**Employee Management System**

**JAVA Advance Project**

**BACHELOR OF TECHNOLOGY**

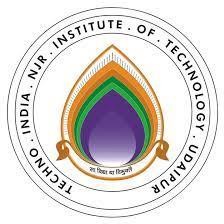
**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

SUBMITTED BY:-

|  |  |  |
| --- | --- | --- |
| **BHAVINI JAIN** | **CSE** | **19ETCCS007** |
| **HARSHIT CHAUBISA** | **CSE** | **19ETCCS024** |
| **JAI KANTHALIA** | **CSE** | **19ETCCS029** |

**UNDER THE GUIDANCE OF**

**Mr. Gaurav Kumawat**

****

**Techno India NJR Institute of Technology 2021-2022**

*Techno India NJR Institute of Technology*

**(APPROVED BY A.I.C.T.E , NEW DELHI & AFFILIATED BY RTU, Kota)**

***RIICO INDUSTRIAL AREA KALADWAS, UDAIPUR***





*CERTIFICATE*

**SESSION 2021-22**

*This is to certify that* ***Bhavini Jain , Jai Kanthalia , Harshit Chaubisa*** *, II year Students of department of Computer Science and Engineering of Techno India NJR Institute of Technology have successfully completed project on “Employee Management System”.*

***Mr***. ***Gaurav Kumawat***

**(H.O.D.,Computer Sc.)**

**Table of Contents**

1. Acknowledgement------------------------------4
2. Preface--------------------------------------------5
3. Introduction-------------------------------------- 6
4. Introduction to Java-----------------------------7
5. Introduction to Project ( EMS )--------------- 12
6. EMS Modules ---------------------------------- 13
7. System Specifications--------------------------17
8. SRS------------------------------------------------19
9. System Analysis and Design-------------------20
10. Data Analysis , Modeling and Design--------23
11. Data Flow Diagram-----------------------------26
12. Database Design--------------------------------30
13. Entity Relationship Diagram------------------31
14. Technical Requirements-----------------------32
15. Snapshots----------------------------------------33
16. Suggestions--------------------------------------40
17. Conclusion---------------------------------------41
18. Bibliography ------------------------------------42

# Acknowledgement

It is a fortune to find opportunity to express our deep sense of gratitude to all those people who helped us with their guidance and assistance, without which the project would not be possible. Their contribution has been invaluable. The enduring pages of the work are the cumulative sequence of extensive guidance and arduous work. We wish to acknowledge and express our personal gratitude to all those without whom this work could not have been reality. We feel very delighted to get this rare opportunity to show our profound senses of reverence and indebtedness to all our esteemed teachers and ***Mr***. ***Gaurav Kumawat (Head Of Computer Science Department)*** for their keen and sustained interest, valuable advice, throughout the project. For this kind of consideration we beholden to them in a special manner, and no words can fully convey our feelings of respect and regards for them.

At last we also want to express our sincere thanks to all the classmates for their coordination and friendly cooperation with their suggestions.

Last but not least we would like to thanks all those who have directly or indirectly help and cooperated in accomplishing this.

## **Preface**

Project is a special significance for engineering students. They learn to tackle different problems that strike while developing, any computer application.

The application titled ‘Employee Management System’ is developed with the intent to keep the track of every data entry in Office and so there is no need to manage and record data manually.

The project has been developed using Apache NetBeans IDE 12.4 , MySQL and Xampp **.** This is user friendly, easy to use, understandable and satisfies all user requirements.

There is no doubt that in spite of strenuous efforts, errors might remain in the text. We truly apologize for that.

## **Introduction**

**PROJECTTITLE** : Employee Management System

**PLATFORM** : Windows 7 , 8 , 10

**FRONTENDTOOLS** : Apache NetBeans IDE 12.4

**BACKENDTOOLS** : MySQL

## **Introduction to JAVA**

JAVA was developed by James Gosling at Sun Microsystems Inc in the year 1991, later acquired by Oracle Corporation. It is a simple programming language. Java makes writing, compiling, and debugging programming easy. It helps to create reusable code and modular programs.

