**CHAPTER 1**

# INTRODUCTION

**1.1. INTRODUCTION**

The world itself has become a global village of technology today, especially with the computer technology. With the advent of new technology, the world is now witnessing an easy way to keep information / data. Hence the introduction of computer is one of the greatest challenges facing man to day.

Recipe Management System is software developed for people from all aspects of life in helping them enhancing their culinary skills. A Recipe Management System, **is an IT application for managing the recipes of a particular manufacturing unit or factory**. It is a system which is supposed to manage the entire set of production recipes, where it is specified how the production equipment should perform their operations; it encompasses and contains individual equipment recipes and that of any subsidiary/additional processes, within the actual production process.

The process of new product development (NPD) and recipe management in food industries has changed drastically in recent years. Today’s food markets are impatient for **new product launches and improved recipes**. The R&D department, QA unit and product recipe management is under constant pressure to deliver new products that of the market. This means continually finding ways to **shorten the time-to-market cycle,** selecting cost-effective ingredients and maintaining a highly reliable reputation. Unfortunately, simple add-ons to your ERP system cannot provide the necessary flexibility and power to do this. And manually updated spreadsheets or small software applications spread around your company does not give your recipe management full control over the data or the development process.

Creating innovative and competitive products remains one of the major challenges in any food company. Adifo's recipe management solution, BESTMIX, supports businesses through each stage of the development process and allows users to **manage ingredients, recipes and products specs using secure and flexible methods**.

**1.2. PURPOSE**

The Recipe Management System has been designed to give the user as much freedom as possible in recipe input, but the ability to find what you want quickly is something you will never find in a cookbook. The users can locate recipes based on almost any criteria we can imagine. The Recipe Management System is to create a recipe which acts as a meal management software ‘book’ that allows you to access an unlimited number of recipes, add your own recipes, share recipes.

**1.3. EXISTING SYSTEM**

Most of the recipe management applications contain only those recipes specified by a cook and is uploaded by the admin. The users are allowed to browse and view the recipes. The present application does not allow any researcher to obtain a survey report based on the age group and professions. The present application does not clearly provide proper remedies for a disease in terms of the type of food he/she should intake.

**1.4. PROPOSED SYSTEM:**

In our application you can publish and maintain a library of recipes. Pictures and videos can be uploaded at the same time. This gives a clear picture or a view of the food to be prepared quickly. This application allows the users to find recipes based on health problems and also suggest nutrition tips for the same. It enables the admin to generate a report for a survey. The survey may include the category of people using it, and also based on the age group of people browsing the application.

**1.4.1. Advantages of Proposed System**



There are many other advantages to standard recipes:

• Controls food costs as managers know how much a standard recipe will yield so they generally can’t make to many or too few. Scheduling of equipment and staff is more efficient.

• Less supervision is required as the recipe tells you step by step how to do everything.

• If a head staff member is absent, lower member of staff can still perform the duties

of his head.

**1.5. FEASIBILITY STUDY**

Feasibility analysis begins once the goals are defined. It starts by generating broad possible solutions, which are possible to give an indication of what the new system should look like. This is where creativity and imagination are used. Analysts must think up new ways of doing things- generate new ideas. There is no need to go into the detailed system operation yet. The solution should provide enough information to make reasonable estimates about project cost and give users an indication of how the new system will fit into the organization. It is important not to exert considerable effort at this stage only to find out that the project is not worthwhile or that there is a need significantly change the original goal.

Feasibility of a new system means ensuring that the new system, which we are going to implement, is efficient and affordable. There are various types of feasibility to be determined. They are,

**1.5.1. Economically Feasibility**

Development of this application is highly economically feasible. The only thing to be done is making an environment with an effective supervision. It is cost effective in the sense that has eliminated the paper work completely. The system is also time effective because the calculations are automated which are made at the end of the month or as per the user requirement.

**1.5.2. Technical feasibility**

The technical requirement for the system is economic and it does not use any other additional Hardware and software. Technical evaluation must also assess whether the existing systems can be upgraded to use the new technology and whether the organization has the expertise to use it.

**1.5.3. Operational Feasibility**

The system working is quite easy to use and learn due to its simple but attractive interface. User requires no special training for operating the system. Technical performance include issues such as determining whether the system can provide the right information for the Department personnel student details, and whether the system can be organized so that it always delivers this information at the right place and on time using intranet services. Acceptance revolves around the current system and its personnel.

**1.6. SYSTEM SPECIFICATION**

**1.6.1. HARDWARE REQUIREMENTS (Minimum Requirement)**

* **Minimum RAM:** 4 GB
* **Hard Disk:** 128 GB
* **Processor:** Intel Pentium 4(1.50 GHZ) or above

**1.6.2. Technology used:**

* Programming Language : Java
* IDE : Eclipse
* Backend : Oracle 10g
* Backend Connectivity : JDBC (Java Data Base Connectivity)
* Web Technologies for UI:
  + HTML 5
  + CSS
  + JavaScript
  + Spring HTTP Commands
  + Spring JDBC Templates
  + Angular Js

**1.7. DEFINITION OF TERMS**

**1.7.1. COMPUTER:** A computer can be finely defined as an automatic electronic machine, that can accept raw fact or data through an input device like the keyboard or mouse: stores, processes the data in the system unit using a particular program and finally supplying the result of information through an output device such as monitor in the form of pictures or texts on the screen called the soft copy or a printer in form of printed paper called the hard copy.

**1.7.2.** [**DATA:**](http://www.pdfcomplete.com/cms/hppl/tabid/108/Default.aspx?r=q8b3uige22) [This is a raw materia](http://www.pdfcomplete.com/cms/hppl/tabid/108/Default.aspx?r=q8b3uige22)l facts that has not been processed. Data are also facts about an organization or institution, and its daily transactions.

**1.7.3. INFORMATION:** These are data that has been processed and is meaningful to the end user.

**1.7.4. COMPUTER-BASED:** It is also referred to as computerized. It is the art of using computer system to carry out task.

**1.7.5. RECIPE MANAGEMENT:** The Recipe Management System is to create a recipe which acts as a meal management software ‘book’ that allows you to access an unlimited number of recipes, add your own recipes, share recipes.

**1.7.6. SYSTEM:** A system is a group of inter-related components working together towards a common goal by accepting input and producing output in an organized transformation process. It is also an organized collection of people; machine and method required to accomplished a set of specific functions.

**1.7.7. DATABASE:** Are not merely collections of files, rather the database is a centre source of data meant to be shared by many users for a variety of applications.

**1.7.8. JAVA:** Java is a general-purpose [computer programming language](https://en.wikipedia.org/wiki/Programming_language) that is [concurrent](https://en.wikipedia.org/wiki/Concurrent_computing), [class-based](https://en.wikipedia.org/wiki/Class-based_programming), [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming),and specifically designed to have as few implementation dependencies as possible.

**1.7.9. Oracle:** An Oracle database is a collection of data treated as a unit. The purpose of a database is to store and retrieve related information. A database server is the key to solving the problems of information management.

**1.8. OVERVIEW**

Recipe Management System basically has three main modules for proper functioning:

First module is the admin, which has right for creating space for new recipe and viewing the details of recipe adding comments.

Second module is handled by the editor who has a right of editing recipes,and procedures added by admin.

Third module is handled by the user who can view all recipes and procedures and comment on them.

**1.9. SCOPE OF THE PROJECT:**

This project is the prototype of Recipe Management System using Eclipse IDE used to develop any dynamic web application. Traditional methods of cookbooks were not user friendly, whereas our application today can instantly search and provide user the result regarding any food or health remedies and also let you to add your own recipe.

**CHAPTER 2**

**LITERATURE REVIEW**

According to Cobuild learners dictionary Attendance is the fact that someone is present at an event or go regularly to an institution, or the attendance at an event is the number of people who are present at it. Furthermore, if someone is in attendance of a place or event, they are present.

According to Turban Mclean. Wetherbe (1998) from their book Information Technology For Management, it was stated that the term information system is a collection of component that collects, processes, stores, analyses and disseminate information for a specific purpose, like any other system. An information system includes inputs (data, instructions and output reports, calculation).

[According to A.N Aniekwu an](http://www.pdfcomplete.com/cms/hppl/tabid/108/Default.aspx?r=q8b3uige22)d Jone Akpodvado (1999) from their book fundamental of practical computer a computer is a machine that performs calculation and processes automatically at high speed according to prescribed sequence of operations to an electronic machine also to one of a mechanic, analogue or other variety.

Geoffrey K, Nick W, Paul C and John E (1987) added that computer system consist of individual element working together with the common aim of processing data to process into information.

C.S French (1996) in his book Computer Science asserts that a record is information relating to one person, production e.t.c in database, a record contains information about one person.

Aghawra E.D (2002) Operate Your Computer asserts that computer areelectronic devices that can be programmed to accept (input), program the input, store it and output useful information as technological society becomes more complex.

[Akin Fapohide (2005) says](http://www.pdfcomplete.com/cms/hppl/tabid/108/Default.aspx?r=q8b3uige22) that, computer is a machine that follows instructions in order to process data, solve specific problem or accomplished task is referred to as program. Attendance Management System facilitates the attendance information of a particular student in a particular class. The information is sorted by the operators, which will be provided by the lecturer for a particular class. This system will also help in evaluating eligibility criteria of a student to sit for an examination.

**2.1 DATA PROCESSING SYSTEM**

Data processing system are conned with the transaction, handling and recordkeeping, usually for a particular functional area. Data are entered and stored in a file format, the stored files are updated during routine processing. The major drawback of data processing system is that, they are flexible and control/ accommodates data, information needs that are not already built into the system.

[**2.2 INFORMATION S**](http://www.pdfcomplete.com/cms/hppl/tabid/108/Default.aspx?r=q8b3uige22)**YSTEM CAPABILITIES**

Information system includes all use of computer that support administrative aspect of an organization, example includes, application of payroll system, registration system, hospital patient bill system, registration combination of hardware, software , peoples procedure for an information system. A computer based information system provides an organization with data processing. Information system has four capabilities as a computer system, which includes the following: input, processing, storage and output.

**2.3 WORKING OF PRESENT SYSTEM**

Most of the recipe management applications contain only those recipes specified by a cook and is uploaded by the admin. The users are allowed to browse and view the recipes. The present application does not allow any researcher to obtain a survey report based on the age group and professions. The present application does not clearly provide proper remedies for a disease interms of the type of food he/she should intake.

**Disadvantages of Present Working System**

* Not User Friendly: The existing system is not user friendly because the retrieval of data is very slow and data is not maintained efficiently.
* [Difficulty in searching reci](http://www.pdfcomplete.com/cms/hppl/tabid/108/Default.aspx?r=q8b3uige22)pes and locating them: We require more time in searching recipes and locating them so search and retrieval of data is difficult.
* Lots of paperwork: Existing system requires lot of paper work. To write down and maintain all those papers is the most difficult task.

