
    {
        out.println(e);
    }
}
%>
</body>
</html>

```

#### **admin.html:**

```

<frameset cols=25%,* border=0 scrolling=no noresize>
<frame name="menu" src="admin_menu.html">
<frame name="targetpage">
</frameset>

```

#### **admin\_menu.html:**

```

<HTML>
<BODY>
<FONT color="maroon" face="bookantiqua">
    <H4><center>Administrator Menu</center></H4></FONT></br>
    <A href="addemp.html" target="targetpage">Add Employee</A> <br>
    <A href="delete.jsp" target="targetpage">Delete Employee</A> <br>
    <A href="mod1emp.jsp" target="targetpage">Modify Employee</A> <br>
    <A href="report.jsp" target=_top>Report Generation</A> <br>
    <A href="framepage.html" target=_top>Log Out</A> <br>
</FONT></BODY></HTML>

```

#### **addemp.html:**

```

<html>
<script language="javascript">
function foc()

```

```

{
addemp.empid.focus();
}
function check()
{
    if(addemp.empid.value=="")
    {
        alert("Empid must not be null")
        addemp.empid.focus();
    }
    else if(addemp.ename.value=="")
    {
        alert("Employee name must not be null")
        addemp.ename.focus();
    }
    else if(addemp.designation.value=="")
    {
        alert("designation must not be null")
        addemp.designation.focus();
    }
    else if(addemp.mgr.value=="")
    {
        alert("Works under(empid) not be null")
        addemp.mgr.focus();
    }
    else if(addemp.add.value=="")
    {
        alert("Address must not be null")
        addemp.add.focus();
    }
    else if(8<addemp.pno.value>10 )
    {
        alert("Phone no. should be between 8-10 digits")
        addemp.pno.focus();
    }
    else
    {
        addemp.submit();
    }
}
}
</script>

```

```

<body onload="foc()">
<form name="addemp" action="add.jsp">
<font face="bookantiqua" color="maroon" >
<h4><center>Employee Details to store in DataBase</center></h4></font>
<pre>
Empid      : <input type="text" name="empid">

```

Employee Name : <input type=text Name="ename">

Designation : <select NAME="designation">  
<option value="agm">AGM</option>  
<option value="mgr">MANAGER</option>  
<option value="cdr">COORDINATOR</option>  
<option value="emp">EMPLOYEE</option>  
</select>

Works Under : <input type=text name="mgr">

Address : <textarea rows=7 cols=15 name="add"> </textarea>

Phone No : <input type=text name="pno">

```
<input type="button" value="Add" onclick="check()"> <input type="reset"
onclick="foc()">
</pre>
</form>
</body>
</html>
```

### add.jsp:

```
<html>
<body>
<%@ page import="java.sql.*" %>
<%
int empid=Integer.parseInt(request.getParameter("empid"));
String ename=request.getParameter("ename");
String designation=request.getParameter("designation");
int worksunder=Integer.parseInt(request.getParameter("mgr"));
String add=request.getParameter("add");
String pno=request.getParameter("pno");
try{
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
    Connection con=DriverManager.getConnection("Jdbc:odbc:dsn1","scott","tiger");
    Statement st=con.createStatement();
    st.executeUpdate("insert into e1_details
(empid,ename,designation,worksunder,emp_add,emp_pno) values
("+empid+", '"+ename+"', '"+designation+"', '"+worksunder+'', '"+add+'', '"+pno+'')");
    con.commit();
    out.println("New employee is added ");
}
catch(Exception e)
{
```

```

        out.println(e);
    }
%>
</body>
</html>

```

### delete.jsp:

```

<html>
<head><h3><center>
<font color="maroon">
DELETE
</font>
</center></h3></head>
<body>
<form name="delete" action="del_data.jsp">
<%@ page import="java.sql.*" %>
<%
    try{
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        Connection
con=DriverManager.getConnection("Jdbc:odbc:dsn1","scott","tiger");
        Statement st=con.createStatement();
        ResultSet rs=st.executeQuery("select empid,ename from e1_details
where w_status='w'");
        %>
        <font color="maroon">
        <pre>
Working Employees list    <select name="emp">
</pre>
        </font>
        <% while(rs.next())
        {
        %>
            <option><%= rs.getString(1) %></option>
        <%
        con.commit();
        }
        }
        catch(Exception e)
        {
            out.println(e);
        }
        %>
        </td>
        </tr>
        <tr>

```

```

        <td>
        <input type="submit" value="Delete">
        </td>
    </tr>
</table>
</body>
</html>

```

### del\_data.jsp:

```

<html>
<head><h3><center>DELETE </center></h3></head>
<body>
<%@ page import="java.sql.*" %>
<%
        int empid=Integer.parseInt(request.getParameter("emp"));
        try{
            Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            Connection
con=DriverManager.getConnection("Jdbc:odbc:dsn1","scott","tiger");
            Statement st=con.createStatement();
            st.executeQuery("select designation,worksunder from e1_details where
empid='"+empid+"'");
            ResultSet rs=st.getResultSet();
            rs.next();
            String designation=rs.getString(1);
            String worksunder=rs.getString(2);

            if(designation.equals("agm") || designation.equals("manager") || designation.
equals("coordinator"))
            {
                %>
                u r going to delete <b><%= designation %></b>
                <br>
                The employes working under this <%= designation %> will work
under <%= worksunder %>
                <%
                    st.executeUpdate("update e1_details set w_status='n' where
empid='"+empid+"'");
                    st.executeUpdate("Update e1_details set worksunder='"+worksunder+"
where worksunder='"+empid+"'");
                }
                else
                {
                    st.executeUpdate("update e1_details set w_status='n' where
empid='"+empid+"'");
                }
            }
        }
    %>

```

```

        <b>You have sucessfully deleted the employee</b>
    <%
        }
        con.commit();
        }
        catch(Exception e)
        {
            out.println(e);
        }
    %>
</body>
</html>

```

### mod1emp.jsp:

```

<html>
<head><h3><center>
<font color="maroon">
Modify
</font>
</center></h3></head>
<body>
<form name="mod1emp" action="mod3emp.jsp">
<%@ page import="java.sql.*" %>
<%
    try{
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        Connection
con=DriverManager.getConnection("Jdbc:odbc:dsn1","scott","tiger");
        Statement st=con.createStatement();

        ResultSet rs=st.executeQuery("select empid from e1_details where
w_status='w'");
        %>
        <font color="maroon">
        <pre>
Working Employees list    <select name="emp"><br><br>
        </pre>
        </font>
        <%while(rs.next())
        {
            int val=rs.getInt(1);
            %>
            <option ><%=val%></option>
            <%
                session.setAttribute("ei",String.valueOf(val));
                con.commit();
            %>
        }
    }
    catch(Exception e)
    {
        out.println(e);
    }
%>
</body>
</html>

```

```

    }
    }
    catch(Exception e)
    {
        out.println(e);
    }
%>
<tr>
<td>
<input type="submit" value="Modify">
</td>
</tr>
</table>
</body>
</html>

```

### **mod3emp.jsp:**

```

<html>
<body>
<form name="mod3emp" action="update.jsp">
<font face="bookantiqua" color="maroon" >
<h4><center>Employee Details to modify in DataBase</center></h4></font>
<%@ page import="java.sql.*" %>
<%
        int empid=Integer.parseInt(request.getParameter("emp"));
        try{
            Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            Connection
con=DriverManager.getConnection("Jdbc:odbc:dsn1","scott","tiger");
            Statement st=con.createStatement();
            ResultSet rs=st.executeQuery("select * from e1_details where
empid="+empid);
            rs.next();
%>
        <pre>
Employee Name  : <input type=text Name="ename"
value="<%=rs.getString("ename")%>">

Designation   : <input type=text name="designation"
value="<%=rs.getString("designation")%>">

Works Under   : <input type=text name="worksunder"
value="<%=rs.getString("worksunder")%>">

Address       : <input type=text name="add"
value="<%=rs.getString("emp_add")%>">

```

```

Phone No      : <input type=text name="pno"
value="<%=rs.getString("emp_pno")%>">
</pre>
<%
        }
        catch(Exception e)
        {
        out.println(e);
        }
        %>
    <input type="submit" value="Update"> <input type="reset" name="reset">
</form>
</body>
</html>

```

### update.jsp:

```

<%@ page import="java.sql.*"%>
<%
Object s1=session.getAttribute("ei");
String ei=(String)s1;
int empid=Integer.parseInt(ei);
String a1=request.getParameter("ename");
String a2=request.getParameter("designation");
String a3=request.getParameter("worksunder");
String a4=request.getParameter("add");
String a5=request.getParameter("pno");
try{
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        Connection
con=DriverManager.getConnection("Jdbc:odbc:dsn1","scott","tiger");
        Statement st=con.createStatement();
        st.executeUpdate("update e1_details set
ename='"+a1+"',designation='"+a2+"',worksunder='"+a3+"',emp_add='"+a4+"',emp_
pno='"+a5+"'where empid="+empid);
        out.println("You have sucessfully modified the employee");
        con.commit();
        }
        catch(Exception e)
        {
                out.println(e);
        }
        %>

```

### report.jsp:

```

<frameset cols=25%,* border=0 scrolling=no noresize>
<frame name="menu" src="report_menu.jsp">
<frame name="targetpage">
</frameset>

```

### report\_menu.jsp:

```

<HTML>
<BODY>
<TABLE align=left border=0 cols=1 rows="8">
  <TBODY>
    <TR>
      <TH><B><FONT color="maroon" face="bookantiqua">
        <H4>REPORT MENU</H4></FONT></B></TH>
    <TR>
      <TD><A href="daily.jsp" target="targetpage">Daily Reports </A>
      </TD></TR>
    <TR>
      <TD><A href="weekly.jsp" target="targetpage">Weekly Reports</A>
      </TD></TR>
    <TR>
      <TD><A href="monthly.jsp" target="targetpage">Monthly Reports</A>
      </TD></TR>
    <TR>
      <TD><A href="framepage.html" target="_top">Log Out</A> </TD></TR>
  </TBODY></TABLE></BODY></HTML>

```

### daily.jsp:

```

<html>
<head><center><h3><b> DAILY REPORT
<body>
<form name="daily" action="daily_reports.jsp">
  <table bgcolor="lightblue" cellpadding=10 cellspacing=5>
    <tr>
      <td>Select Date here </td>
      <td>day</td>
      <td><select name="day" >
        <option value="1">01</option>
        <option value="2">02</option>
        <option value="3">03</option>
        <option value="4">04</option>
        <option value="5">05</option>
        <option value="6">06</option>
        <option value="7">07</option>

```