[Java](https://www.geeksforgeeks.org/java/) is a class-based, object-oriented programming language and is designed to have as few implementation dependencies as possible. A general-purpose programming language made for developers to write once run anywhere that is compiled Java code can run on all platforms that support Java. Java applications are compiled to byte code that can run on any Java Virtual Machine. The syntax of Java is similar to c/c++.

**Innovations**

The benefits of object-oriented programming are not necessarily obvious to non-programmers. Essentially what this means to managers is that code written in **Java** will be easier to maintain and reuse in the long run. The downside is that object-oriented programming requires better planning from the beginning of a project, and may increase the amount of initial development time a project requires. For this reason alone, **Java** is rarely a good choice for small projects requiring a fast turnaround time.

The second innovation that **Java** provides, platform neutrality, is perhaps the

greatest reason for its wide acceptance. The fact that **Java** was originally intended a language for writing device controllers for items such as garage openers and microwave ovens is the key reason for this. In practice, however, this ability has been more useful in writing enterprise class business applications, where mission critical software may be required to run on a variety of platforms over its lifetime. Theoretically at least, once compiled, a **Java** binary should be able to run on any machine that also has a piece of software called a **Java** Virtual Machine. In reality, this is not always the case. However, more often than not **Java** does succeed in this regards, whereas this is impossible with an application written in a language such as C++.

**Java in Web Development**

In terms of web development, **Java** is frequently used in two ways. Most commonly **Java** is used to write server-side web applications using two technologies: JSPs and servlets.

Using **Java** in this capacity is a good choice for complex applications, that will have large numbers (~1000+) of concurrent users, and will be developed by a team of programmers. Less complex projects, with fewer concurrent users may have better outcomes when developed in procedural scriptinglanguagessuch as PHP or PERL.  
  
The second way in which **Java** is used, is to create special, browser embeddable, programs called Applets. While applets had a brief period of acceptance, their use is becoming increasingly rare, being replaced by a number of technologies such a Flash and JavaScript, which are more effective at providing design enhancements such as animation and rollovers. Using applets for these purposes is a mistake frequently made by beginning developers. Still, Applets do have a place in writing specialized browser-based applications that cannot be accomplished by these other technologies. Careful consideration should be made before approving any project that uses Applets.

**Classes and Objects**

**Classes**

In the real world, you'll often find many individual objects all of the same kind. There may be thousands of other bicycles in existence, all of the same make and model. Each bicycle was built from the same set of blueprints and therefore contains the same components. In object-oriented terms, we say that your bicycle is an instance of the class of objects known as bicycles. A class is the blueprint from which individual objects are created.

**Objects**

Objects are key to understanding object-oriented technology. Look around right now and you'll find many examples of real-world objects: your dog, your desk, your television set, your bicycle.

Real-world objects share two characteristics: They all have state and behavior. Dogs have state (name, color, breed, hungry) and behavior (barking, fetching, wagging tail). Bicycles also have state (current gear, current pedal cadence, current speed) and behavior (changing gear, changing pedal cadence, applying brakes). Identifying the state and behavior for real-world objects is a great way to begin thinking in terms of object-oriented programming.

Take a minute right now to observe the real-world objects that are in your immediate area. For each object that you see, ask yourself two questions: "What possible states can this object be in?" and "What possible behavior can this object perform?". Make sure to write down your observations. As you do, you'll notice that real-world objects vary in complexity; your desktop lamp may have only two possible states (on and off) and two possible behaviors (turn on, turn off), but your desktop radio might have additional states (on, off, current volume, current station) and behavior (turn on, turn off, increase volume, decrease volume, seek, scan, and tune). You may also notice that some objects, in turn, will also contain other objects. These real-world observations all translate into the world of object-oriented programming.