**2.4 PROPOSED SYSTEM**

In our application you can publish and maintain a library of recipes. Pictures and videos can be uploaded at the same time. This gives a clear picture or a view of the food to be prepared quickly. This application allows the users to find recipes based on health problems and also suggest nutrition tips for the same. It enables the admin to generate a report for a survey. The survey may include the category of people using it, and also based on the age group of people browsing the application

**Characteristics of the proposed system**

User Friendly:-The proposed system is user friendly because the retrieval and storing of data is fast and data is maintained.

Efficiently:- Moreover the graphical user interface is provided in the proposed system, which provides user to deal with the system very easily.

Very less paper work:- The proposed system requires very less paper work. All the data is feted into the computer immediately and recipes can be fetched through computers. Moreover work becomes very easy because there is no need to keep data on papers.

Computer operator control:- Computer operator control will be there so no chance of errors. Moreover storing and retrieving of information is easy. So work can be done speedily and in time.

Recipe Management System allows the user to upload a recipe, images, videos and also rate recipes. The uploaded recipe will be approved by an expert chef within a day and then it will be made available to the users. In future, it can be used to add a feature where one can provide a diet chart based on the user’s weight, height and gender.

**CHAPTER 3**

**PROBLEM DESCRIPTION**

**3.1 INTRODUCTION**

Recipe Management System is software developed for people from all aspects of life in helping them enhancing their culinary skills. A Recipe Management System, **is an IT application for managing the recipes of a particular manufacturing unit or factory**. It is a system which is supposed to manage the entire set of production recipes, where it is specified how the production equipment should perform their operations; it encompasses and contains individual equipment recipes and that of any subsidiary/additional processes, within the actual production process.

**3.2 PROBLEM STATEMENT**

In this application one can publish and maintain a library of recipes. Pictures and videos can be uploaded at the same time. This gives a clear picture or a view of the food to be prepared quickly. This application allows the users to find recipes based on health problems and also suggest nutrition tips for the same. It enables the admin to generate a report for a survey. The survey may include the category of people using it, and also based on the age group of people browsing the application.

**3.3 MODULE DESCRIPTION**

The system should be designed in such a way that only authorized people should be allowed to access some particular modules. The recipes should be added by only administrators and no one else. The user should always be in control of the application and not the vice versa.

The user interface should be consistent so that the user can handle the application with ease and speed. The application should be visually, conceptually clear.

**3.3.1 ADMINISTRATOR MODULE:**

* **Recipe Addition:**

This is a main functionality where addition of recipes is done. An Administrator is the one who adds recipes.

* **Recipe Details:**
* This is where details of various recipes get tabulated. The main entities displayed are the name of the recipe, ingredients and procedure to make a recipe.
* The recipe details are made easily accessible for the users to have a glimpse of the recipe and try their hands out if they are interested in preparing a recipe.
* **Recipe Type details:**
* This is where a type of the recipe gets mentioned. Besides mentioning the type the description of the type and chief ingredients of the type and place of origin of the type are also specified.
* It will help us to gain knowledge about a recipe’s type and the common food of a particular place besides getting to know the chief ingredients and place of origin of a particular recipe.
* A person is also able to add recipes, delete recipes and comment on a recipe being an administrator. A person being an administrator can have complete control over the portal and on all actions over it.

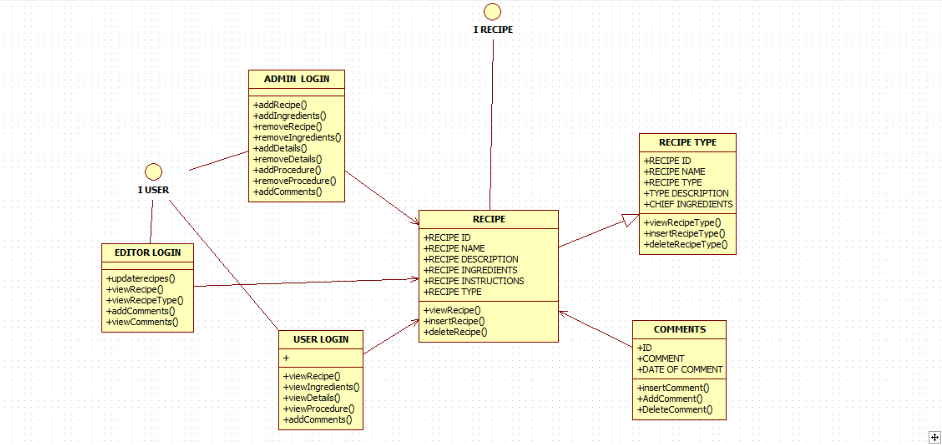
**3.3.2 USER’S MODULE:**

* **Recipe details:**
* It facilitates the users to view various recipes as per their choice. This will authenticate the user before making the entry into the portal.
* **Comment details:**
* Various Comments for various recipes are given by each user individually. A user besides commenting on a recipe can also view comments made by other users for a particular recipe.

**3.4 CLASS DIAGRAM:**

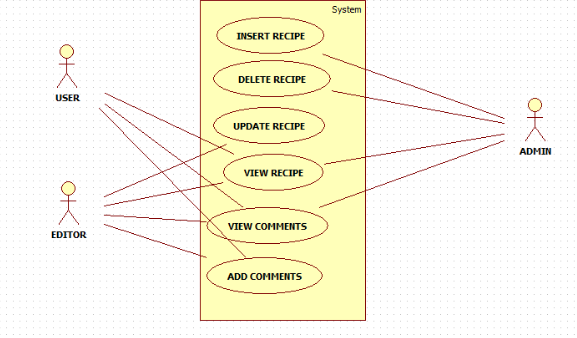
In software engineering, a class diagram in the Unified Modelling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.

The class diagram is the main building block of [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming) modelling. It is used both for general [conceptual modelling](https://en.wikipedia.org/wiki/Conceptual_model) of the systematics of the application, and for detailed modelling translating the models into [programming code](https://en.wikipedia.org/wiki/Programming_code). Class diagrams can also be used for [data modeling](https://en.wikipedia.org/wiki/Data_modeling).The classes in a class diagram represent both the main elements, interactions in the application, and the classes to be programmed.



**3.5 USE CASE DIAGRAM:**

**Use case diagrams** are usually referred to as [behaviour diagrams](http://www.uml-diagrams.org/uml-25-diagrams.html#behavior-diagram) used to describe a set of actions ([use cases](http://www.uml-diagrams.org/use-case.html)) that some system or systems ([subject](http://www.uml-diagrams.org/use-case-subject.html)) should or can perform in collaboration with one or more **external users** of the system ([actors](http://www.uml-diagrams.org/use-case-actor.html)). Each use case should provide some observable and valuable result to the actors or other stakeholders of the system.



**3.6 ER DIAGRAM:**

# Entity

Entities are represented by means of rectangles. Rectangles are named with the entity set they represent.

# Attributes

Attributes are the properties of entities. Attributes are represented by means of ellipses. Every ellipse represents one attribute and is directly connected to its entity *rectangle*.

# Relationship

Relationships are represented by diamond-shaped box. Name of the relationship is written inside the diamond-box. All the entities *rectangles* participating in a relationship, are connected to it by a line.

# Binary Relationship and Cardinality

A relationship where two entities are participating is called a binary relationship. Cardinality is the number of instance of an entity from a relation that can be associated with the relation.

**One-to-one** − When only one instance of an entity is associated with the relationship, it is marked as '1:1'. The following image reflects that only one instance of each entity should be associated with the relationship. It depicts one-to-one relationship.

**One-to-many** − When more than one instance of an entity is associated with a relationship, it is marked as '1:N'. The following image reflects that only one instance of entity on the left and more than one instance of an entity on the right can be associated with the relationship.

TYPE

USER

CAN HAVE

COMMENTS

CAN HAVE

RECIPE

CAN VIEW

**CHAPTER 4**

**METHODOLOGY / EXPERIMENTAL SETUP**

**4.1 ABSTRACT**

Recipe Management System is software developed for people from all aspects of life in helping them enhancing their culinary skills. A Recipe Management System, **is an IT application for managing the recipes of a particular manufacturing unit or factory**. It is a system which is supposed to manage the entire set of production recipes, where it is specified how the production equipment should perform their operations; it encompasses and contains individual equipment recipes and that of any subsidiary/additional processes, within the actual production process.

**4.2 DATABASE DESIGN**

**4.2.1 LOGIN TABLE:**

* To create a login details for the table.

|  |  |  |  |
| --- | --- | --- | --- |
| S.No | Field Name | Data type | Description |
| 1. | User Name | Text | Store user name for checking correct user name |
| 2. | Password | Text | Store password corresponding to user name |
| 3. | User Type | Text | Specifies the user type |

**4.2.2 RECIPE TABLE:**

To have a view of various recipes and their procedures along with their ingredients.

|  |  |  |  |
| --- | --- | --- | --- |
| Field Name | Data type | Constraint | Description |
| Recipe Id | Varchar(20) | Primary key | Defines separate unique Id |
| Recipe Name | Varchar(30) | Not Null | Recipe’s name |
| Recipe Type | Varchar(30) | Not Null | Type of a recipe. |

**4.2.3 RECIPE TYPE TABLE:**

To create table for Student personal details for our department.

|  |  |  |  |
| --- | --- | --- | --- |
| Fields | Data type | Constraint | Description |
| Recipe Type(Roll No) | Varchar(15) | Primary Key | Type of recipe |
| Type Description | Varchar(30) | Not Null | Description of type |
| Chief Ingredients | Varchar(15) | Not Null | Ingredients of the type. |
| Place Of Origin | Number | Not Null | Place of orgin of the type. |

**4.2.4 RECIPE PROCEDURE TABLE:**

|  |  |  |  |
| --- | --- | --- | --- |
| Field Name | Data type | Constraint | Description |
| Recipe Id | Varchar(20) | Primary key | Defines separate unique Id |
| Recipe Ingredients | Varchar(30) | Not Null | Recipe’s Ingredients |
| Recipe Procedure | Varchar(30) | Not Null | Procedure of a recipe. |

**4.2.4 RECIPE COMMENTS TABLE:**

|  |  |  |  |
| --- | --- | --- | --- |
| Field Name | Data type | Constraint | Description |
| Recipe Id | Varchar(20) | Primary key | Defines separate unique Id |
| User Id | Varchar(30) | Not Null | User’s Unique login Id |
| Recipe Comment | Varchar(30) | Not Null | Comments given to a recipe. |
| Type Of Comment | Varchar(30) | Not Null | Indicates whether comment given is positive or negative. |

**4.3. SOFTWARE REQUIREMENTS SPECIFICATION:**

**4.3.1 Apache Tomcat Installation**

Apache Tomcat is an open-source Web server and servlet container. It requires a Java Standard Edition Runtime Environment (JRE) version 6 or later. The chit fund system developed uses an Apache Tomcat server version 8.5. Apache Tomcat’s server is called Catalina that contains all the specification for the server that is modified to the needs. Coyote connector component is used to get the response to and fro the server using JK protocol.