```

        <option value="8">08</option>
        <option value="9">09</option>
        <option value="10">10</option>
        <option value="11">11</option>
        <option value="12">12</option>
        <option value="13">13</option>
        <option value="14">14</option>
        <option value="15">15</option>
        <option value="16">16</option>
        <option value="17">17</option>
        <option value="18">18</option>
        <option value="19">19</option>
        <option value="20">20</option>
        <option value="21">21</option>
        <option value="22">22</option>
        <option value="23">23</option>
        <option value="24">24</option>
        <option value="25">25</option>
        <option value="26">26</option>
        <option value="27">27</option>
        <option value="28">28</option>
        <option value="29">29</option>
        <option value="30">30</option>
        <option value="31">31</option>
    </td>
</select>
<td>month</td>
<td><select name="month">
    <option value="1">01</option>
    <option value="2">02</option>
    <option value="3">03</option>
    <option value="4">04</option>
    <option value="5">05</option>
    <option value="6">06</option>
    <option value="7">07</option>
    <option value="8">08</option>
    <option value="9">09</option>
    <option value="10">10</option>
    <option value="11">11</option>
    <option value="12">12</option>
</select>
</td>
<td>Year</td>
<td><select name="year">
    <option value="2003">2003</option>
    <option value="2004">2004</option>
    <option value="2005">2005</option>
    <option value="2006">2006</option>
</td>

```

```

        </tr>
        <tr>
        <td><input type="submit" value="Report">
        </td>
        </tr>
    </table>
</body>
<html>

```

### daily\_reports.jsp:

```

<html>
<head><center><h3><b> DAILY REPORT</b></center></h3>
<body>
<table>
<tr>
<td>
<h3><b>On Going works</b></h3>
        <table bgcolor="lightblue" border=1 align="left">
        <tr><td>Work ID</td>
        <td> Work Name</td>
        <td>Work Done By</td>
        </tr>
<%@ page import="java.sql.*" %>
<%
    int day=Integer.parseInt(request.getParameter("day"));
    int month=Integer.parseInt(request.getParameter("month"));
    int year=Integer.parseInt(request.getParameter("year"));
    try{
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        Connection
con=DriverManager.getConnection("Jdbc:odbc:dsn1","scott","tiger");
        Statement st=con.createStatement();
st.executeQuery("select r.workid,w.workname,r.work_done_by from w1_report
r,w1_details w where r.workstatus='ongoing' and r.day="+day+" and
r.month="+month+" and r.year="+year+" and r.workid=w.workid");
        ResultSet rs=st.getResultSet();
        while(rs.next())
        {
%>
        <tr><td> <%= rs.getString(1) %></td>
        <td> <%= rs.getString(2) %></td>
        <td> <%= rs.getString(3) %></td>
        <%
        }
        %>
        </tr>
        </table>

```

```

</td>
</tr>
<tr>
<td>
<h3><b>Pending Works</b></h3>
<table bgcolor="lightblue" border=1 align="left">
<tr><td>Work ID</td>
<td> Work Name</td>
<td>Work Done By</td>
</tr>
<%
st.executeQuery("select r.workid,w.workname,r.work_done_by from w1_report
r,w1_details w where r.workstatus='pending' and r.day="+day+" and
r.month="+month+" and r.year="+year+" and r.workid=w.workid");
rs=st.getResultSet();
while(rs.next())
{
%>
<tr><td> <%= rs.getString(1) %></td>
<td> <%= rs.getString(2) %></td>
<td> <%= rs.getString(3) %></td>
<%
}
%>
</tr>
</table>
</td>
</tr>
<tr>
<td>
<h3><b>Completed Works</b></h3>
<table bgcolor="lightblue" border=1 align="left">
<tr><td>Work ID</td>
<td> Work Name</td>
<td>Work Done By</td>
</tr>
<%
st.executeQuery("select r.workid,w.workname,r.work_done_by from w1_report
r,w1_details w where r.workstatus='completed' and r.day="+day+" and
r.month="+month+" and r.year="+year+" and r.workid=w.workid");
rs=st.getResultSet();
while(rs.next())
{
%>
<tr><td> <%= rs.getString(1) %></td>
<td> <%= rs.getString(2) %></td>
<td> <%= rs.getString(3) %></td>
<%
}

```

```

        %>
    </tr>
</table>
<%
    }
    catch(Exception e)
    {
        out.println(e);
    }
    %>
</td></tr>
</table>
</body>
</html>

```

### weekly.jsp:

```

<html>
<head><center><h3><b> WEEKLY REPORT
</body>
<form action="weekly_report.jsp">
<table bgcolor="lightblue" cellpadding=10 cellspacing=5>
<tr>
<td>
Enter date from (dd-mon-yy)
</td>
<td>
<input type="text" name="datefrom">
</td>
</tr>
<tr>
<td>
Enter date to (dd-mon-yy)
</td>
<td>
<input type="text" name="dateto">
</td>
</tr>
<tr>
<td>
<input type="submit" value="Report">
</td>
</tr>
</table>
</form>
</body>
</head>

```

```
</html>
```

### weekly\_report.jsp:

```
<html>
<head><center><h3><b> WEEKLY REPORT</b></center></h3>
<body>
<table>
<tr>
<td>
<h3><b>On Going works</b></h3>
      <table bgcolor="lightblue" border=1 align="left">
        <tr><td>Work ID</td>
          <td> Work Name</td>
          <td>Work Done By</td>
        </tr>
<%@ page import="java.sql.*" %>
<%
    String fromdate=request.getParameter("datefrom");
    String todate=request.getParameter("dateto");
    try{
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        Connection
con=DriverManager.getConnection("Jdbc:odbc:dsn1","scott","tiger");
        Statement st=con.createStatement();
        String sq="select r.workid,w.workname,r.work_done_by from
w1_report r,w1_details w where r.workstatus='ongoing' and r.workid=w.workid
and w.startdate between '"+fromdate+"' and '"+todate+"'";
        ResultSet rs=st.executeQuery(sq);
rs.next();
while(rs.next())
    {
%>
        <tr><td> <%= rs.getString(1) %></td>
          <td> <%= rs.getString(2) %></td>
          <td> <%= rs.getString(3) %></td>
        <%
        }
        %>
        </tr>
        </table>
        </td>
        </tr>
        <tr>
        <td>
        <h3><b>Pending Works</b></h3>
        <table bgcolor="lightblue" border=1 align="left">
```

```

        <tr><td>Work ID</td>
        <td> Work Name</td>
        <td>Work Done By</td>
    </tr>
    <%
st.executeQuery("select r.workid,w.workname,r.work_done_by from w1_report
r,w1_details w where r.workstatus='pending' and r.workid=w.workid and
w.startdate between '"+fromdate+"' and '"+todate+"'");
    rs=st.getResultSet();
    while(rs.next())
    {
%>
        <tr><td> <%= rs.getString(1) %></td>
        <td> <%= rs.getString(2) %></td>
        <td> <%= rs.getString(3) %></td>
        <%
        }
    %>
    </tr>
</table>
</td>
</tr>
<tr>
<td>
<h3><b>Completed Works</b></h3>
<table bgcolor="lightblue" border=1 align="left">
<tr><td>Work ID</td>
<td> Work Name</td>
<td>Work Done By</td>
</tr>
<%
st.executeQuery("select r.workid,w.workname,r.work_done_by from w1_report
r,w1_details w where r.workstatus='completed' and r.workid=w.workid and
w.startdate between '"+fromdate+"' and '"+todate+"'");
    rs=st.getResultSet();
    while(rs.next())
    {
%>
        <tr><td> <%= rs.getString(1) %></td>
        <td> <%= rs.getString(2) %></td>
        <td> <%= rs.getString(3) %></td>
        <%
        }
    %>
    </tr>
</table>
<%
    }

```

```

catch(Exception e)
{
    out.println(e);
}
%>

```

```

</td></tr>
</table>
</body>
<html>

```

### monthly.jsp:

```

<html>
<head><center><h3><b> MONTHLY REPORT
<body>
<form action="month_report.jsp">
    <table bgcolor="lightblue" cellpadding=10 cellspacing=5>
    <tr>
    <td>Select month here </td>
    <td>month</td>
    <td><select name="month" >
        <option value="1">01</option>
        <option value="2">02</option>
        <option value="3">03</option>
        <option value="4">04</option>
        <option value="5">05</option>
        <option value="6">06</option>
        <option value="7">07</option>
        <option value="8">08</option>
        <option value="9">09</option>
        <option value="10">10</option>
        <option value="11">11</option>
        <option value="12">12</option>
    </td>
    </select>
    <td>Year</td>
    <td><select name="year">
        <option value="2002">2002</option>
        <option value="2003">2003</option>
        <option value="2004">2004</option>
        <option value="2005">2005</option>
        <option value="2006">2006</option>
    </select>
    </td>
    </tr>

```

```

<tr>
<td>
<input type="submit" value="Report">
</td>
</tr>
</form>
</body>
</html>

```

### month\_report.jsp:

```

<html>
<head><center><h3><b> MONTHLY REPORT</b></center></h3>
<body>
<table>
<tr>
<td>
<h3><b>On Going works</b></h3>
<table bgcolor="lightblue" border=1 align="left">
<tr><td>Work ID</td>
<td> Work Name</td>
<td>Work Done By</td>
</tr>
<%@ page import="java.sql.*" %>
<%
    int month=Integer.parseInt(request.getParameter("month"));
    int year=Integer.parseInt(request.getParameter("year"));
    try{
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        Connection
con=DriverManager.getConnection("Jdbc:odbc:dsn1","scott","tiger");
        Statement st=con.createStatement();
st.executeQuery("select r.workid,w.workname,r.work_done_by from w1_report
r,w1_details w where r.workstatus='ongoing' and r.month="+month+" and
r.year="+year+" and r.workid=w.workid");
        ResultSet rs=st.getResultSet();
        while(rs.next())
        {
%>
        <tr><td> <%= rs.getString(1) %></td>
        <td> <%= rs.getString(2) %></td>
        <td> <%= rs.getString(3) %></td>
        <%
        }
        %>
        </tr>
        </table>
        </td>

```

```

</tr>
<tr>
<td>
<h3><b>Pending Works</b></h3>
<table bgcolor="lightblue" border=1 align="left">
<tr><td>Work ID</td>
<td> Work Name</td>
<td>Work Done By</td>
</tr>
<%
st.executeQuery("select r.workid,w.workname,r.work_done_by from w1_report
r,w1_details w where r.workstatus='pending' and r.month="+month+" and
r.year="+year+" and r.workid=w.workid");
rs=st.getResultSet();
while(rs.next())
{
%>
<tr><td> <%= rs.getString(1) %></td>
<td> <%= rs.getString(2) %></td>
<td> <%= rs.getString(3) %></td>
<%
}
%>
</tr>
</table>
</td>
</tr>
<tr>
<td>
<h3><b>Completed Works</b></h3>
<table bgcolor="lightblue" border=1 align="left">
<tr><td>Work ID</td>
<td> Work Name</td>
<td>Work Done By</td>
</tr>
<%
st.executeQuery("select r.workid,w.workname,r.work_done_by from w1_report
r,w1_details w where r.workstatus='completed' and r.month="+month+" and
r.year="+year+" and r.workid=w.workid");
rs=st.getResultSet();
while(rs.next())
{
%>
<tr><td> <%= rs.getString(1) %></td>
<td> <%= rs.getString(2) %></td>
<td> <%= rs.getString(3) %></td>
<%
}
%>

```