Software objects are conceptually similar to real-world objects: they too consist of state and related behavior. An object stores its state in fields (variables in some programming languages) and exposes its behavior through methods (functions in some programming languages). Methods operate on an object's internal state and serve as the primary mechanism for object-to-object communication. Hiding internal state and requiring all interaction to be performed through an object's methods is known as data encapsulation — a fundamental principle of object-oriented programming.

By attributing state (current speed, current pedal cadence, and current gear) and providing methods for changing that state, the object remains in control of how the outside world is allowed to use it. For example, if the bicycle only has 6 gears, a method to change gears could reject any value that is less than 1 or greater than 6.

Bundling code into individual software objects provides a number of benefits, including:

1. **Modularity:** The source code for an object can be written and maintained independently of the source code for other objects. Once created, an object can be easily passed around inside the system.
2. **Information-hiding:** By interacting only with an object's methods, the details of its internal implementation remain hidden from the outside world.
3. **Code re-use:** If an object already exists (perhaps written by another software developer), you can use that object in your program. This allows specialists to implement/test/debug complex, task-specific objects, which you can then trust to run in your own code.

**Pluggability and debugging ease:**If a particular object turns out to be problematic, you can simply remove it from your application and plug in a different object as its replacement. This is analogous to fixing mechanical problems in the real world. If a bolt breaks, you replace it, not the entire machine.

**Package**

A package is a namespace that organizes a set of related classes and interfaces. Conceptually you can think of packages as being similar to different folders on your computer. You might keep HTML pages in one folder, images in another, and scripts or applications in yet another. Because software written in the Java programming language can be composed of hundreds or thousands of individual classes, it makes sense to keep things organized by placing related classes and interfaces into packages.

The Java platform provides an enormous class employee (a set of packages) suitable for use in your own applications. This ERS is known as the "Application Programming Interface", or "API" for short. Its packages represent the tasks most commonly associated with general-purpose programming. For example, a String object contains state and behavior for character strings; a File object allows a programmer to easily create, delete, inspect, compare, or modify a file on the filesystem; a Socket object allows for the creation and use of network sockets; various GUI objects control buttons and checkboxes and anything else related to graphical user interfaces. There are literally thousands of classes to choose from. This allows you, the programmer, to focus on the design of your particular application, rather than the infrastructure required to make it work.

The Java Platform API Specification contains the complete listing for all packages, interfaces, classes, fields, and methods supplied by the Java Platform 6, Standard Edition. Load the page in your browser and bookmark it. As a programmer, it will become your single most important piece of reference documentation.

**JAVA SWING INTRODUCTION:**

Java Swing or Swing was developed based on earlier APIs called Abstract Windows Toolkit (AWT).Swing provides richer and more sophisticated GUI components than AWT. The GUI components are ranging from a simple label to complex tree and table

Swing is platform independent and enhanced MVC (Model -View -Controller) framework for Java application. Here are the most important features in Swing architecture.

**Pluggable look andfeel**:

Swing supports several looks and feels and currently supports Windows, UNIX, Motif, and native Java metal look andfeel.

Swing allows users to switch look and feel at runtime without restarting theapplication.

**Lightweightcomponents**:

All swing components are lightweight except some top-level containers.

Lightweight means component renders or paints itself using drawing primitives of the *Graphics* object instead of relying on the host operating system(OS).

**Simplified MVC**:

Swing uses simplified model-view-architecture (MVC) as the core design behind each of its components calledmodel-delegate.

Based on this architecture, each swing component contains a model and a UIdelegate.

**JFrame and JPanel**

JFrameis the most commonly used top-level container. It adds basic functionality such as minimize, maximize, close, title and border to basic frames and windows. Some important JFrame methods are: setBounds(x,y,w,h), setLocation(x,y), setSize(w,h), setResizable(bool), setTitle(str), setVisible(bool), isResizable() and getTitle(). The setDefaultCloseOperation(constant) method controls the action that occurs when the close icon is clicked. Normally the constant used isJFrame.EXIT\_ON\_CLOSE**.**

JPanelis the most commonly used content pane. An instance of the pane iscreated and then added to a frame. Theadd() method allows widgets (GUI components) to be added to the pane. The way they are added is controlled by the current layout manager.