**STEPS TO INSTALL:**

1. Download then install the JRE from http://www.oracle.com/technetwork/java/javase/downloads/index.html
2. Then download and install the Apache Tomcat, a binary distribution of tomcat from <http://tomcat.apache.org/>
3. Unpack the binary distribution so that it resides in its own     directory (conventionally named "apache-tomcat-[version]").
4. Configure Environment Variables
   * 1. Set CATALINA\_HOME and CATALINA\_BASE (optional).
     2. The CATALINA\_HOME environment variable should be set to the location of the root directory of the "binary" distribution of Tomcat.
     3. The CATALINA\_BASE environment variable specifies location of the root directory of the "active configuration" of Tomcat. It is optional. It defaults to be equal to CATALINA\_HOME.
5. Set JRE\_HOME or JAVA\_HOME.

**5.1.1** The JRE\_HOME variable is used to specify location of a JRE. The JAVA\_HOME variable is used to specify location of a JDK.

**5.1.2** Using JAVA\_HOME provides access to certain additional start-up options that are not allowed when JRE\_HOME is used. If both JRE\_HOME and JAVA\_HOME are specified, JRE\_HOME is used. The best place to include these variables is a "setenv" script.

1. Other Variables like CATALINA\_OPTS are optional to set with. It allows specification of additional options for the java command to start tomcat.
2. Start Tomcat
   * 1. On Windows

                             %CATALINA\_HOME%\bin\startup.bat

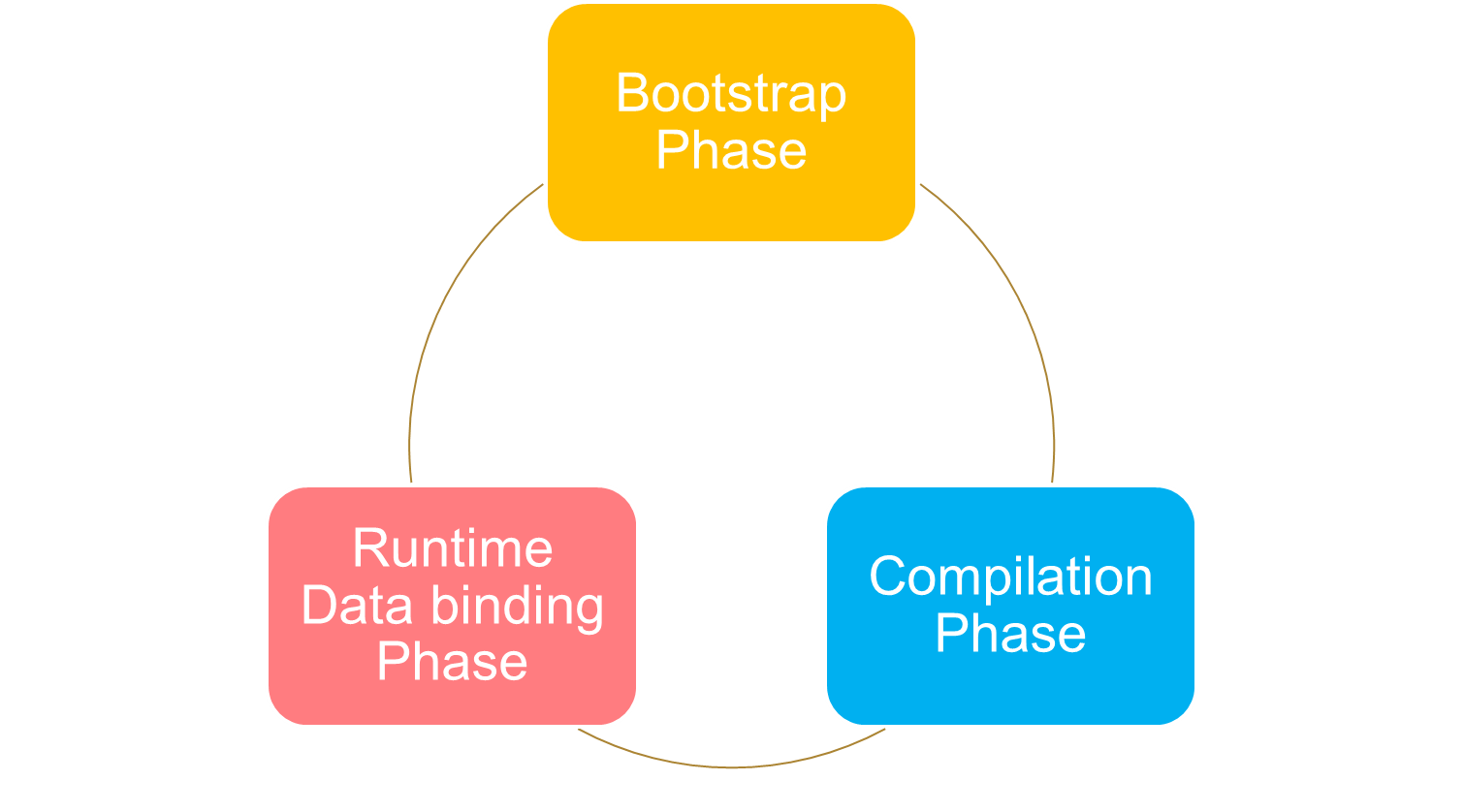
                     Or

                      %CATALINA\_HOME%\bin\catalina.bat start

                  Or

                    $CATALINA\_HOME/bin/catalina.sh start

**4.4. Angular JS**

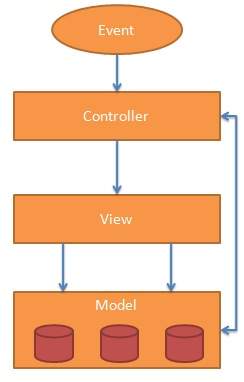
Angular is a client side JavaScript framework for dynamically adding interactivity for HTML. Data-binding is an automatic way of updating the view whenever the model changes, as well as updating the model whenever the view changes. It saves developer’s productivity by reducing considerable amount of code for manipulating, traversing and listening to DOM. Angular JS has a built-in dependency injection subsystem that helps the developer by making the application easier to develop, understand, and test.

# 4.4.1 Angular JS - MVC Architecture

**M**odel **V**iew **C**ontroller or MVC as it is popularly called, is a software design pattern for developing web applications. A Model View Controller pattern is made up of the following three parts −

* **Model** − It is the lowest level of the pattern responsible for maintaining data.
* **View** − It is responsible for displaying all or a portion of the data to the user.
* **Controller** − It is a software Code that controls the interactions between the Model and View.

MVC is popular because it isolates the application logic from the user interface layer and supports separation of concerns. The controller receives all requests for the application and then works with the model to prepare any data needed by the view. The view then uses the data prepared by the controller to generate a final presentable response. The MVC abstraction can be graphically represented as follows.



## The Model

The model is responsible for managing application data. It responds to the request from view and to the instructions from controller to update itself.

## The View

A presentation of data in a particular format, triggered by the controller's decision to present the data. They are script-based template systems such as JSP, ASP, PHP and very easy to integrate with AJAX technology.

## The Controller

The controller responds to user input and performs interactions on the data model objects. The controller receives input, validates it, and then performs business operations that modify the state of the data model.

An AngularJ S application consists of following three important parts −

* **ng-app** − This directive defines and links an Angular JS application to HTML.
* **ng-model** − This directive binds the values of Angular JS application data to HTML input controls.
* **ng-bind** − This directive binds the Angular JS Application data to HTML tags.

## 4.4.2 Steps to create Angular JS Application

### Step 1 − Load framework

Being a pure JavaScript framework, it can be added using <Script> tag.

<script src = "https://ajax.googleapis.com/ajax/libs/angularjs/1.3.14/angular.min.js">

</script>

### Step 2 − Define AngularJS Application using ng-app directive

<div ng-app = "">

...

</div>

### Step 3 − Define a model name using ng-model directive

<p>Enter your Name: <input type = "text" ng-model = "name"></p>

### Step 4 − Bind the value of above model defined using ng-bind directive.

<p>Hello <span ng-bind = "name"></span>!</p>

## 4.4.3 Steps to run Angular JS Application

Use above mentioned three steps in an HTML page.

<html>

<head>

<title>AngularJS First Application</title>

</head>

<body>

<h1>Sample Application</h1>

<div ng-app = "">

<p>Enter your Name: <input type = "text" ng-model = "name"></p>

<p>Hello <span ng-bind = "name"></span>!</p>

</div>

<script src = "https://ajax.googleapis.com/ajax/libs/angularjs/1.3.14/angular.min.js"></script>

</body>

</html>

## 4.4.4 How AngularJS integrates with HTML

* ng-app directive indicates the start of AngularJS application.
* ng-model directive then creates a model variable named "name" which can be used with the html page and within the div having ng-app directive.
* ng-bind then uses the name model to be displayed in the html span tag whenever user input something in the text box.
* Closing</div> tag indicates the end of AngularJS application.

AngularJS directives are used to extend HTML. These are special attributes starting with ng- prefix. We're going to discuss following directives −

* **ng-app** − This directive starts an AngularJS Application.
* **ng-init** − This directive initializes application data.
* **ng-model** − This directive binds the values of AngularJS application data to HTML input controls.
* **ng-repeat** − This directive repeats html elements for each item in a collection.

## ng-app directive

ng-app directive starts an AngularJS Application. It defines the root element. It automatically initializes or bootstraps the application when web page containing AngularJS Application is loaded. It is also used to load various AngularJS modules in AngularJS Application. In following example, we've defined a default AngularJS application using ng-app attribute of a div element.

<div ng-app = "">

...

</div>

## ng-init directive

ng-init directive initializes an AngularJS Application data. It is used to put values to the variables to be used in the application. In following example, we'll initialize an array of countries. We're using JSON syntax to define array of countries.

<div ng-app = "" ng-init = "countries = [{locale:'en-US',name:'United States'}, {locale:'en-GB',name:'United Kingdom'}, {locale:'en-FR',name:'France'}]">

...

</div>

## ng-model directive

This directive binds the values of AngularJS application data to HTML input controls. In following example, we've defined a model named "name".