```

        </tr>
    </table>
    <%
    }
    catch(Exception e)
    {
        out.println(e);
    }
    %>
</td></tr>
</table>
</body>
<html>

```

### **agmframe.html:**

```

<frameset cols=25%,* border=0 scrolling=no>
<frame name=f1 src="agm_menu.html" scrolling=no>
<frame name=f2>
</frameset>
</frameset>

```

### **agm\_menu.html:**

```

<html>
<BODY>
<TABLE align=left border=0 cols=1 rows="8" cellspacing=5>
<TBODY>
<TR>
<TH><B><FONT color="maroon" face="bookantiqua">
<H4>AGM MENU</H4></FONT></B></TH>
<TR>
<TD><A href="addwork.html" target="f2">Add Work</A>
</TD></TR>
<TR>
<TD><A href="Assign.jsp" target="f2">Assign Work</A>
</TD></TR>
<TR>
<TD><A href="delwork.html" target="f2">Delete Work</A>
</TD></TR>
<TR>
<TD><A href="view_work_status.jsp" target="f2">Work Status
</A> </TD></TR>
<TR>
<TD><A href="search.html" target="_top">Search</A> </TD></TR>
<TR>
<TD><A href="framepage.html" target="_top">Log Out</A> </TD></TR>

```

```
</TBODY></TABLE></BODY></HTML>
```

### **addwork.html:**

```
<html>
<body >
<font face="bookantique" color="maroon">
<head><h2><center>Add Work</center> </h2></head></font>
<script language="javascript">
function foc()
{
  add.workid.focus();
}
function check()
{
  if(add.workid.value=="")
  {
    alert("Please enter Work Id")
    add.workid.focus();
  }
  else if(add.workname.value=="")
  {
    alert("Please enter Work Name")
    add.workname.focus();
  }
  else if(add.duration.value=="")
  {
    alert("Please enter Duration")
    add.duration.focus();
  }
  else
  {
    add.action="add_work.jsp"
  }
}
</script>
<body bgcolor="white" onload="foc()">
<font face="bookantiqua" color="maroon">
<h4>
<form name="add" onsubmit="check()">
<pre>
<center>
Work Id    : <input type="text" name="workid">
Work Name  : <input type="text" name="workname">
Duration(Days): <input type="text" name="duration">
  <input type="submit" name="submit" value="Add">  <input
type="reset" name="reset" value="reset">
```

```
</center></pre></form></h4></font></body></body></html>
```

### **add\_work.jsp:**

```
<html>
<body>
<%@ page import="java.sql.*" %>
<%
int workid=Integer.parseInt(request.getParameter("workid"));
String workname=request.getParameter("workname");
int duration=Integer.parseInt(request.getParameter("duration"));
    try{
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        Connection
con=DriverManager.getConnection("Jdbc:odbc:dsn1","scott","tiger");
        Statement st=con.createStatement();
st.executeUpdate("insert into w1_details(workid,workname,duration)
values('"+workid+"','"+workname+"','"+duration +"'");
con.commit();
out.println("work is added to the database");
    }
    catch(Exception e)
    {
        out.println(e);
    }
%>
</body>
</html>
```

### **assign.jsp:**

```
<html>
<head><font color="maroon"><b><h4><center> ASSIGN
WORK</center></h4></b></font>
<script language="javascript">
function check()
{
    assign.submit()
}
</script>
</head>
<body>
<form name="assign" action="work_status.jsp">
<%@ page import="java.sql.*" %>
<%
    String user=(String)session.getAttribute("userid");
    String userid=user;
```

```

try{
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
    Connection
con=DriverManager.getConnection("Jdbc:odbc:dsn1","scott","tiger");
    Statement st=con.createStatement();
    ResultSet rs=st.executeQuery("Select * from e1_details where
w_status='w'and worksunder='"+userid+"'");
%>
    <TABLE align="left" border=0 cellpadding=10 cellspacing=25
bgcolor="lightblue">
    <TBODY>
    <TR>
    <Th><B><FONT color="maroon" face="bookantiqua">
    <H5>Employees working under you and works that are to be
assigned</H5></b></FONT></Th>
    </TR>
    <tr><td>Employees working under you</td>
    <td><select name="ename">
        <%
        while(rs.next())
        {
        %>
        <option ><%= rs.getString(1) %></option>
        <%
        }
        %>
        </select></td>
    </tr>
    <tr>
    <td>Work ID
    </td>
    <td>
    <select name="wname">
    <%
    rs=st.executeQuery("select * from w1_details where s='notassigned'");
    while(rs.next())
    {out.println("hello");
    %>
        <option><%= rs.getString(1) %></option>
    <%
    }
    }
    catch(Exception e)
    {
        e.printStackTrace();
    }
    %>
    </td>
    <tr>

```

```

        <tr>
        <td><input type="button" value="Assign" onclick="check()">
        </td>
        </tr>
    </tbody>
</table>
</form>
</body>
</html>

```

### **work\_status.jsp:**

```

<html>
<body>
<form name=work_status>
<%@ page import="java.sql.*,java.util.*" %>
<%@ page import="java.text.SimpleDateFormat" %>
<%
    Object user=session.getAttribute("userid");
    String work_by=(String)user;
    String work_to=request.getParameter("ename");
    String workid=request.getParameter("wname");
    SimpleDateFormat dateFormat=new SimpleDateFormat("dd-MM-yyyy");
    String date=(String)dateFormat.format(new java.util.Date());
    int day=Integer.parseInt(date.substring(0,2));
    int month=Integer.parseInt(date.substring(3,5));
    int year=Integer.parseInt(date.substring(6,10));
    try{
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        Connection
con=DriverManager.getConnection("Jdbc:odbc:dsn1","scott","tiger");
        Statement st=con.createStatement();
        ResultSet rs=st.executeQuery("Select designation from e1_details
where empid='"+work_by+"'");
        rs.next();
        String designation=rs.getString(1);
        if(designation.equals("agm"))
        {
            st.executeUpdate("insert into
work1_status(workid,work_done_by,work_assigned_by,work_status)
values('"+workid+"','"+work_to+"','"+work_by+"',startstage)");
        }
        else
        {
            String sql="insert into
work1_status(workid,work_done_by,work_assigned_by,work_status)
values('"+workid+"','"+work_to+"','"+work_by+"',startstage)";
            st.executeUpdate(sql);

```

```

        st.executeUpdate("update work1_status set
work_status='assigned' where work_done_by='"+work_by+"' and
workid='"+workid+"'");
    }
    %>
    <br><br><br><br><b>Work is assigned sucessfully to <%=
work_to %></b>
    <%
        String sq="update w1_details set
startdate=sysdate,day='"+day+"',month='"+month+"', year='"+year+"',s='assigned'
where workid='"+workid+"'";
        st.executeUpdate(sq);
        con.commit();
    }
    catch(Exception e)
    {
        out.println(e);
    }
    %>
</form>
</body>
</html>

```

### view\_work\_status.jsp:

```

<html>
<head><h4><center>OVERALL WORK STATUS</center></h4></head>
<body>
<table border=1 bgcolor="lightblue">
    <tr>
        <td><h5><b>WORKID</b></h6></td>
        <td><h5><b>WORK NAME</b></h6></td>
        <td><h5><b>WORK ASSIGNED TO</b></h6></td>
        <td><h5><b>WORK ASSIGNED BY</b></h6></td>
        <td><h5><b>START DATE</b></h6></td>
        <td><h5><b>WORK STATUS</b></h6></td>
    </tr>
    <%@ page import="java.sql.*" %>
    <%
        Object o1=session.getAttribute("userid");
        String userid=(String)o1;
        try{
            Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            Connection
con=DriverManager.getConnection("Jdbc:odbc:dsn1","scott","tiger");
            Statement st=con.createStatement();
            System.out.println(userid);

```

```

        ResultSet rs=st.executeQuery("select
w.workid,w.workname,s.work_done_by,s.work_assigned_by,w.startdate,s.work_sta
tus from w1_details w,work1_status s where s.workid=w.workid and
s.work_assigned_by='"+userid+"'");
        //ResultSet rs=st.executeQuery("select * from work1_status where
work_assigned_by='"+userid+"'");
        rs.next();
        while(rs.next())
        {
        %>
        <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.getString(6) %></td>
        </tr>
        <%
        }
        }
        catch(Exception e)
        {
        out.println(e);
        }
        %>
</table>
<body>
<html>

```

### search.html:

```

<frameset cols=25%,* border=0 scrolling=no noresize>
<frame name="menu" src="search_menu.jsp">
<frame name="targetpage">
</frameset>

```

### search\_menu.jsp:

```

<BODY>
<TABLE align=left border=0 cols=1 rows="8" cellspacing=5>
<TBODY>
<TR>
<TH><B><FONT color="maroon" face="bookantiqua">
<H4>SEARCH MENU</H4></FONT></B></TH>
<TR>
<TD><font face="bookantiqua" color="maroon" size="2">STATUS</font>

```

```

</TD></TR>
<TR>
<TD><A href="completed.jsp" target="targetpage">Completed works</A>
</TD></TR>
<TR>
<TD><A href="pending.jsp" target="targetpage">Pending Works
</A> </TD></TR>
<TR>
<TD><A href="ongoing.jsp" target="targetpage">Ongoing Work
</A> </TD></TR>
<TR>
<TD><A href="search_emp_status.jsp" target="targetpage">Employee Status
</A> </TD></TR>
<TR>
<TD><font face="bookantiqua" color="maroon" size="2">STATUS THROUGH
DATE</font>
</TD></TR>
<TR>
<TD><A href="daily.jsp" target="targetpage">Enter Date
</A> </TD></TR>
<TR>
<TD><A href="agmframe.html" target="_top">AGM Menu
</A> </TD></TR>
<TR>
<TD><A href="framepage.html" target="_top">Log Out
</A> </TD></TR>
</TBODY></TABLE></BODY></HTML>

```

### completed.jsp:

```

<html>
<head><center><h3><b> COMPLETED WORKS</b></center></h3>
<body>
<table>
<tr>
<td>
<h3><b>Completed Work Details</b></h3>
<table bgcolor="lightblue" border=1 align="left">
<tr><td>Work ID</td>
<td> Work Name</td>
<td>Work Done By</td>
</tr>
<%@ page import="java.sql.*" %>
<%
try{
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");

```