**Labels, Icons and Buttons**

Labels are non-interactive text objects most commonly used as prompts. They are created using the JLabel() constructor with the required text as the first parameter. Another parameter can be added using a Swing Constant value to set horizontal alignment. Vertical alignment is through the setVerticalAlignment() method. The contents of a label can be changed with the setText() method.

Icons can be easily added to labels or other controls either to brand, dress up, or aid accessibility. Icons are constructed from the ImageIcon class and then added as a parameter to the label (or other) control. An extra parameter can be used to control the position of the text relative to the icon. It must use one of the SwingConstants values.

Simple buttons are used to start operations. They are created with the JButton() constructor. They can be deactivated with the setEnabled(false) method and tested with the isEnabled() method. One useful button method is setMnemonic(char) which allows a hot key to be associated with the button

Simple buttons require an ActionEvent event listener that reacts to the button click.

Toggle buttons are a visual push on - push off mechanism. They are created with the JToggleButton() constructor. The isSelected() method returns the state of the button. And in addition to ActionEvent, the ChangeEvent is triggered.

**Event Listeners**

GUIs are event-based. That is they respond to buttons, keyboard input or mouse activities. Java uses event listeners to monitor activity on specified objects and react to specific conditions. For a listing of useful event listeners check the appendix. For techniques on organizing many different events in larger projects, view advanced event listeners.

The first step in adding a basic button push event handler to the above example is to import awt.event.\* which contains all of the event classes. Next add the phrase implements ActionListener to the class header. Register event listeners for each button widget using the addActionListener(this) method.

## **Introduction to Project**

This Employee Management System Project application stores all the employee’s information in a database. It is an application developed in Java GUI technology and database used is MySQL. It contains employee information like employee id, first name, surname, and age, etc. It is an easy to use application and has a user-friendly interface. It is totally built at the administrative end which means that only the admin has access rights to change or modify any records. So this makes it safe and reliable application to use. The main aim of developing this application was to reduce the errors that occur in the manual system. One can search the details easily by just entering employee id. In earlier systems, there was not such a facility to do so. All the details are stored in an MySQL database. It is easy to update any employee details. All the employee records are integrated and so this makes it user-friendly and easy to use application. Employee Management System Project in Java is java project developed using java GUI or java swings.

Employee Management System is a distributed application, developed to maintain the details of employees working in any organization. It maintains the information about the personal details of their employees, also the details about the payroll system which enable to generate the payslip. The application is actually a suite of applications developed using Java.

It is simple to understand and can be used by anyone who is not even familiar with simple employees system. It is user friendly and just asks the user to follow step by step operations by giving him few options. It is fast and can perform many operations of a company.

This software package has been developed using the powerful coding tools of JAVA at Front End and Microsoft Sql Server at Back End.The software is very user friendly. The package contains different modules like Employee details. This version of the software has multi-user approach. For further enhancement or development of the package, user?s feedback will be considered.

Employee Management system is an application that enables users to create and store Employee Records. The application also provides facilities of a payroll system which enables user to generate Pay slips too. This application is helpful to department of the organization which maintains data of employees related to an organization .

Java is a platform independent language. Its created applications can be used on a standalone machine as well as on distributed network. More over applications developed in java can be extended to Internet based applications.

Thus java was chosen as background to design this application.

**OBJECTIVE OF THE PROJECT**

------------------------

In this world of growing technologies everything has been computerized. With large number of work opportunities the Human workforce has increased. Thus there is a need of a system which can handle the data of such a large number of Employees in an organization. This project simplifies the task of maintain records because of its user friendly nature.

## **Employee ManagementSystem Modules**

## • **Login module**: This module is created for login details of admin.

## • **Manage employee module**: This module enables admin to add and edit employee records. It is also for viewing the each and every employee details by the admin.

## • **Delete module**: This module is created to delete any employee details.

## • **Search module**: This module is an added feature in this system which enables to search employee records easily using employee id.