<div ng-app = "">

...

<p>Enter your Name: <input type = "text" ng-model = "name"></p>

</div>

## ng-repeat directive

ng-repeat directive repeats html elements for each item in a collection. In following example, we've iterated over array of countries.

<div ng-app = "">

...

<p>List of Countries with locale:</p>

<ol>

<li ng-repeat = "country in countries">

{{ 'Country: ' + country.name + ', Locale: ' + country.locale }}

</li>

</ol>

</div>

**4.4.5 Angular Js – HTML DOM**

Following directives can be used to bind application data to attributes of HTML DOM Elements.

|  |  |  |
| --- | --- | --- |
| **Sr.No.** | **Name** | **Description** |
| 1 | ng-disabled | disables a given control. |
| 2 | ng-show | shows a given control. |
| 3 | ng-hide | hides a given control. |
| 4 | ng-click | represents a AngularJS click event. |

## ng-disabled directive

Add ng-disabled attribute to a HTML button and pass it a model. Bind the model to an checkbox and see the variation.

<input type = "checkbox" ng-model = "enableDisableButton">Disable Button

<button ng-disabled = "enableDisableButton">Click Me!</button>

## ng-show directive

Add ng-show attribute to a HTML button and pass it a model. Bind the model to an checkbox and see the variation.

<input type = "checkbox" ng-model = "showHide1">Show Button

<button ng-show = "showHide1">Click Me!</button>

## ng-hide directive

Add ng-hide attribute to a HTML button and pass it a model. Bind the model to an checkbox and see the variation.

<input type = "checkbox" ng-model = "showHide2">Hide Button

<button ng-hide = "showHide2">Click Me!</button>

## ng-click directive

Add ng-click attribute to a HTML button and update a model. Bind the model to html and see the variation.

<p>Total click: {{ clickCounter }}</p>

<button ng-click = "clickCounter = clickCounter + 1">Click Me!</button>

**4.5 Spring MVC**

The attendance management system uses Spring web MVC framework. It has Model View Controller, MVC architecture and ready components which are used to produce loosely coupled and flexible dynamic web applications. The architecture results in modulating the different aspects of the application into UI logic, business logic, and input logic, ensuring a loose coupling between these elements.

* The **Model**
  + encapsulates the application data that will consist POJO.
* The **View**
  + responsible for rendering the model data. In general it generates the HTML output that the client's browser can interpret and visualize.
* The **Controller**
  + processes the user requests, build an appropriate model and passes it to the view for rendering.

**4.6 Maven**

Maven is a comprehension and software project management tool that allows a developer to develop based on project object model POM. Maven’s key feature is dependency management. It helps to download the required JAR’s for the project build.

Maven simplifies and standardizes the project build process. It handles compilation, distribution, documentation, team collaboration and other tasks seamlessly. Maven increases reusability and takes care of most of build related tasks.

**4.6.1 Maven – POM**

POM stands for *Project Object Model*. It is fundamental Unit of Work in Maven. It is an XML file. It always resides in the base directory of the project as pom.xml.

The POM contains information about the project and various configuration detail used by Maven to build the project(s).

POM also contains the goals and plugins. While executing a task or goal, Maven looks for the POM in the current directory. It reads the POM, gets the needed configuration information, then executes the goal. Some of the configuration that can be specified in the POM are following:

* project dependencies
* plugins
* goals
* build profiles
* project version
* developers
* mailing list

Before creating a POM, we should first decide the project **group** (groupId), its **name**(artifactId) and its version as these attributes help in uniquely identifying the project in repository.

## Example POM

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0

http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.companyname.project-group</groupId>

<artifactId>project</artifactId>

<version>1.0</version>

</project>

It should be noted that there should be a single POM file for each project.

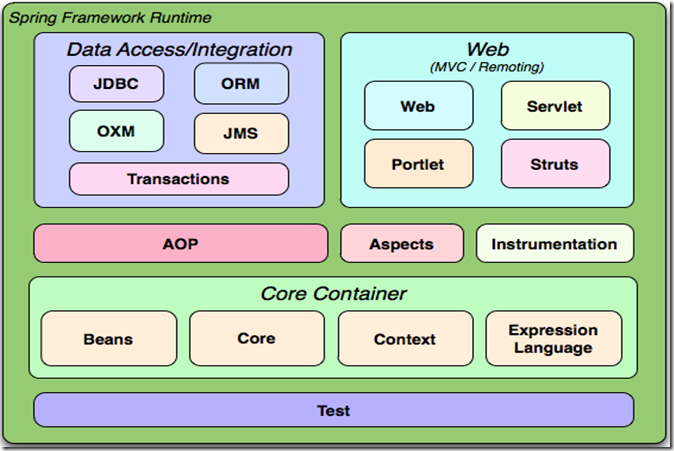
* All POM files require the **project** element and three mandatory fields: **groupId, artifactId,version.**
* Projects notation in repository is **groupId:artifactId:version.**
* Root element of POM.xml is **project** and it has three major sub-nodes :

|  |  |
| --- | --- |
| **Node** | **Description** |
| groupId | This is an Id of project's group. This is generally unique amongst an organization or a project. For example, a banking group com.company.bank has all bank related projects. |
| artifactId | This is an Id of the project.This is generally name of the project. For example, consumer-banking. Along with the groupId, the artifactId defines the artifact's location within the repository. |
| version | This is the version of the project.Along with the groupId, It is used within an artifact's repository to separate versions from each other. |

**CHAPTER 5**

**PRESENT WORK**

**5.1 Conceptual Model / Proposed Architecture**

**5.1.1. SPRING MVC ARCHITECTURE - BLOCK DIAGRAM**

The system developed uses spring web MVC framework. It provides a model-view-controller architecture and in built components that are used to develop flexible and loosely coupled web applications. The MVC pattern helps in separating the different parts of the application like UI logic, business logic, and input logic), by providing a loose coupling between these elements.

**Model –** Every class are developed according to the Spring modal. Every possible interaction, use cases has a set of bean classes, DAO classes, interfaces, Services, repositories.

**View –** All the view components like html, jsp are under different modules. Angular components are used under directives for secured and modular approach.

**Controller –** RESTful API is separated as controllers and used to pass data to and from the back end to the angular modules.

**5.1.2.BENEFITS OF SPRING FRAMEWORK**

* Spring provides a consistent transaction management interface
* Spring framework helps the developers to develop responsive enterprise class applications using POJOs. EJB container has such as an application server instead used only a robust servlet container such as Tomcat Server.
* Lightweight containers that incorporate Inversion of Control are useful for developing and deploying applications on systems with limited memory and CPU resources.
* Spring framework is organized in a modular way.
* It makes use of the existing frameworks like ORM, logging, JEE, Quartz and JDK timers.
* Spring applications ease the testing works. It is simple and flexible because environment dependent code is integrated into the framework. By using Java Bean style POJOs, it becomes easier to use dependency injection for injecting test data.
* Spring offers suitable and compatible API to translate technology specific exceptions thrown by JDBC, Hibernate, or JDO into consistent, unchecked exceptions. Hibernate ORM is used to associate bean classes with database directly to achieve two way data binding.
* Spring Session has an API to manage user's session information. It also provides transparent integration with:

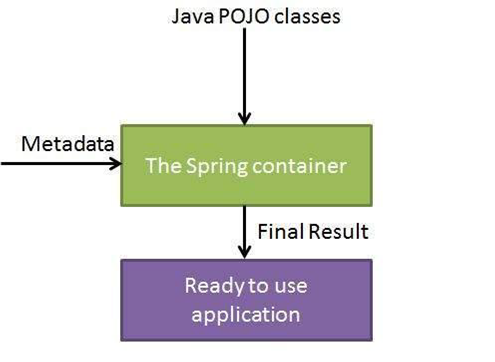
1. Http Session
2. Clustered Sessions
3. Multiple Browser Sessions
4. RESTful APIs
5. Web Socket

All the above mentioned sessions are used to make the system more secured. RESTful API and web sockets are used to implement two way data binding.

**5.1.3 SPRING IOC CONTAINERS**

     The spring has IOC container. The IOC container is the core element of Spring Framework. The container instantiates the objects, wire them together, configure them based on the requirements, and manage their complete lifecycle from creation to destruction.

             The IOC container uses the concept of dependency injection to manage the components. These are called Spring Beans. The container gathers information from ORM on what objects to create, assemble, and configure from the configuration metadata specified which can be represented as XML,  Java code, or Java annotations. The diagram below is the high level view of how spring works. The Spring container makes use of  POJO classes. It provides configuration metadata to produce a completely configured and executable application.

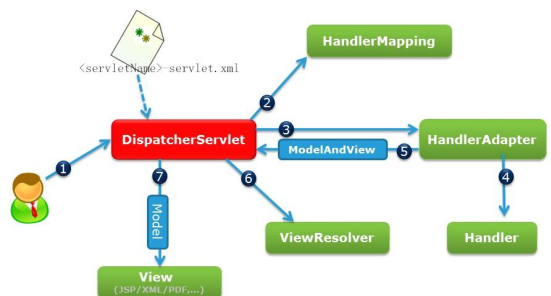


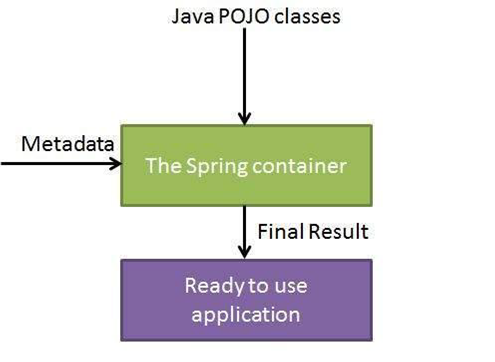
**5.1.4. ASPECT -ORIENTED PROGRAMMING**

               Aspect oriented programming, AOP is one of the major components in spring framework.  The functions that span multiple points in an application are called cross-cutting concerns. They are theoretically separate from the application's business logic. There are various aspect examples including security, declarative transactions, logging, and caching.

               The prime aspect of modularity in Object Oriented Programming is the class, while in Aspect Oriented Programming the unit of modularity is the aspect. The AOP module has aspect oriented programming implementation that enables you to define interceptors, method and point cuts to precisely decouple code that implements functionality that should be modularised.

* Aspect
* modularization of a concern that cuts across multiple classes. Transaction management in chit fund system.
* Join point
  + a point that represents execution and occurs during the execution of a program or handling of an exception.
* Advice
  + Any action taken at a particular joins point. Spring model an advice as an interceptor. Before, After and Around are aspects are used in the developed system.
* Pointcut
  + It is a predicate which is matched with join points. Advice is associated with a pointcut expression and runs at any join point matched by the point cut.