```

        Connection
con=DriverManager.getConnection("Jdbc:odbc:dsn1","scott","tiger");
        Statement st=con.createStatement();
st.executeQuery("select r.workid,w.workname,r.work_done_by from w1_report
r,w1_details w where r.workstatus='completed' and r.workid=w.workid");
        ResultSet rs=st.getResultSet();
        if(rs!=null)
        {
            while(rs.next())
            {
%>
                <tr><td> <%= rs.getString(1) %></td>
                <td> <%= rs.getString(2) %></td>
                <td> <%= rs.getString(3) %></td>
                <%
                }
                }
            else
            {
                out.println("NO RECORDS FOUND");
            }
        }
        catch(Exception e)
        {
            out.println(e);
        }
        %>
        </tr>
        </table>
    </body>
</html>

```

### pending.jsp:

```

<html>
<head><center><h3><b> PENDING WORKS</b></center></h3>
<body>
<table>
<tr>
<td>
<h3><b>Pending Work Details</b></h3>
        <table bgcolor="lightblue" border=1 align="left">
            <tr><td>Work ID</td>
            <td> Work Name</td>
            <td>Work Done By</td>
            </tr>
        <%@ page import="java.sql.*" %>

```

```

<%
    try{
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        Connection
con=DriverManager.getConnection("Jdbc:odbc:dsn1","scott","tiger");
        Statement st=con.createStatement();
st.executeQuery("select r.workid,w.workname,r.work_done_by from w1_report
r,w1_details w where r.workstatus='pending' and r.workid=w.workid");
        ResultSet rs=st.getResultSet();
        if(rs!=null)
        {
            while(rs.next())
            {
%>
                <tr><td> <%= rs.getString(1) %></td>
                <td> <%= rs.getString(2) %></td>
                <td> <%= rs.getString(3) %></td>
                <%
                }
                }
            else
            {
%>
                NO RECORDS FOUND
                <%
                }
                }
            catch(Exception e)
            {
                out.println(e);
            }
%>
        </tr>
        </table>
</body>
</html>

```

### ongoing.jsp:

```

<html>
<head><center><h3><b> ONGOING WORKS</b></center></h3>
<body>
<table>
<tr>
<td>
<h3><b>OnGoing Work Details</b></h3>
        <table bgcolor="lightblue" border=1 align="left">

```

```

        <tr><td>Work ID</td>
        <td> Work Name</td>
        <td>Work Done By</td>
        </tr>
<%@ page import="java.sql.*" %>
<%
    try{
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        Connection
con=DriverManager.getConnection("Jdbc:odbc:dsn1","scott","tiger");
        Statement st=con.createStatement();
st.executeQuery("select r.workid,w.workname,r.work_done_by from w1_report
r,w1_details w where r.workstatus='ongoing' and r.workid=w.workid");
        ResultSet rs=st.getResultSet();
        if(rs!=null)
        {
            while(rs.next())
            {
%>
                <tr><td> <%= rs.getString(1) %></td>
                <td> <%= rs.getString(2) %></td>
                <td> <%= rs.getString(3) %></td>
                <%
                }
                }
            else
            {
                out.println("NO RECORDS FOUND");
            }
        }
        catch(Exception e)
        {
            out.println(e);
        }
    }
%>
    </tr>
    </table>
</body>
</html>

```

### search\_emp\_status.jsp:

```

<html>
<body bgcolor="white">
<form action="emp_status_diss.jsp">
<table bgcolor="lightblue" cellpadding=5 cellspacing=10 align="left">
<tr>
<td> Select Employee Name

```

```

<td>
<select name="ename">
<%@ page import="java.sql.*" %>
<%
    String user=(String)session.getAttribute("userid");
    int userid=Integer.parseInt(user);
    try{
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        Connection
con=DriverManager.getConnection("Jdbc:odbc:dsn1","scott","tiger");
        Statement st=con.createStatement();
        st.executeQuery("select empid,ename from e1_details where empid not
in("+userid+""));
        ResultSet rs=st.getResultSet();
        while(rs.next())
        {
            %>
            <option><%= rs.getString(1) %></option>
            <%
            }
            }
        catch(Exception e)
        {
            out.println(e);
        }
        %>
    }
</select>
</td>
</tr>
<tr>
<td>
<input type="submit" value="Status">
</td>
</tr>
</table>
</form>
</body>
</html>

```

### **emp\_status\_diss.jsp:**

```

<html>
<body bgcolor="white">
<table>
<tr>
<td>
<h3><b><center>WORK STATUS</center></b></h3>
        <table bgcolor="lightblue" border=1 align="left">

```

```

        <tr><td>Work ID</td>
        <td> Work Name</td>
        <td>Work status</td>
        </tr>
<%@ page import="java.sql.*" %>
<%
        String user=request.getParameter("ename");
        int ename=Integer.parseInt(user);
        try{
            Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            Connection
con=DriverManager.getConnection("Jdbc:odbc:dsn1","scott","tiger");
            Statement st=con.createStatement();
            ResultSet rs=st.executeQuery("select
r.workid,w.workname,r.workstatus from w1_details w,w1_report r where
r.workid=w.workid and r.work_done_by="+ename);
            rs.next();
            while(rs.next())
            {
%>
                <tr><td> <%= rs.getString(1) %></td>
                <td> <%= rs.getString(2) %></td>
                <td> <%= rs.getString(3) %></td>
                </tr>
                <%
                }
                }
                catch(Exception e)
                {
                    out.println(e);
                }
                %>
                </tr>
                </table>
                </td>
                </tr>
            </table>
        </body>
</html>

```

## Manager.jsp:

```
<frameset cols=25%,* border=0 scrolling=no noresize>
<frame name="menu" src="manager_menu.jsp">
<frame name="targetpage">
</frameset>
```

## manager\_menu.jsp:

```
<html>
<BODY>
<TABLE align=left border=0 cols=1 rows="8" cellspacing=5>
  <TBODY>
    <TR>
      <TH><B><FONT color="maroon" face="bookantiqua">
        <H4>MANAGER\ COORDINATOR MENU</H4></FONT></B></TH>
    <TR>
      <TD><A href="assign_mgr.jsp" target="targetpage">Assign Work</A>
    </TD></TR>
    <TR>
      <TD><A href="view_work.jsp" target="targetpage">View Work</A>
    </TD></TR>
    <TR>
      <TD><A href="status_update.jsp" target="targetpage">Status Update</A>
    </TD></TR>
    <TR>
      <TD><A href="view_work_status_mgr.jsp" target="targetpage">Work Status
    </A> </TD></TR>
    <TR>
      <TD><A href="search.html" target="_top">Search</A> </TD></TR>
      <TR>
      <TD><A href="framepage.html" target="_top">Log Out</A> </TD></TR>
  </TBODY></TABLE></BODY></HTML>
```

## assign\_mgr.jsp:

```
<html>
<body>
<head>
<font color="maroon"><b><h4><center> ASSIGN
WORK</center></h4></b></font>
<script language="javascript">
function check()
{
    assign.submit()
}
</script>
```

```

</head>
<body>
<form name="assign" action="work_status.jsp">
<%@ page import="java.sql.*" %>
<%@ import="java.util.*" %>
<%
    Object user=session.getAttribute("userid");
    String userid=(String)user;
    try{
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        Connection
con=DriverManager.getConnection("Jdbc:odbc:dsn1","scott","tiger");
        Statement st=con.createStatement();
        ResultSet rs=st.executeQuery("Select * from e1_details where
worksunder="+userid);
%>
        <TABLE align="left" border=0 cols=2 rows=1 cellpadding=10
cellspacing=25>
        <TBODY>
        <TR>
        <Th><B><FONT color="black" face="bookantiqua">
        <H5>Employees working under you and works that r to be
assigned</H5></b></FONT></Th>
        </TR>
        <tr><td>Employees working under you</td>
        <td><select name="ename">
            <%
                while(rs.next())
                {
            %>
                <option><%= rs.getString(1) %></option>
            <%
                }
            %>
        </select></td>
        </tr>
        <tr>
        <td>Work ID
        </td>
        <td>
        <select name="wname">
            <%
                rs=st.executeQuery("select * from w1_details where workid in( select
workid from work1_status where work_done_by="+userid+" and
work_status='startstage')");
                while(rs.next())
                {
            %>
                <option><%= rs.getString(1) %></option>

```

```

        <%
        }
        }
        catch(Exception e)
        {
                e.printStackTrace();
        }
        %>
    </td>
    <tr>
    <tr>
    <td><input type="button" value="Assign" onclick="check()">
    </td>
    </tr>
</tbody>
</table>
</form>
</body>
</html>

```

### view\_work.jsp:

```

<html>
<head><h4><center>WORKS ASSIGNED TO YOU</center></h4></head>
<body>
<table border=1 bgcolor="lightblue">
    <tr>
    <td><h5><b>WORKID</b></h6></td>
    <td><h5><b>WORK NAME</b></h6></td>
    <td><h5><b>WORK ASSIGN BY</b></h6></td>
    <td><h5><b>START DATE</b></h6></td>
    <td><h5><b>DURATION</b></h6></td>
    </tr>
<%@ page import="java.sql.*" %>
<%
    Object o=session.getAttribute("userid");
    String user=(String)o;
    int userid=Integer.parseInt(user);
    try{
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        Connection
con=DriverManager.getConnection("Jdbc:odbc:dsn1","scott","tiger");
        Statement st=con.createStatement();
        String sq="select
s.workid,w.workname,s.work_assigned_by,w.startdate,w.duration from w1_details
w,work1_status s where s.workid=w.workid and s.work_done_by='"+userid+"'";
        st.executeQuery(sq);

```

```

        ResultSet rs=st.getResultSet();
        int count=0;
        while(rs.next())
        {
++count;
        }
        int workid[]=new int[count];
        int work_assign_by[]=new int[count];
        String startdt[]=new String[count];
        String workname[]=new String[count];
        String dur[]=new String[count];
        st.executeQuery("select
s.workid,w.workname,s.work_assigned_by,w.startdate,w.duration from w1_details
w,work1_status s where s.workid=w.workid and s.work_done_by='"+userid+"'");
        rs=st.getResultSet();
        int i=0;
        while(rs.next())
        {
            workid[i]=Integer.parseInt(rs.getString(1));
            work_assign_by[i]=Integer.parseInt(rs.getString(3));
            startdt[i]=rs.getString(4);
            workname[i]=rs.getString(2);
            dur[i]=rs.getString(5);
%>
            <tr><td><%= workid[i]%></td>
            <td><%=workname[i]%></td>
            <td><%=work_assign_by[i]%></td>
            <td><%=startdt[i]%></td>
            <td><%=dur[i]%></td>
%>
            <%=
                i++;
            }
        }
        catch(Exception e)
        {
            out.println(e);
        }
%>
</table>
<body>
<html>

```

**status\_update.jsp:**