**Existing system:**

Earlier systems were manual where there was no way of properly storing information. Employee records were stored manually which lead to errors. There was no proper way of tracking employee records. It was very difficult and required a lot of paperwork which makes the application time-consuming and was not secured. There was no administrator which could handle the records. So there was the need to develop a system which could manage all these things and reduce the paperwork.

**Proposed system:**

Employee information system is easy to use application which is created to manage the employee data. It is created to record the details of the employees. This reduces the dependency on the manual system which could create errors. This system can easily help in tracking employee records. There is a search feature which allows getting records of a specific employee. There is an administrator which can add, edit, delete and save records in a database.

## **SYSTEMSPECIFICATIONS**

Software Engineers have been trying various tools, methods and procedures to control the process of software development in order to build high quality software with high productivity. This method provides “how it is” for building the software while the tools provide automated or semi automated support for the methods. They are used in all stages of software development process, namely, planning, analysis, design, development and maintenance. The software development procedure integrates the methods and tools together and enables rational and timely development of the software system.

**Tools**

**Methods**

**Procedures**

**Software Development**

They provide the guidelines as how to apply these methods and tools, how to produce the deliverable at each stage, what controls to apply, and what milestones to use to assess the performance of the program. There exist a number of software development paradigms each using a different set of methods and tools. The selection of a particular paradigm depends on the nature of application of the programming language used for the controls and the delivariables required. The development of such successful systems depends not only on the use of appropriate methods and techniques but also the developers’ commitment to the objective of the system.

A successful system must: -

1. Satisfy the user requirements
2. Be easy to understand by user and operator
3. Be easy to operate
4. Have a good user interface
5. Be easy to modify
6. Be expandable
7. Have adequate security control against the misuse of data
8. Handle the errors and exceptions satisfactorily
9. Be delivered on schedule within the budget

## **SOFTWARE REQUIREMENT SPECIFICATION (SRS)**

The aim of the system is to develope **“EMPLOYEE MANAGEMENTSYSTEM”** software, which should automate the process to create and store employee details . The system is supposed to be used as a subsystem in a large office system, which could be manual system or a computerized one. Therefore, the proposed system must be able to function under both circumstances.

**The proposed system** is not a freeware and due to the usage of swings, becomes user interactive.

* The project demand a page of employee details that include:
* Employees personel detail.
* Employees salary, allowances, deductions.

## **System Analysis and Design of proposed system**

### Introduction

From the inception of an idea for a software system, until it is implemented and delivered to a customer, and even after that, the system undergoes several changes. The software is said to have a life cycle composed of several phase. Each of these phase results in the development of either a part of the system or something associated with the system, such as a test plan or a user manual. In the traditional life cycle model, called the waterfall model, each phase has well-defined starting and ending points, with clearly identifiable inputs to the next phase. In practice, it is really so simple.

A sample waterfall life cycle model comprises the following phases:

### Requirements analysis andspecification:

Requirements analysis is usually the first phase of large-scale software development project. It is undertaken after a **feasibility study** has been performed to the defined precise costs and benefits of a software system. The purpose of this phase is to identify and document the exact requirements for the system.

Such study may be performed by the customer, the developer, a marketing organization, or and combination of the three. In case where the requirements are not clear, e.g. for a system that has never been done before - much interaction is required between the user and the developer. The requirements at this stage are in end- user terms. Various software engineering methodologies advocate that this phase must also produce user manuals and system test plan.

### Design andspecification:

Once the requirements for a system have been documented, software engineers design a software system to meet them. This phase is sometimes split into two sub phases: architectural of high-level design and detailed design. High-level design with the overall module structure and organization, rather than details of the modules. The high-level design is refined by designing each module in detail.

Separating the requirements analysis phase from the design phase is instance of a fundamental what/how dichotomy that we encounter quite often in computer science. The general principle involves making a clear distinction between what the problem is and how to solve the problem. In this case, the requirements .

phase attempts to specify what the problem is. There are usually many ways that the requirements may be met, including some solutions that do not involve the user of computers at all.