**5.1.5. DEPENDENCY INJECTION (DI)**

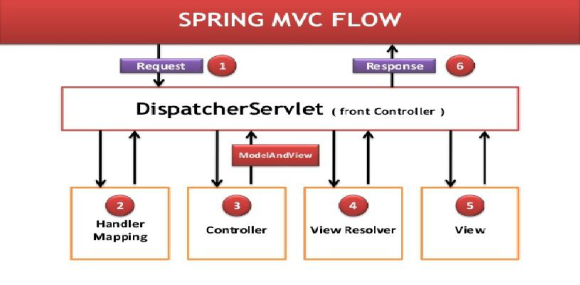
The spring framework has Dependency Injection as a flavour of Inversion of Control. Dependency Injection is the concrete example of Inversion of Control.

         When complex Java applications are being built, DAO classes should be independent of other Java classes to increase the modular reusability and to test it independently   Dependency Injection aids in associating these classes together and keeping them independent.

             Dependency translates into a relationship or an association between two classes. For example, class X is dependent on class Y. Now, the Injection is that class Y will get injected into class X by the Inversion of Control. It can be as of passing parameters to the constructor or by post-construction using setter methods.

**5.1.6. DISPATCHER SERVLET**

             The Spring MVC web framework is designed with a Dispatcher Servlet that handles all the HTTP responses and requests. Sequence of events for an incoming HTTP request to Dispatcher Servlet:

* Receiving an HTTP request, the Dispatcher Servlet goes to the handler mapping to call the appropriate Controller.
* The Controller processes the request and calls the appropriate service methods based on GET or POST methods. The service method will set model data based on business logic defined and  view name is returned to the Dispatcher Servlet.
* With the help of View Resolver, Dispatcher Servlet picks defined view for the request .Dispatcher Servlet passes model data to view.