```

<html>
<head><font color="maroon"><b><h4><center> STATUS
UPDATE</center></h4></b></font>
</head>
<body>
<form name="status" action="update_work_status.jsp";>
<%@ page import="java.sql.*" %>
<%
    String user=(String)session.getAttribute("userid");
    int userid=Integer.parseInt(user);
    try{
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        Connection
con=DriverManager.getConnection("Jdbc:odbc:dsn1","scott","tiger");
        Statement st=con.createStatement();
        ResultSet rs=st.executeQuery("select * from w1_details where workid
in (select workid from work1_status where work_status='ongoing' or
work_status='pending'and work_done_by="+userid+"));
    %>
    <TABLE align="left" border=0 cols=2 rows=1 cellpadding=10
cellspacing=30>
    <tr><td>Works assigned to you</td>
    <td><select name="wid">
        <%
        while(rs.next())
        {
        %>
            <option><%= rs.getString(1) %></option>
        <%
        }
        }
    catch(Exception e)
    {
        e.printStackTrace();
    }
    %>
</select>
</td></tr>
<tr>
    <td>Select ur status</td>
    <td>
        <input type="radio" name="sts" value="c">Completed
        <input type="radio" name="sts" value="p">Pending
        <input type="radio" name="sts" value="o">OnGoing
    </td>
</tr>
<tr>
    <td><input type="submit" value="Update Status">

```

```

        </td>
    </tr>
</TABLE>
</form>
</body>
<html>

```

### update\_work\_status.jsp:

```

<html>
<head><h4><center> Status Update </center></h4></head>
<body>
<%@ page import="java.sql.*" %>
<%@ page import="java.text.SimpleDateFormat" %>
<%
    String user=(String)session.getAttribute("userid");
    int userid=Integer.parseInt(user);
    String status=request.getParameter("sts");
    int workid=Integer.parseInt(request.getParameter("wid"));
    SimpleDateFormat dateFormat=new SimpleDateFormat("dd-MM-yyyy");
    String date=(String)dateFormat.format(new java.util.Date());
    int day=Integer.parseInt(date.substring(0,2));
    int month=Integer.parseInt(date.substring(3,5));
    int year=Integer.parseInt(date.substring(6,10));
    try
    {
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        Connection
con=DriverManager.getConnection("Jdbc:odbc:dsn1","scott","tiger");
        Statement st=con.createStatement();
        st.executeUpdate("update work1_status set work_status='"+status+"'
where workid='"+workid+"' and work_done_by='"+userid);
        st.executeUpdate("insert into
w1_report(workid,work_done_by,workstatus,day,month,year)
values '"+workid+"','"+userid+"','"+status+"','"+day+"','"+month+"','"+year+"'");
        con.commit();
    %>
    <br>
    <br>
    <br>
    <center>
        <b> Your status is updated </b></center>
    <%
    }
    catch(Exception e)
    {
        out.println(e);

```

```

    }
    %>
</body>
</html>

```

### view\_work\_status\_mgr.jsp:

```

<html>
<head><h4><center>WORK STATUS</center></h4></head>
<body>
<table border=1 bgcolor="lightblue">
    <tr>
        <td><h5><b>WORKID</b></h6></td>
        <td><h5><b>WORK NAME</b></h6></td>
        <td><h5><b>WORK ASSIGNED TO</b></h6></td>
        <td><h5><b>WORK ASSIGNED BY</b></h6></td>
        <td><h5><b>START DATE</b></h6></td>
        <td><h5><b>WORK STATUS</b></h6></td>
    </tr>
<%@ page import="java.sql.*" %>
<%
    String userid=(String)session.getAttribute("userid");
    try{
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        Connection
con=DriverManager.getConnection("Jdbc:odbc:dsn1","scott","tiger");
        Statement st=con.createStatement();
        ResultSet rs=st.executeQuery("select
s.workid,w.workname,s.work_done_by,s.work_assigned_by,w.startdate,s.work_stat
us from w1_details w,work1_status s where s.workid=w.workid and
s.work_done_by='"+userid+"'");
        while(rs.next())
        {
            %>
            <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.getString(6) %></td>
            <%
                }
            }
        catch(Exception e)
        {
            out.println(e);
        }
    }

```