The purpose of the design phase is to specify a particular software system that will meet the stated requirements. Again, there are usually many ways to build the specified system. In the coding phase, which follows the design phase, a particular system is coded to meet the designspecification.

### Coding and moduletesting:

This is the phase that produces the actual code that will be delivered to the customer as the running system. The other phases of the life cycle may also develop code, such as prototypes, tests and test drivers, but these are for use by the developer. Individual modules developed in this phase are also tested before being delivered to the next phase.

### Integration and systemtesting:

### All the modules that have been developed before and tested individually are put together - integrated - in this phase and tested as a wholesystem.

### Delivery andmaintenance:

Once the system passes all the tests, it is delivered to the customer and enters the maintenance phase. Any modifications and to the system after initial delivery are usually attributed to this phase.

A commonly used terminology distinguishes between high phase and down phase of the software life cycle: the feasibility study, requirements analysis, and high-level design contribute to the former, and implementation-oriented activities contributed to thelatter.

As presented here, the phases give a partial, simplified view of the conventional waterfall software life cycle. The process may be decomposed into a different set of phases, with different names, different purposes, and different granularity. Entirely different life cycle schemes may even be proposed, not based on a strictly phased waterfall development. For example, it is clear that if any tests uncover defects in the system, we have to go back at lease to the coding phase and perhaps to the design phase to correct some mistakes. In general, any phase may uncover problems in previous phases; this wills inconsistencies going back to the previous phases and redoing some earlier work. For example, if the system design phase uncovers inconsistencies or ambiguities in the system requirements, the requirements analysis phase must be revisited to determine what requirements we reallyintended.

Another simplification in the above presentation is that it assumes that a phase is completed before the next one begins. In practice, it is offer expedient to start a phase before a previous one is finished. This may happen, for example, if some data necessary for the completion of the requirements phase will not be available for some time. Or it might be necessary because the people ready to start the next phase are available and have nothing else to do. System Analysis is the first technical step in software development process. It is at this point that a general statement of software scope is refined into a concrete specification that becomes the foundation for all software development activities that follow. The objective of the problem is understood within the framework of the organization’s MIS objective.

## **Data Analysis, Modeling and Design**

#### Introduction

A key requirement of the Employee Management System is the need to identify the significant data that the organization needs to hold in order to operate effectively. Data analysis and modeling are techniques for identifying this important business data and the structures between them.

Most people involved in application development follow some kind of methodology. A methodology is a pre describe set of processes through which the developer analyses the client’s requirements and develops an application. Major database vendors and computer gurus all practice and promote their own methodology. Some database vendors even make their analysis, design and development tools conform to particular methodology (McLellan, Tim 1995a).

One technique commonly used in analyzing the client’s requirements is *data modeling.* The purpose of data modeling is to develop an accurate model, or graphical representation, of the client’s information needs and business processes. The data model acts as a framework for the development of the new or enhanced application. It is the most important task in the development of effective database application. If the data model incorrectly represents the user’s view of the data, the user will find the application difficult to use, incomplete and very frustrating. Data modeling is the basis for all subsequent work in the development of databases and their applications (Kroenk, David M. 1998). It involves interviewing users,analyzing data, documenting requirements and from those requirements building the data model and prototypes.

#### Data Models

A data model is used to represent the conceptual organization of the data. The conceptual model, also called an enterprise model, is a communication tool between the various users of data and is used to organize, visualize, plan and communicate ideas. It is developed without any concern of physical representations; it should be independent of a database management system. The data model represents the data that is significant to an organization, and the relationship between the data. Key elements of data model are entity, attributes, identifiers and relationship.

#### Entities

#### An entity is a thing or object of significance of the business, whether real or imagined, about which the business must collect and maintain data, or about which information needs to be known or held. An entity may be a tangible or real object like a person or building; it may be an activity like an appointment or an operation; it may be conceptual as in a cost center or an organizational unit. Whatever is chosen, as an entity must be described in real terms. It must be uniquely identifiable from all other instances of that type of entity. Examples of entity are employee, customer, products etc.