**CHAPTER 6**

**RESULTS AND DISCUSSIONS**

**6.1 ACTOR 1 – STUDENT**

****

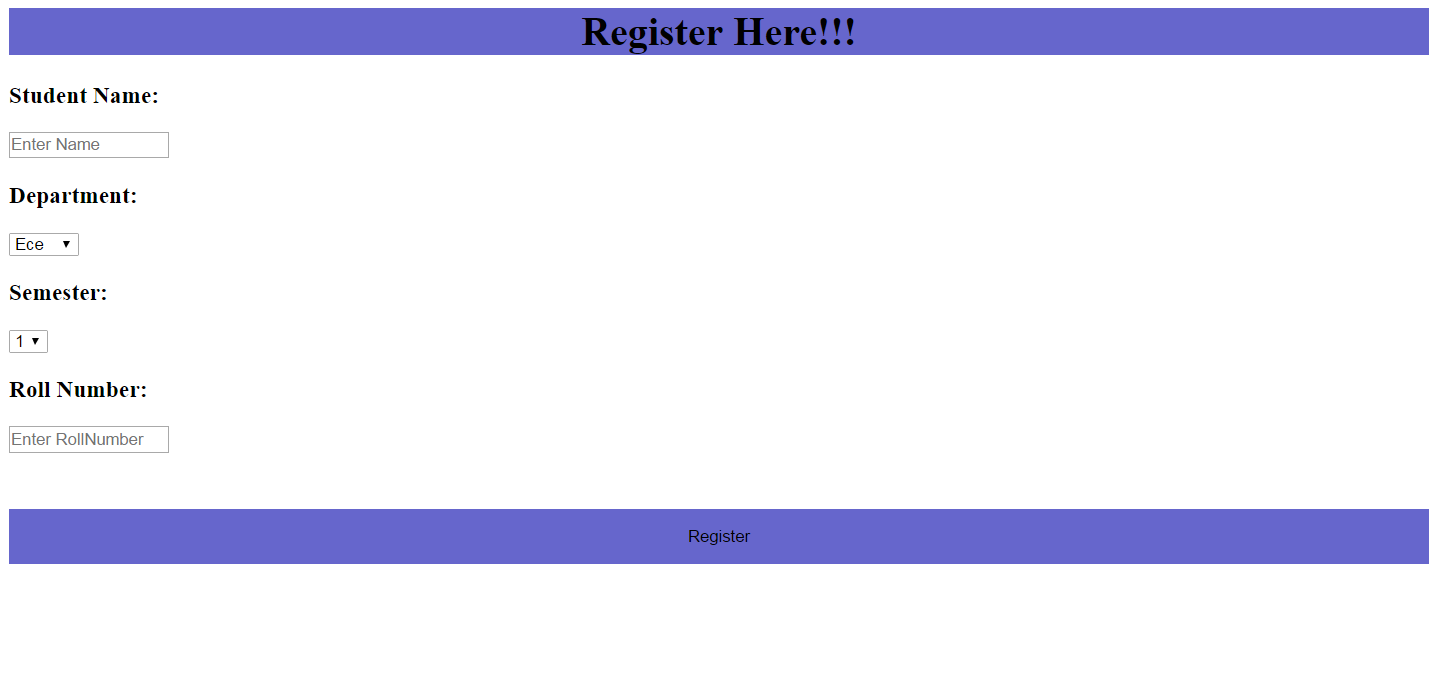
Fig6.1.1Loginpage 

Fig 6.1.2 Registration page

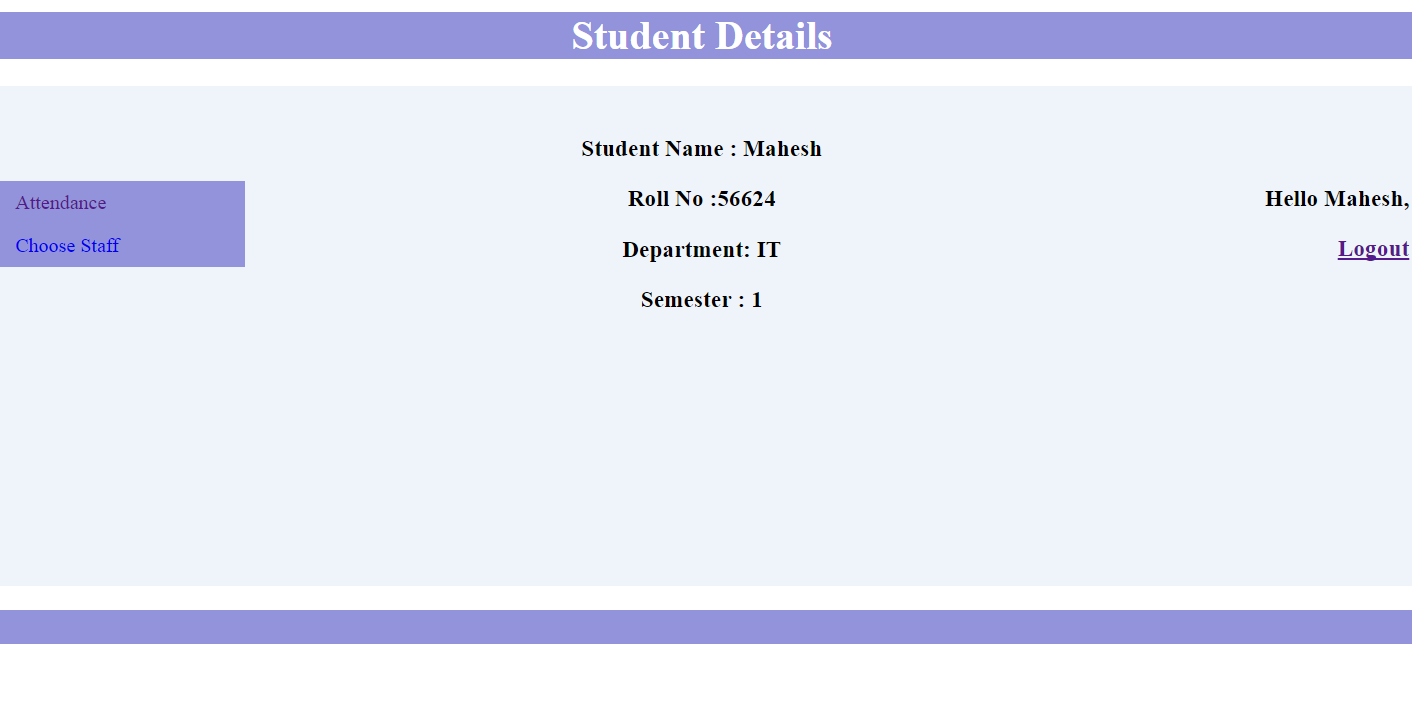


Fig 6.1.3 Student Details page



Fig 6.1.4 Subjectwise Attendance page