```
%>  
</table>  
<body>  
<html>
```

# TESTING

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding. The increasing visibility of software as a system element and the attendant costs associated with a software failure are motivating forces for we planned through testing.

There are basically two types of test approaches White box and Black box-testing methods. In this project all the loop constructs have been tested for their boundary and intermediate conditions .The test data was designed with a view to check for all conditions and logical decisions. Error handling has been taken care of by use of exception handlers.

A strategy for software testing integrates software test case design techniques into a well planned series of steps that result in the successful construction of software.

## **VARIOUS TESTINGS:**

### **UNIT TESTING:**

Unit testing focuses verification efforts on the smallest unit of software design, in the module. Using procedural design description as a guide, important control paths are tested to uncover errors within the boundaries of the module. The unit testing is normally White box testing oriented and the step can be conducted in parallel for multiple modules.

### **INTEGRATION TESTING:**

Integration testing is a systematic technique for constructing the program structure while conducting tests to uncover errors associated with interfacing. The objective is to take unit tested modules and build a program structure that has been dictated by design.

## UNIT TESTING

## TEST CASES:

TEST NAME	INPUT	EXPECTED OUTPUT	ACTUAL OUTPUT	TEST STATUS
Login Form	Valid User id, Invalid password	Password is invalid	Password is invalid	Pass
Login Form	Invalid User id, Valid password	User id is invalid	User id is invalid	Pass
Login Form	Invalid User id, Invalid password	User id, password are invalid	User id, password are invalid	Pass
Login Form	Valid User id, valid password	User id, password are correct	User id, password are correct	Pass

<b>Administrator Form:</b> Add employee Form	If employee id is given in characters	Invalid employee id	Invalid employee id	Pass
Add employee Form	If phone no exceeds 10 digits	Inserted value is too large	Inserted value is too large	Pass
Add employee Form	If address field exceeds 25 characters	Inserted value is too large	Inserted value is too large	Pass
Add Employee Form	If works under field is given in characters	Invalid works under field	Invalid Works under field	Pass
Delete Form	Select an employee id from the list	Employee is deleted from database	Employee is deleted from database	Pass
Modify Form	Select an employee id from the list	Employee details are displayed	Employee details are displayed	Pass
Report Generation Form	Select type of report to be generated	Type of report is displayed	Type of report is displayed	Pass
Daily Report Form	Select a day from the list	Report of that day is generated	Report of that day is generated	Pass
Daily Report Form	If the selected date is 31 <sup>st</sup> ,feb	There is no 31 <sup>st</sup> ,feb	Report is generated with no records in it	Fail
Weekly Report Form	Select a week from the list	Report of that week is generated	Report of that week is generated	Pass

Monthly Report Form	Select a month from the list	Report of that month is generated	Report of that month is generated	Pass
<b>AGM Form:</b> Add Work Form	If work id is given in characters	Invalid work id	Invalid work id	Pass
Add Work Form	If the duration no. exceeds 3 digits	Inserted value is too large	Inserted value is too large	Pass
Add Work Form	If the work name exceeds 15 characters	Inserted value is too large	Inserted value is too large	Pass
Assign Work Form	Employee id, work id are to be selected from the lists	Successful assignment	Successful assignment	Pass
Work Status Form	Work details comes from database table	Overall work status is displayed	Overall work status is displayed	Pass
Search Form: Completed Works form	Work details comes from database table	Displays completed works table	Displays completed works table	Pass
Pending Works Form	Work details comes from database table	Displays pending works table	Displays pending works table	Pass
Ongoing Works table	Work details comes from database table	Displays ongoing works table	Displays ongoing works table	Pass
Employee status Form	Select an employee id from the list	Status of the employee is to be displayed	Status of the employee is to be displayed	Pass
Date Form	Select a day from the list	Report of that day is generated	Report of that day is generated	Pass

<b>Manager/coordinator Form:</b> Assign Work Form	Employee id, work id are to be selected from the lists	Successful assignment	Successful assignment	Pass
View Work form	Work details comes from database table	Table containing work details	Table containing work details	Pass

Status Update form	Select a work id from list to update status	Updated status message	Updated status message	Pass
Work Status Form	Work details comes from database table	Work Status is displayed	Work Status is displayed	Pass
Search Form: Status: Completed Works	Work details comes from database table	Displays completed works table	Displays completed works table	Pass
Pending Works Form	Work details comes from database table	Displays pending works table	Displays pending works table	Pass
Ongoing Works table	Work details comes from database table	Displays ongoing works table	Displays ongoing works table	Pass
Employee status Form	Select an employee id from the list	Status of the employee is to be displayed	Status of the employee is to be displayed	Pass
Date Form	Select a day from the list	Report of that day is generated	Report of that day is generated	Pass

<b>Employee Form:</b> View Work	Work details comes from database table	Table containing work details	Table containing work details	Pass
Status Update form	Select a work id from list to update status	Updated status message is generated	Updated status message is generated	Pass

# INTEGRATION TESTING

## TEST CASES:

TEST NAME	INPUT	EXPECTED OUTPUT	ACTUAL OUTPUT	TEST STATUS
Login Program	Valid User id, Invalid password	Password is invalid	Password is invalid	Pass
Login Program	Invalid User id, Valid password	User id is invalid	User id is invalid	Pass
Login Program	Invalid User id, Invalid password	User id, password are invalid	User id, password are invalid	Pass
Login Program	Valid User id, valid password	Goes to AGM/ Administrator/ Manager/Employee Program	Goes to AGM/ Administrator/ Manager/Employee Program	Pass

<b>Administrator Program:</b> Add Employee Program	Employee Details are entered	Employee details should be added to the database	Employee details should be added to the database	Pass
Delete Employee Program	Select employee id from the list	Employee details should be deleted from database	Employee details should be deleted from database	Pass
Modify Employee details program	Select employee id from the list	Employee details should be modified in the database	Employee details should be modified in the database	Pass
Report Generation Program	Select type of report to be generated	Type of report is displayed	Type of report is displayed	Pass
Daily Report Program	Select a day from the list	Report of that day is generated	Report of that day is generated	Pass
Weekly Report Program	Select a week from the	Report of that week is generated	Report of that week is generated	Pass

	list			
Monthly Report Program	Select a month from the list	Report of that month is generated	Report of that month is generated	Pass

<b>AGM Program:</b> Add work program	Work details are entered	Work details are added to the Database	Work details are added to the database	Pass
Assign work Program	Select a work id and assign it to an empid	Work is assigned to particular employee	Work is assigned to particular employee	Pass
Delete work program	Select a work id from the list	Work is deleted from the database	Work is deleted from the database	Pass