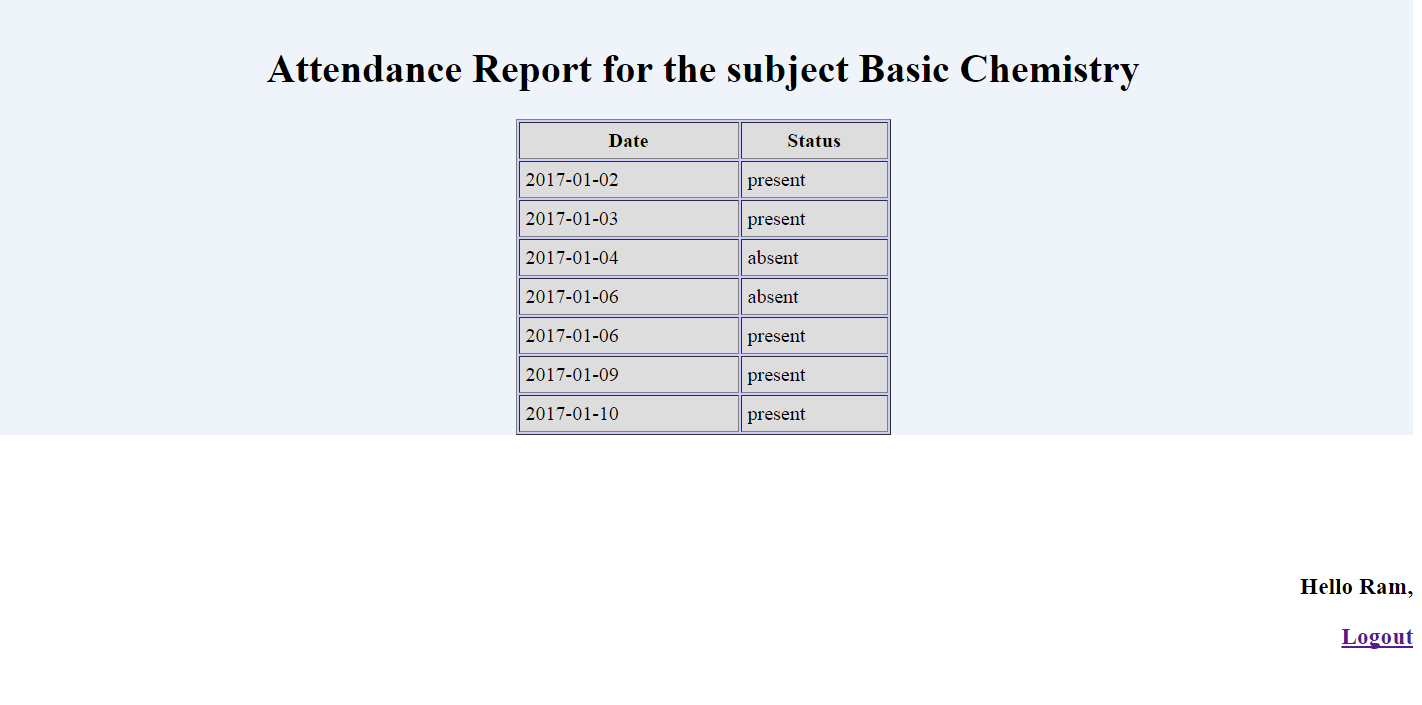


Fig 6.1.5 Subjectwise Attendance page

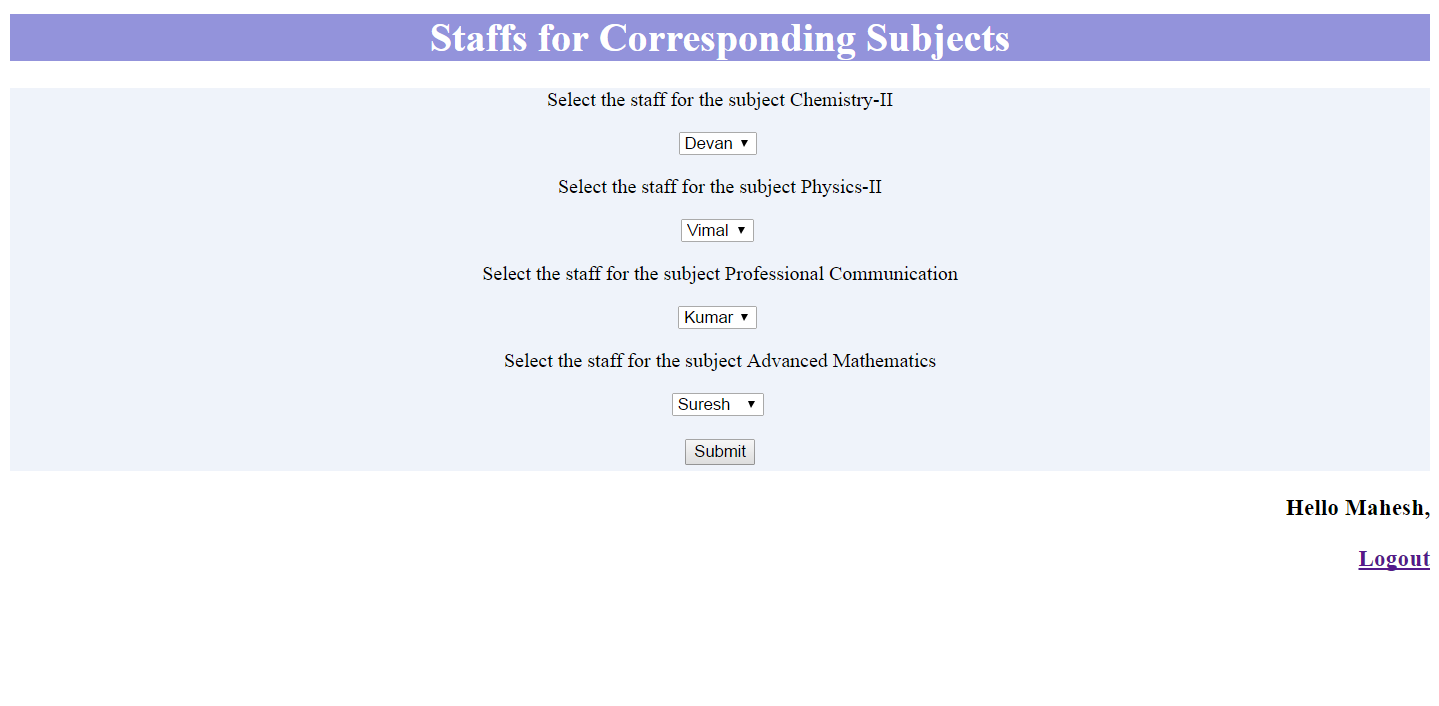
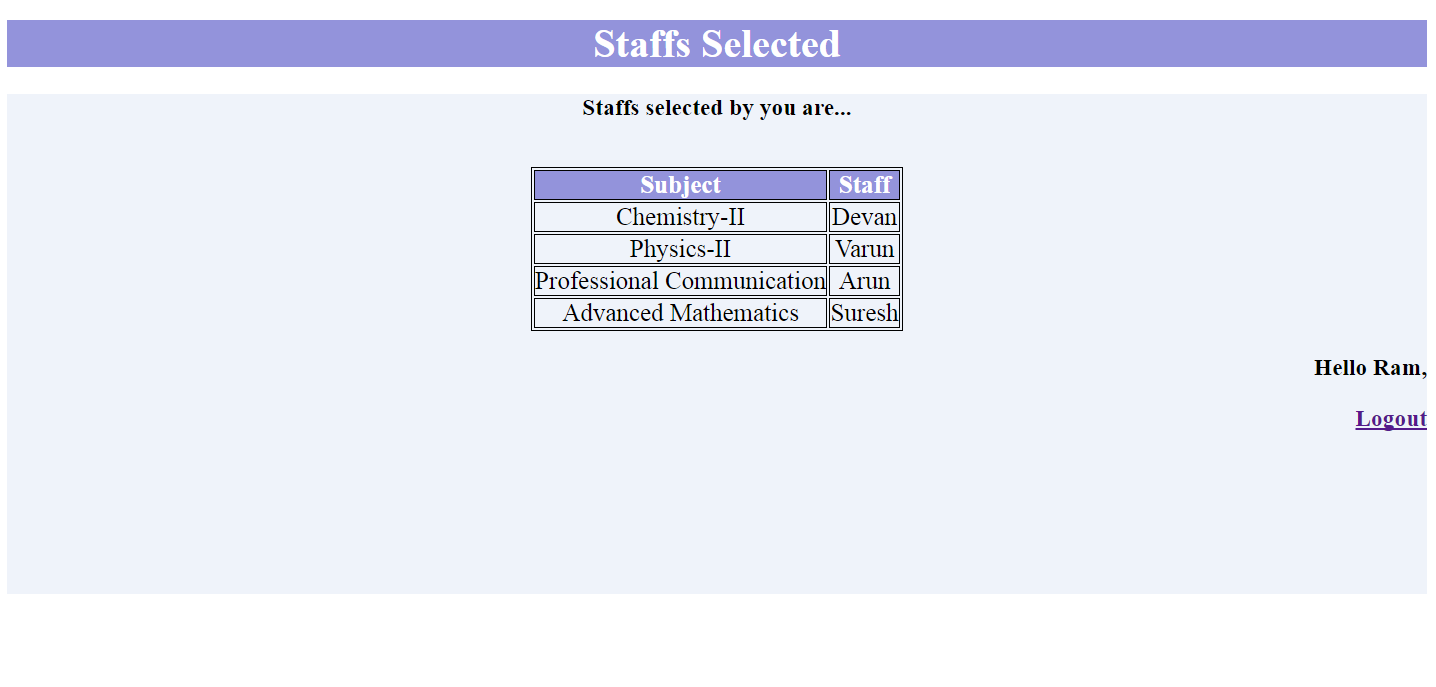
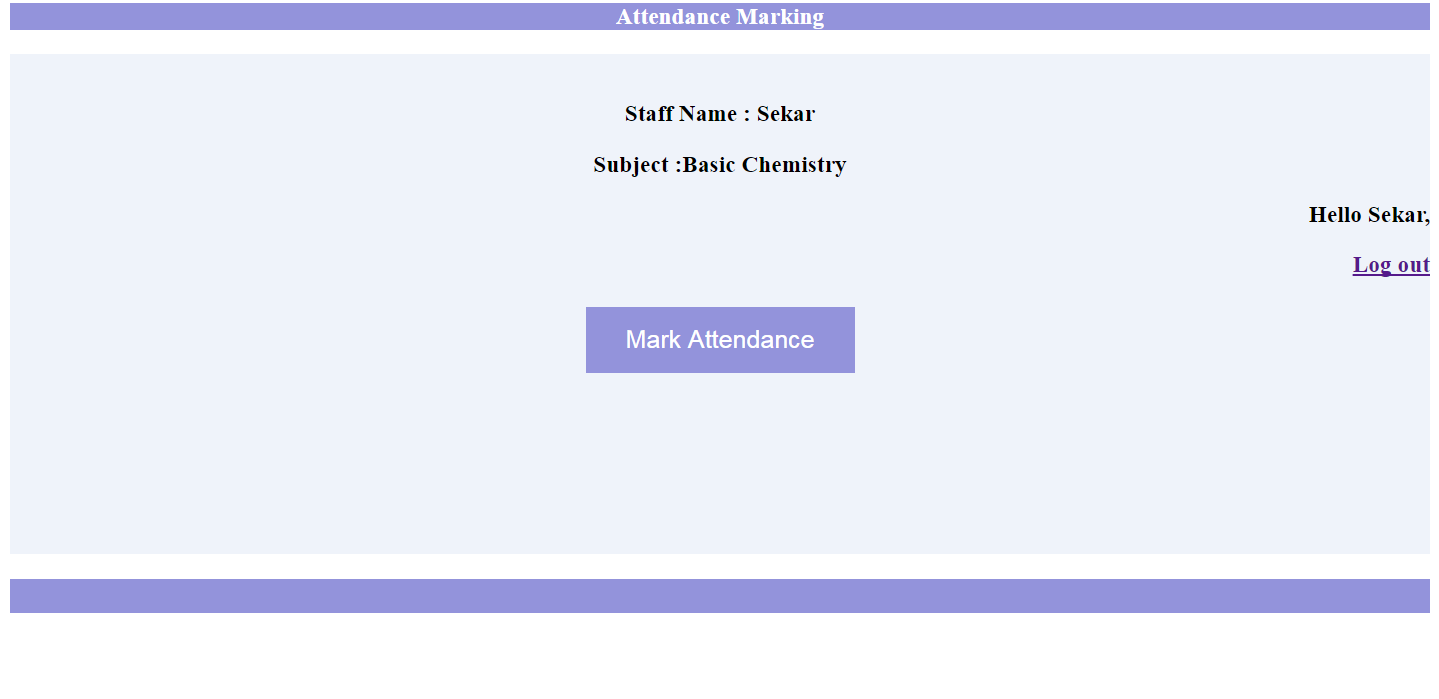
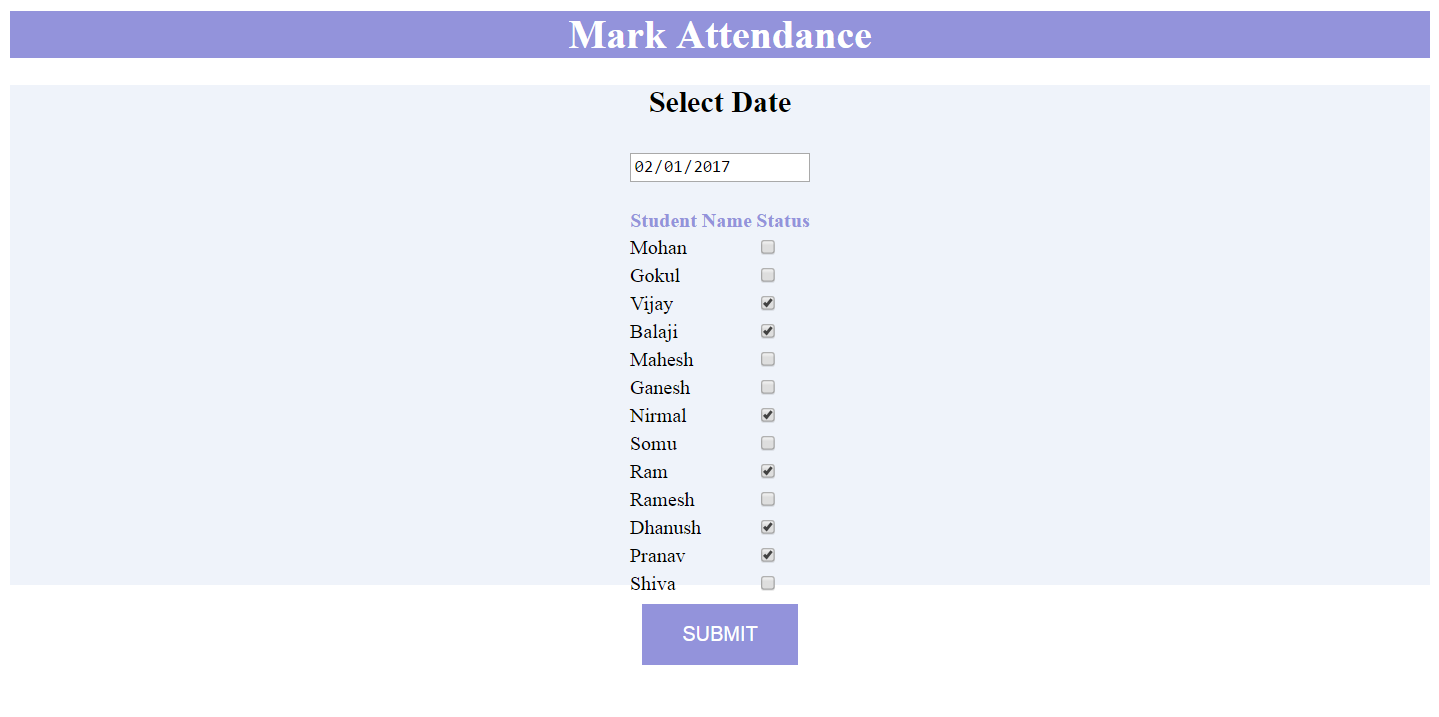