Work Status Program	Workdetails comes from database table	Overall work status is displayed	Overall work status is displayed	Pass
<b>Search Program:</b> Completed Works Program	Work details comes from database table	Displays completed works table	Displays completed works table	Pass
Pending Works Program	Work details comes from database table	Displays pending works table	Displays pending works table	Pass
Ongoing Works Program	Work details comes from database	Displays ongoing works table	Displays ongoing works table	Pass
Employee status Program	Select an employee id from the list	Status of the employee is to be displayed	Status of the employee is to be displayed	Pass
Date Program	Select a day from the list	Report of that day is generated	Report of that day is generated	Pass
<b>Manager/coordinator Program:</b> Assign Work Program	Select a work id and assign it to an empid	Work is assigned to particular employee	Work is assigned to particular employee	Pass

View Work Program	Work details comes from database table	Table containing work details	Table containing work details	Pass
Status Update Program	Select a work id from list to update status	Updated status message	Updated status message	Pass
Work Status Program	Work details comes from database table	Work Status is displayed	Work Status is displayed	Pass
<b>Search Program:</b> <b>Status:</b> Completed Works Program	Work details comes from database table	Displays completed works table	Displays completed works table	Pass
Pending Works Program	Work details comes from database table	Displays pending works table	Displays pending works table	Pass
Ongoing Works Program	Work details comes from database table	Displays ongoing works table	Displays ongoing works table	Pass

Employee status Program	Select an employee id from the list	Status of the employee is to be displayed	Status of the employee is to be displayed	Pass
Status through Date Program	Select a day from the list	Report of that day is generated	Report of that day is generated	Pass

<b>Employee Program:</b> View Work Program	Work details comes from database table	Table containing work details is displayed	Table containing work details	Pass
Status Update Program	Select a work id from list to update	Updated status message is generated	Updated status message is generated	Pass

	status			
--	--------	--	--	--

# SAMPLE SCREENS

## Login Menu:



### **Definition:**

*Work Flow Management systems allow users to define and control various activities associated with the business process.*

A screenshot of a login form titled "LoginHere" on a light blue background. The form contains two input fields: "User id" with the value "551" and "Password" with the value "jolak". Below the fields are two buttons: "submit" and "reset".

**LoginHere**

**User id**

**Password**

## Administrator Menu:



### **Administrator Menu**

- [Add Employee](#)
- [Delete Employee](#)
- [Modify Employee](#)
- [View Employee](#)
- [Report Generation](#)
- [Log Out](#)

## Add employee:

**Administrator Menu**

- [Add Employee](#)
- [Delete Employee](#)
- [Modify Employee](#)
- [Report Generation](#)
- [Log Out](#)

**Employee Details to store in DataBase**

Empid : 99

Employee Name : example

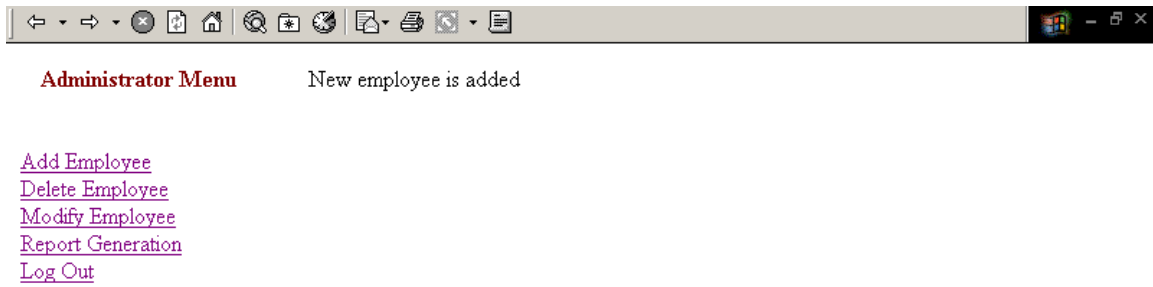
Designation : MANAGER

Works Under : 523

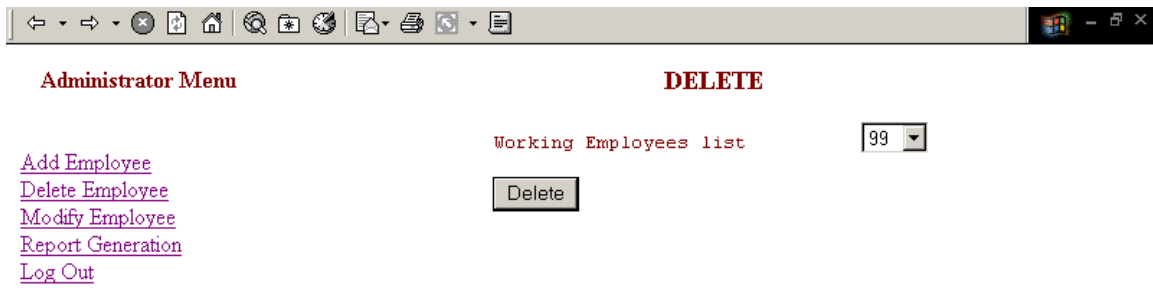
Address : maggi colony

Phone No : 143141153

Output screen:

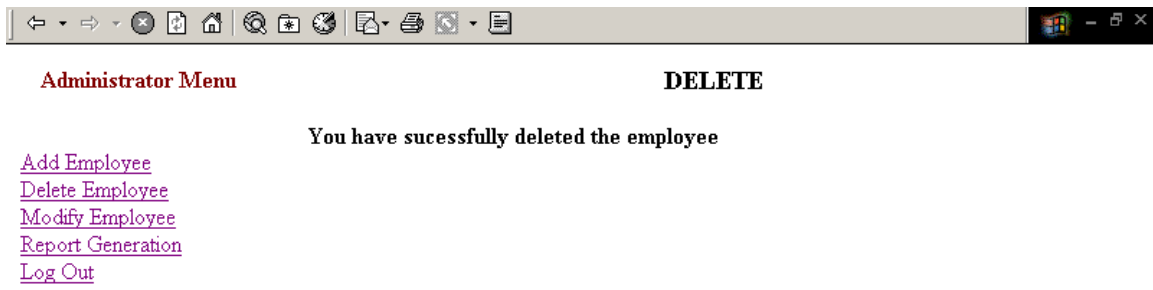


Delete employee:

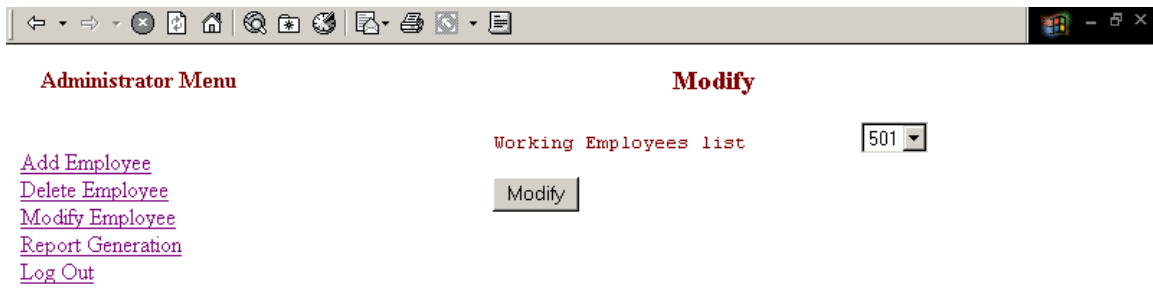


The screenshot shows a web browser window with a standard toolbar at the top. Below the toolbar, the page content is organized into two main sections. On the left, under the heading "Administrator Menu", there is a vertical list of links: "Add Employee", "Delete Employee", "Modify Employee", "Report Generation", and "Log Out". On the right, under the heading "DELETE", there is a label "Working Employees list" followed by a dropdown menu currently displaying "99". Below this dropdown is a button labeled "Delete".

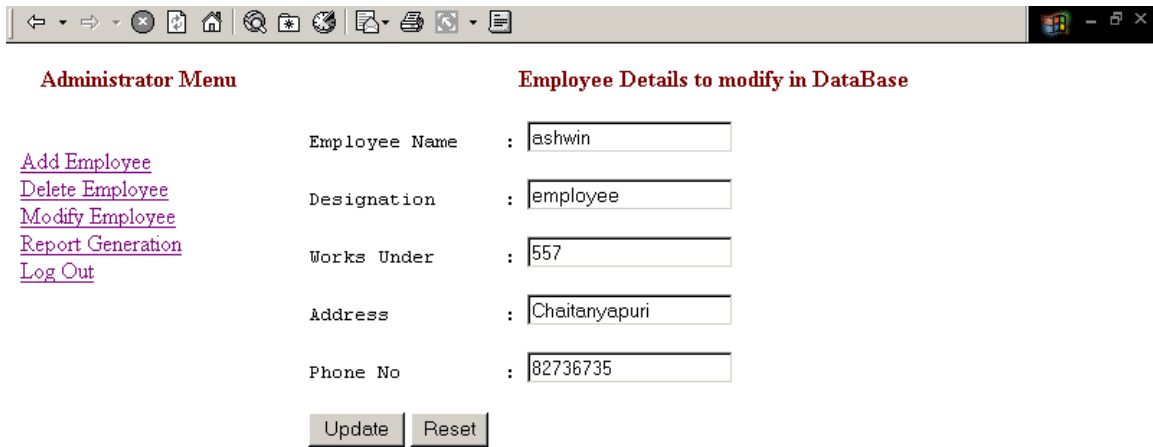
Output Screen:



Modify employee:



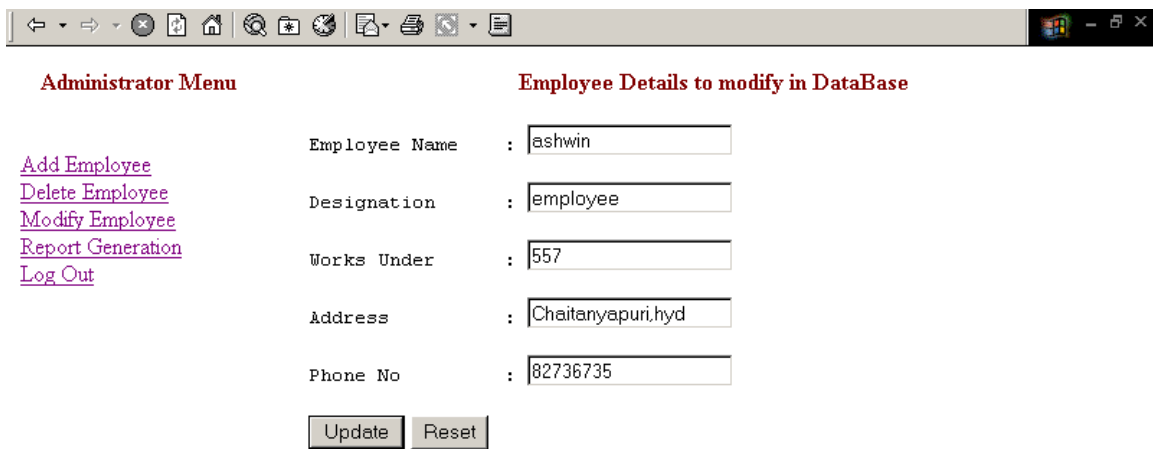
Output screen1:



The screenshot shows a web browser window with a toolbar at the top. The main content is divided into two sections. On the left, under the heading "Administrator Menu", there are five links: "Add Employee", "Delete Employee", "Modify Employee", "Report Generation", and "Log Out". On the right, under the heading "Employee Details to modify in DataBase", there is a form with five input fields: "Employee Name" (containing "ashwin"), "Designation" (containing "employee"), "Works Under" (containing "557"), "Address" (containing "Chaitanyapuri"), and "Phone No" (containing "82736735"). Below the form are two buttons: "Update" and "Reset".

Administrator Menu		Employee Details to modify in DataBase	
<a href="#">Add Employee</a>	Employee Name	:	<input type="text" value="ashwin"/>
<a href="#">Delete Employee</a>	Designation	:	<input type="text" value="employee"/>
<a href="#">Modify Employee</a>	Works Under	:	<input type="text" value="557"/>
<a href="#">Report Generation</a>	Address	:	<input type="text" value="Chaitanyapuri"/>
<a href="#">Log Out</a>	Phone No	:	<input type="text" value="82736735"/>
	<input type="button" value="Update"/>		<input type="button" value="Reset"/>

Output screen2:

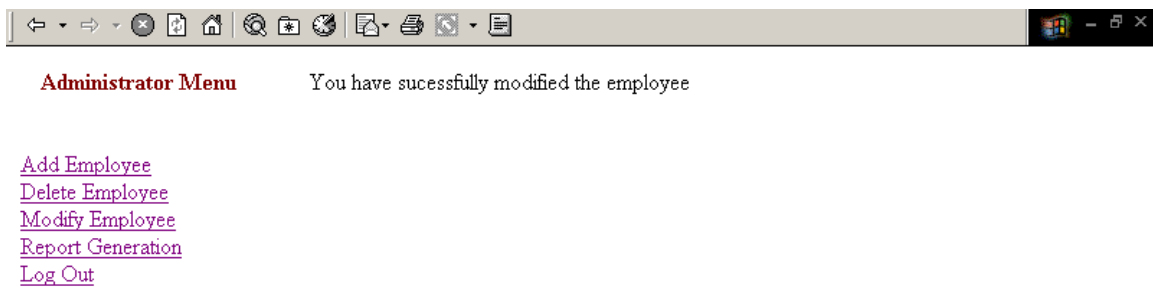


The screenshot shows a web browser window with a navigation bar at the top containing various icons. The main content is divided into two sections:

- Administrator Menu:** A list of links: [Add Employee](#), [Delete Employee](#), [Modify Employee](#), [Report Generation](#), and [Log Out](#).
- Employee Details to modify in DataBase:** A form with the following fields:
  - Employee Name :
  - Designation :
  - Works Under :
  - Address :
  - Phone No :

At the bottom of the form are two buttons:  and .

### Output screen3:



## View Employee



### Administrator Menu

- [Add Employee](#)
- [Delete Employee](#)
- [Modify Employee](#)
- [View Employee](#)
- [Report Generation](#)
- [Log Out](#)

### Employee Details

EMPID	EMP NAME	EMP ADDRESS	EMP PHONE NO	WORKS UNDER	DESIGNATION
557	Smitha	Saidabad	42371265	523	manager
520	Sudha	Chaitanyapuri	73472487	557	coordinator
501	Ashwin	Chaitanyapuri	52763828	557	employee
502	Surya	Kothapet	62533868	520	employee
518	Kranthi	Malakpet	32352578	523	manager
527	Ravali	Malakpet	76423842	523	manager
514	Rajesh	Kothapet	23737212	518	employee
504	Swapna	Amberpet	342452452	518	employee

Report generation:



**REPORT MENU**

[Daily Reports](#)

[Weekly Reports](#)

[Monthly Reports](#)

[Log Out](#)

Daily report:

**REPORT MENU**

- [Daily Reports](#)
- [Weekly Reports](#)
- [Monthly Reports](#)
- [Log Out](#)

**DAILY REPORT**

Select Date here   day      month      Year

Output Screen:



**REPORT MENU**

- [Daily Reports](#)
- [Weekly Reports](#)
- [Monthly Reports](#)
- [Log Out](#)

**DAILY REPORT**

**On Going works**

Work ID	Work Name	Work Done By
3	Conducting Seminar	520
5	Conducting Project Vivas	520

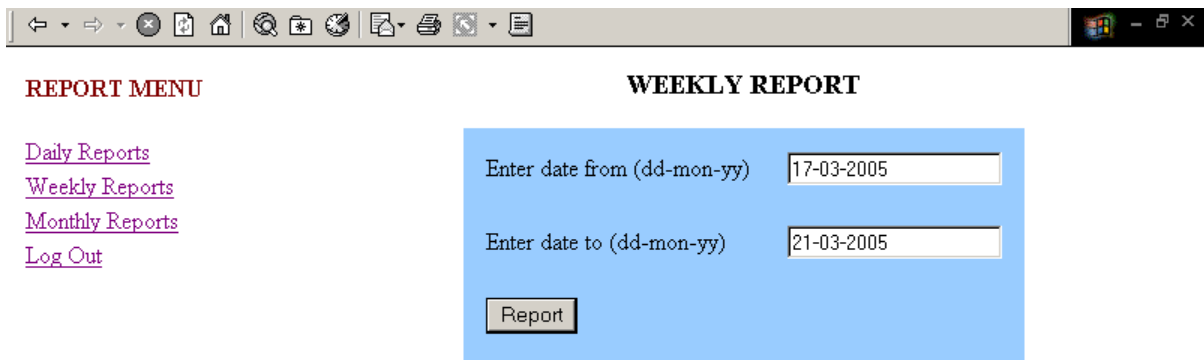
**Pending Works**

Work ID	Work Name	Work Done By
4	Visiting Colleges	520
8	Collecting Completed Projects	520

**Completed Works**

Work ID	Work Name	Work Done By
6	Issuance of Certificates	520

Weekly report:



**REPORT MENU**

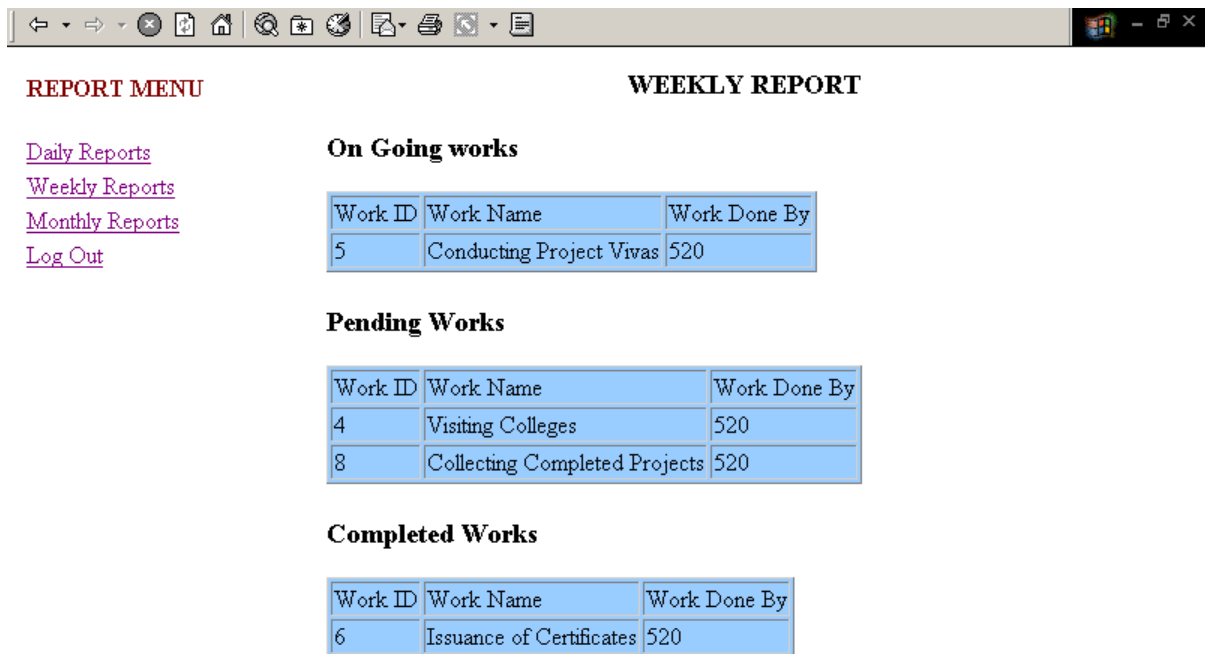
- [Daily Reports](#)
- [Weekly Reports](#)
- [Monthly Reports](#)
- [Log Out](#)

**WEEKLY REPORT**

Enter date from (dd-mon-yy)

Enter date to (dd-mon-yy)

Output Screen:



**REPORT MENU**

[Daily Reports](#)  
[Weekly Reports](#)  
[Monthly Reports](#)  
[Log Out](#)

**WEEKLY REPORT**

**On Going works**

Work ID	Work Name	Work Done By
5	Conducting Project Vivas	520

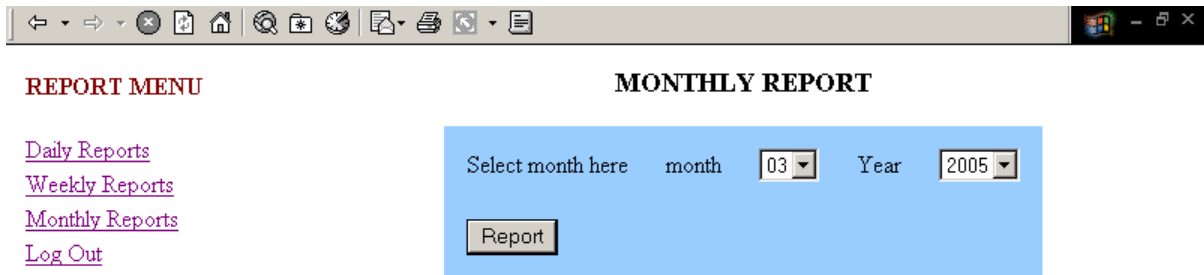
**Pending Works**

Work ID	Work Name	Work Done By
4	Visiting Colleges	520
8	Collecting Completed Projects	520

**Completed Works**

Work ID	Work Name	Work Done By
6	Issuance of Certificates	520

## Monthly Report:



REPORT MENU

[Daily Reports](#)  
[Weekly Reports](#)  
[Monthly Reports](#)  
[Log Out](#)

**MONTHLY REPORT**

Select month here   month    Year

Output Screen:



**REPORT MENU**

- [Daily Reports](#)
- [Weekly Reports](#)
- [Monthly Reports](#)
- [Log Out](#)

**MONTHLY REPORT**

**On Going works**

Work ID	Work Name	Work Done By
3	Conducting Seminar	520
5	Conducting Project Vivas	520

**Pending Works**

Work ID	Work Name	Work Done By
4	Visiting Colleges	520
8	Collecting Completed Projects	520

**Completed Works**

Work ID	Work Name	Work Done By
6	Issuance of Certificates	520

## AGM Menu:



### **AGM MENU**

[Add Work](#)

[Assign Work](#)

[Delete Work](#)

[Work Status](#)

[Search](#)

[Log Out](#)

Add work:



**AGM MENU**

- [Add Work](#)
- [Assign Work](#)
- [Delete Work](#)
- [Work Status](#)
- [Search](#)
- [Log Out](#)

## Add Work

<b>Work Id</b>	:	<input type="text" value="09"/>
<b>Work Name</b>	:	<input type="text" value="jgf"/>
<b>Duration(Days)</b>	:	<input type="text" value="7"/>
		<input type="button" value="Add"/> <input type="button" value="reset"/>

## Output Screen:



### **AGM MENU**

work is added to the database

[Add Work](#)

[Assign Work](#)

[Delete Work](#)

[Work Status](#)

[Search](#)

[Log Out](#)

Assign Work:

**AGM MENU**

- [Add Work](#)
- [Assign Work](#)
- [Delete Work](#)
- [Work Status](#)
- [Search](#)
- [Log Out](#)

**ASSIGN WORK**

**Employees working under you and works that are to be assigned**

Employees working under you

Work ID

Output Screen:



**AGM MENU**

[Add Work](#)

[Assign Work](#)

[Delete Work](#)

[Work Status](#)

[Search](#)

[Log Out](#)

**Work is assigned sucessfully to 527**

Work status:

**AGM MENU**

**OVERALL WORK STATUS**

- [Add Work](#)
- [Assign Work](#)
- [Delete Work](#)
- [Work Status](#)
- [Search](#)
- [Log Out](#)

WORKID	WORK NAME	WORK ASSIGNED TO	WORK ASSIGNED BY	START DATE	WORK STATUS
5	Conducting Project Vivas	557	523	20-MAR-05	startstage
6	Issuance of Certificates	557	523	20-MAR-05	assigned
8	Collecting Completed Projects	557	523	20-MAR-05	assigned
1	Visit to ATC	527	523	21-MAR-05	startstage
7	Managing Projects	557	523	20-MAR-05	startstage
3	Conducting Seminar	518	523	20-MAR-05	startstage

Search menu:

## SEARCH MENU

### STATUS

[Completed works](#)

[Pending Works](#)

[Ongoing Work](#)

[Employee Status](#)

### STATUS THROUGH DATE

[Enter Date](#)

[AGM Menu](#)

[Log Out](#)

Completed works:



**SEARCH MENU**

**COMPLETED WORKS**

**STATUS**

[Completed works](#)

[Pending Works](#)

[Ongoing Work](#)

[Employee Status](#)

**STATUS THROUGH DATE**

[Enter Date](#)

[AGM Menu](#)

[Log Out](#)

**Completed Work Details**

Work ID	Work Name	Work Done By
6	Issuance of Certificates	520

Pending works:

**SEARCH MENU**

**PENDING WORKS**

**STATUS**

[Completed works](#)

[Pending Works](#)

[Ongoing Work](#)

[Employee Status](#)

**STATUS THROUGH DATE**

[Enter Date](#)

[AGM Menu](#)

[Log Out](#)

**Pending Work Details**

Work ID	Work Name	Work Done By
4	Visiting Colleges	520
8	Collecting Completed Projects	520

Ongoing works:

**SEARCH MENU**

**ONGOING WORKS**

**STATUS**

[Completed works](#)

[Pending Works](#)

[Ongoing Work](#)

[Employee Status](#)

**STATUS THROUGH DATE**

[Enter Date](#)

[AGM Menu](#)

[Log Out](#)

**OnGoing Work Details**

Work ID	Work Name	Work Done By
3	Conducting Seminar	520
5	Conducting Project Vivas	520

Employee status:



**SEARCH MENU**

**STATUS**

[Completed works](#)

[Pending Works](#)

[Ongoing Work](#)

[Employee Status](#)

**STATUS THROUGH DATE**

[Enter Date](#)

[AGM Menu](#)

[Log Out](#)

Select Employee Name

Output Screen:

**SEARCH MENU**

**STATUS**

[Completed works](#)

[Pending Works](#)

[Ongoing Work](#)

[Employee Status](#)

**STATUS THROUGH DATE**

[Enter Date](#)

[AGM Menu](#)

[Log Out](#)

**WORK STATUS**

Work ID	Work Name	Work status
1	a	ongoing
1	a	pending
1	a	c
2	b	ongoing
5	e	c
24	y4	completed
26	y6	pending
34	c2	ongoing
90	x3	completed

Enter Date:

**SEARCH MENU**

**DAILY REPORT**

**STATUS**

[Completed works](#)

[Pending Works](#)

[Ongoing Work](#)

[Employee Status](#)

**STATUS THROUGH DATE**

[Enter Date](#)

[AGM Menu](#)

[Log Out](#)

Select Date here   day      month      Year  

Output Screen:

**SEARCH MENU**

**DAILY REPORT**

**STATUS**

[Completed works](#)

[Pending Works](#)

[Ongoing Work](#)

[Employee Status](#)

**STATUS THROUGH DATE**

[Enter Date](#)

[AGM Menu](#)

[Log Out](#)

**On Going works**

Work ID	Work Name	Work Done By
3	Conducting Seminar	520
5	Conducting Project Vivas	520

**Pending Works**

Work ID	Work Name	Work Done By
4	Visiting Colleges	520
8	Collecting Completed Projects	520

**Completed Works**

Work ID	Work Name	Work Done By
6	Issuance of Certificates	520

**SEARCH MENU**

**DAILY REPORT**

**STATUS**

[Completed works](#)

[Pending Works](#)

[Ongoing Work](#)

[Employee Status](#)

**STATUS THROUGH DATE**

[Enter Date](#)

[AGM Menu](#)

[Log Out](#)

Select Date here   day      month      Year

## Manager Menu:



### MANAGER\ COORDINATOR MENU

[Assign Work](#)

[View Work](#)

[Status Update](#)

[Work Status](#)

[Search](#)

[Log Out](#)

## Assign work:

**MANAGER\ COORDINATOR MENU**

- [Assign Work](#)
- [View Work](#)
- [Status Update](#)
- [Work Status](#)
- [Search](#)
- [Log Out](#)

**ASSIGN WORK**

**Employees working under you and works that r to be assigned**

Employees working under you

Work ID

## Output Screen:



**MANAGER\  
COORDINATOR  
MENU**

[Assign Work](#)

[View Work](#)

[Status Update](#)

[Work Status](#)

[Search](#)

[Log Out](#)

**Work is assigned sucessfully to 501**

## View Work:



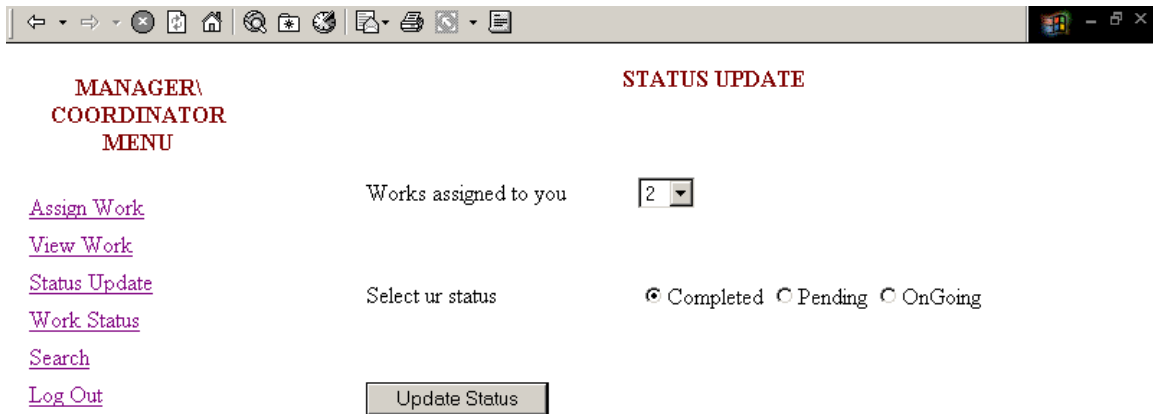
### MANAGER\ COORDINATOR MENU

- [Assign Work](#)
- [View Work](#)
- [Status Update](#)
- [Work Status](#)
- [Search](#)
- [Log Out](#)

### WORKS ASSIGNED TO YOU

WORKID	WORK NAME	WORK ASSIGN BY	START DATE	DURATION
5	Conducting Project Vivas	523	20-MAR-05	15
6	Issuance of Certificates	523	20-MAR-05	8
8	Collecting Completed Projects	523	20-MAR-05	5
7	Managing Projects	523	20-MAR-05	90

## Status Update:



The screenshot shows a web browser window with a toolbar at the top. The page content is divided into two main sections: a navigation menu on the left and a status update form on the right.

**MANAGER\ COORDINATOR MENU**

- [Assign Work](#)
- [View Work](#)
- [Status Update](#)
- [Work Status](#)
- [Search](#)
- [Log Out](#)

**STATUS UPDATE**

Works assigned to you

Select ur status  Completed  Pending  OnGoing

## Output Screen:



**MANAGER\  
COORDINATOR  
MENU**

**Status Update**

[Assign Work](#)  
[View Work](#)  
[Status Update](#)  
[Work Status](#)  
[Search](#)  
[Log Out](#)

**Your status is updated**

Employee menu:



**EMPLOYEE MENU**

[View Work](#)

[Status Update](#)

[Log Out](#)

## Conclusion:

Though the proposed system is very effective the person should be able to work with the computer. The person may or may not check the message he has received. Some times the server may fail by which the work cannot be assigned to the right person

## Future Enhancements:

The discussions about a work among the employees can be maintained (as a history) or stored in the database.

Even the work can be divided into sub groups and can be distributed among the employees

Even though this project has several benefits, it has some disadvantages, which do not affect the operational cost and performance, to overcome these disadvantages some features may be added:

- 1) A superior can assign work only to those employees who are working under him, but not to the sub-employees. Instead, if the superior assign the work directly to his sub-employees, then the “Work Flow Management” would become more efficient.
- 2) The Project can further be enhanced by providing an “ONLINE MESSAGE TRANSFER SYSTEM” through which an employee can clarify his doubts any time without moving from his place