Fig 6.1.6 Staff Choosing page

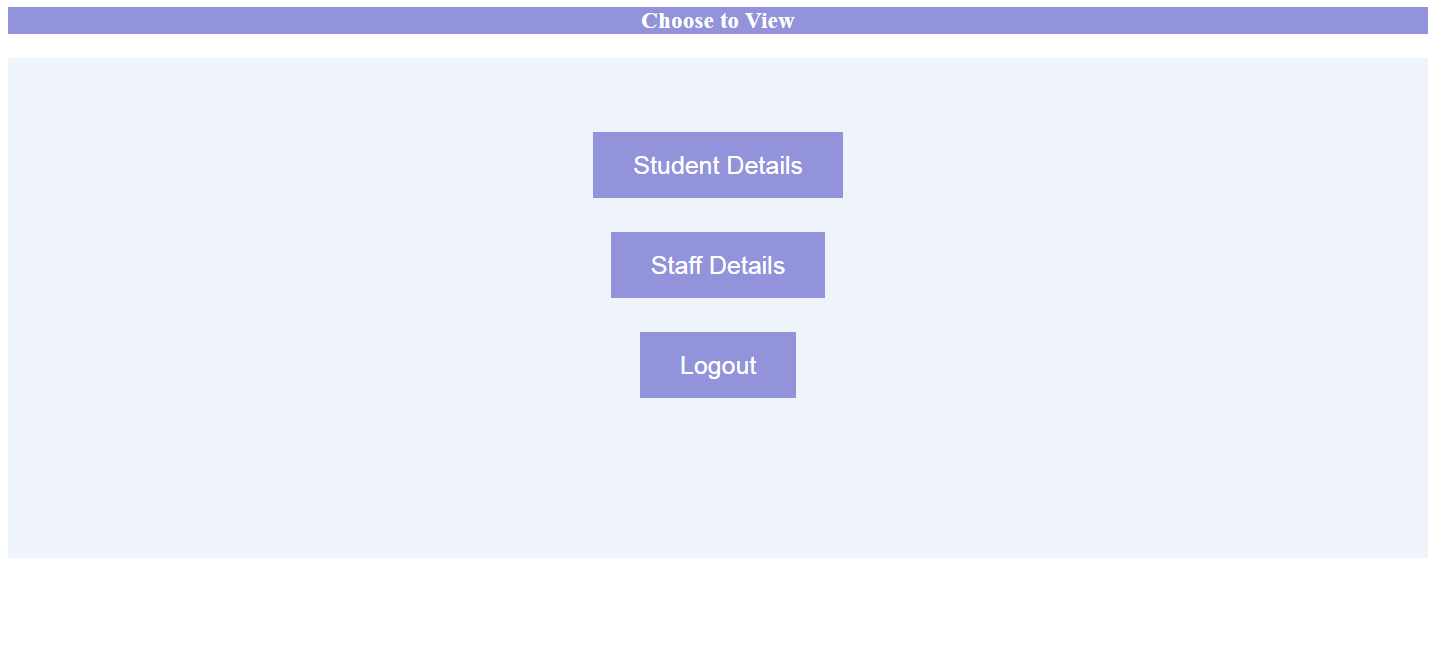


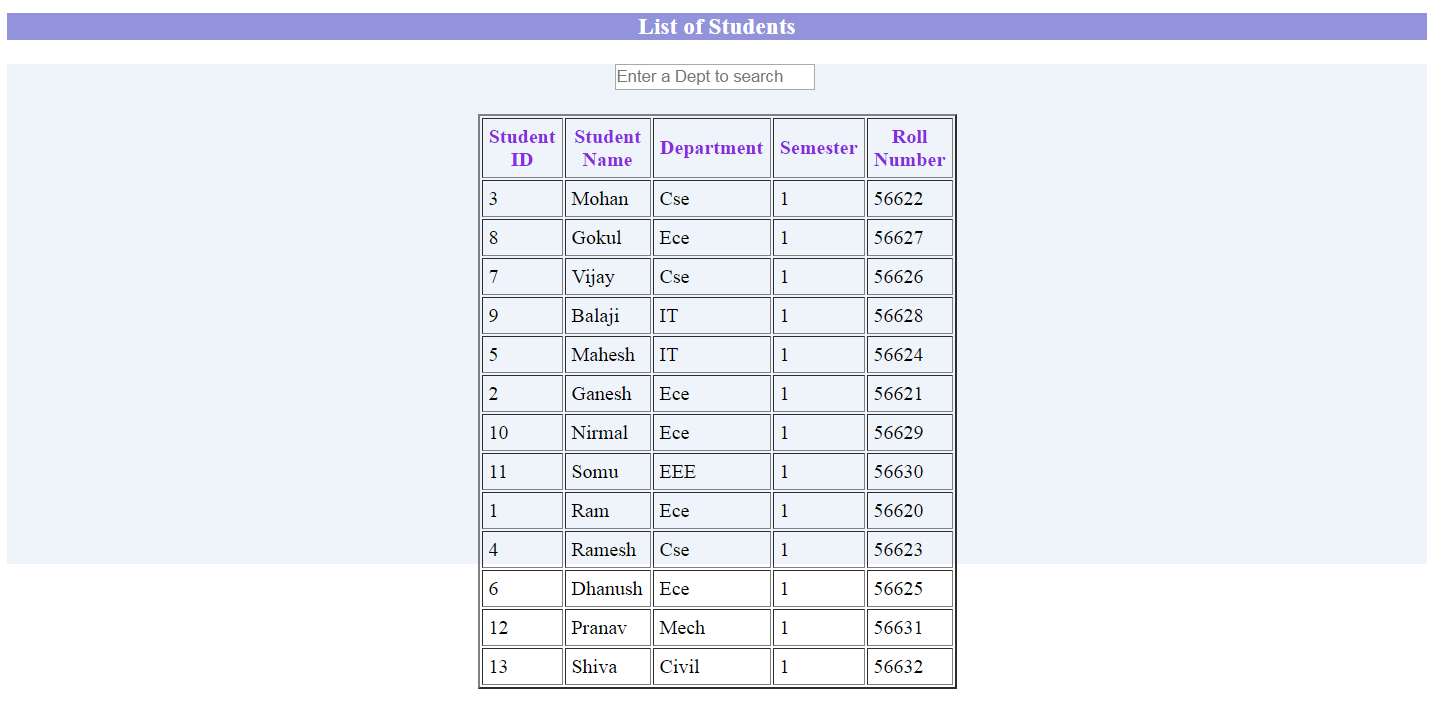
**6.2 ACTOR 2 – STAFF**

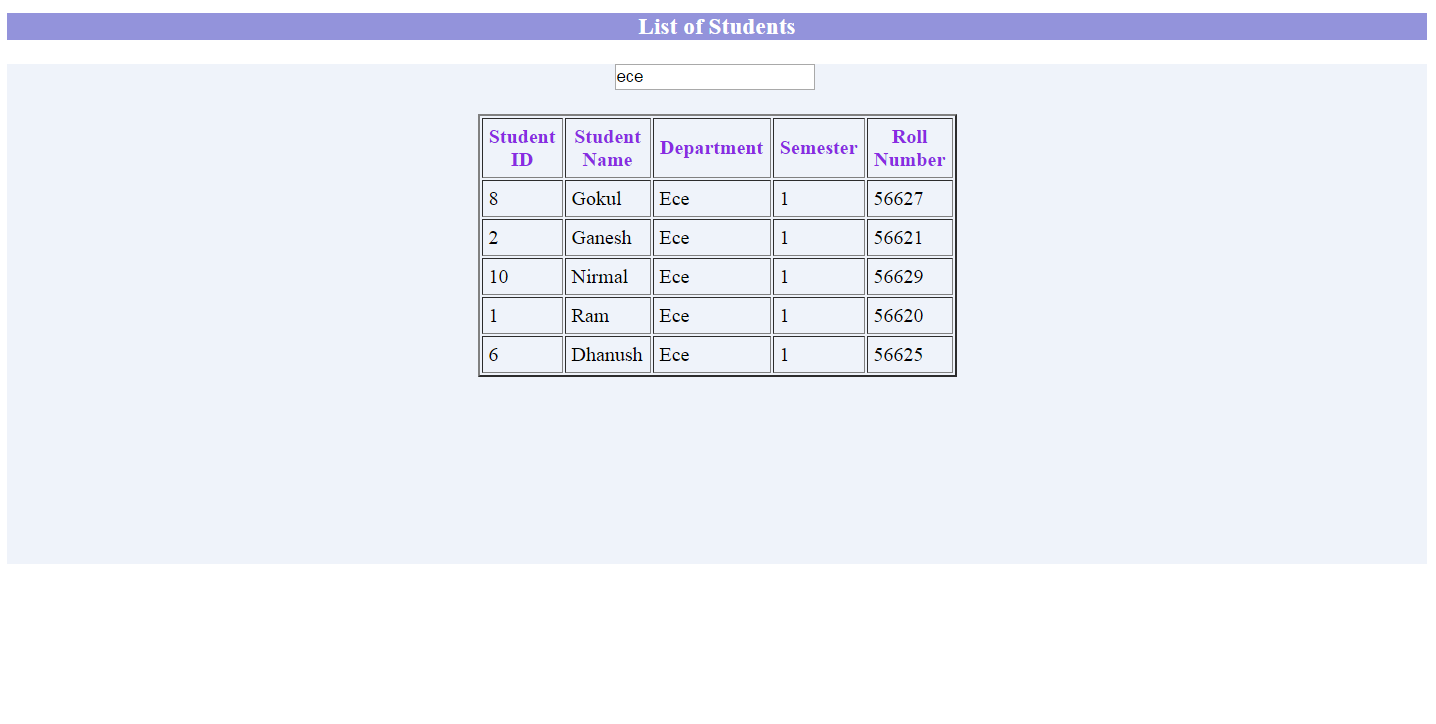
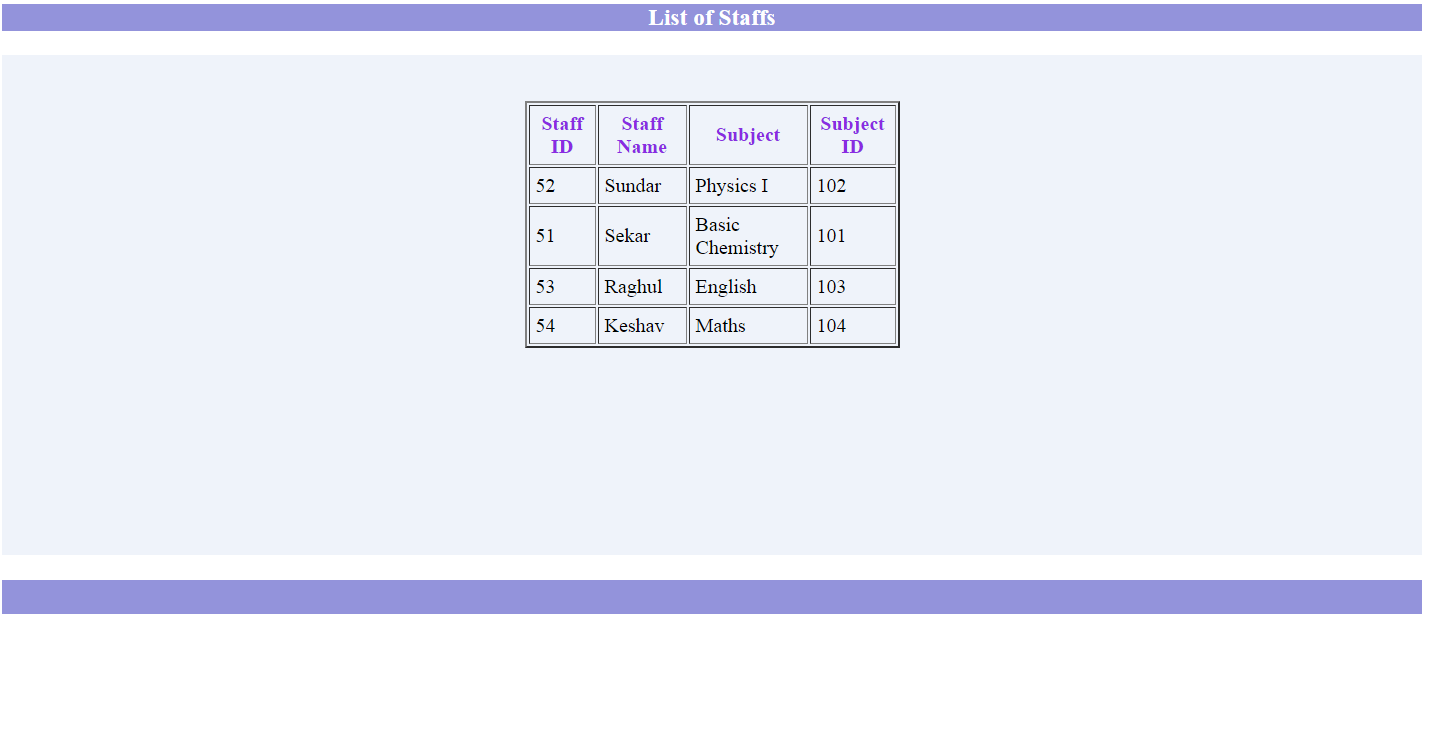
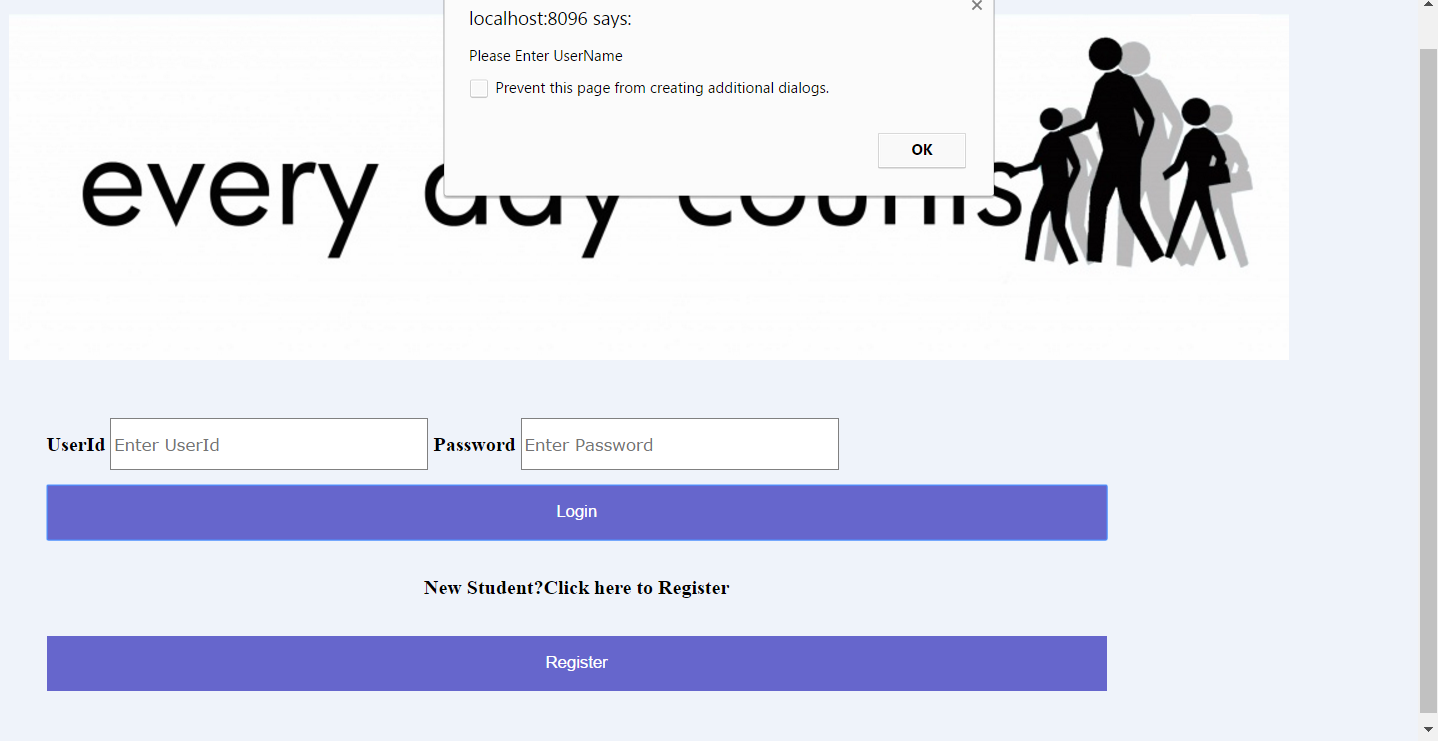
****

****

**6.3 ACTOR 3 – ADMIN :**

****

****

**** **** 

**CHAPTER 7**

**CONCLUSION**

To conclude, Project Data Grid works like a component which can access all the databases and picks up different functions. It overcomes the many limitations incorporated in the attendance.

* Easy implementation Environment
* Generate report Flexibly

The project has a very vast scope in future. The project can be implemented on intranet in future. Project can be updated in near future as and when requirement for the same arises, as it is very flexible in terms of expansion. With the proposed software of database Space Manager ready and fully functional the client is now able to manage and hence run the entire work in a much better, accurate and error free manner. The following are the future scope for the project.

* Discontinue of particular student eliminate potential attendance.
* Bar code Reader based attendance system.
* Individual Attendance system With photo using Student login.