#### Attributes

#### After you identify an entity, then you describe in real terms, or through its attributes. An attribute is any detail that serves to identify quality, classify, quantify, or otherwise express the state of an entity occurrence or a relationship. Attributes are specific pieces of information, which need to be known or held. Attributes are either required or optional. When it’s required, we must have a value for it, a value must be known for each entity occurrence. When it’s optional, we could have a value for it, a value may be known for each entity occurrence. Examples of attributes are name of book, type of book , date of issue and depositetc.

#### Identifiers

Each instance of entity has an identifier that is an attribute, or attributes, which name or identify the entity. For example book number could identify book instances. The attributes, which uniquely define an occurrence of an entity, are called primary keys. If such an attribute doesn’t exist naturally, a new attribute is defined for that purpose. For example an ID number or code. Identifiers that consist of two or more attributes are called composite attributes.

#### Relationships

After two or more entities are identified and defined with attributes, the particulars determine if a relationship exits between the entities.A relationship is any association, linkage, or connection between the entities of interest of business it is a two directional significant association between two entities ,or between an entity and it self .Each relationship has a name, an optionally(optional or mandatory),and a degree (how many).A relationship is descried in real terms .Rarely will there be a relationship between every entity and every other entity in an application .If there are only two or three entities ,then perhaps there will be relationships between them all. In a larger application there are not always relationships between an entity and all if the others .Assigning a name ,an optionally ,and a degree to a relationships helps confirm the validity of that relationships .If you cant give a relationships all these things then perhaps there really is no relationships atall.

#### Data Flow Diagram

Data flow diagrams (DFD) are part of a structured model in the development of software. They are a graphical technique that depicts information flow and the transforms that are applied as data move from input to output. Basically, the function of DFDs is to show the user a graphical analysis of a software system. It is like a flowchart, except DFDs show the flow of data throughout the system.

Data flow diagramming shows a system at any level of detail with a graphic network of symbols showing data stores, data processes, and data sources/destinations.

As you can see from the picture, the rectangle represents an external entity. The external entity is a producer or consumer of information that resides outside the bounds of the system to be modeled. The circle (process) is a transformer of information that resides within the bounds of the system. The line with an arrow (data item) is a single item, or a collection of data items. The arrow head represents the directions of the data. The three-sided rectangle (data store) represents a repository of data that is to be stored for use by one or more processes; maybe as simple as a buffer or a queue or as sophisticated as a relational database.

DFDs can be broken down or decomposed to show the system at

increasing levels of detail. The number of inputs (processes or number of circles) a system may have will determine how many levels a user will have will determine how many levels a user will have to go to get a detailed enough design of the software system. Normally, a designer will go three levels deep in the dataflow design (kozar, Kenneth A1997).The purpose of data flow diagrams is to provide a semantic bridge between users and systems developers and to have a commonly understood model of a system. The diagramsare:

* Graphical, eliminating thousands ofwords;
* Logical representations, modeling WHAT a system does, rather than physical models showing HOW it doesit;
* Hierarchical, showing systems at any level of detail;and
* Jargon less, allowing user understanding andreviewing.

Data flow diagrams have the objective of avoiding the cost of;

* + User/developer misunderstanding of a system, resulting in a need to redo systems or in not using thesystem.
  + Having to start documentation from scratch when the physical system changes since the logical system, WHAT gets done often remains the same when technologychanges.
  + Being unable to evaluate system project boundaries or degree of automation, resulting in a project of inappropriatescope.

The literature review of data flow diagramming highlighted the importance of understanding the boundaries of a system and the interfaces that will be required. Additionally, the review showed that DFDs are useful tools for communicating with users. During system development, it may be necessary to understand the movement of data within and outside the system in increasing levels of detail, and the literature review made us aware of this.

#### DFD

**Admin**

**EMS**

On failure

Redirecting to login page

On Success

Entering credentials

Admin login

Invalid login message

**Update**

**Remove**

Details section

**1-level DFD**

**View**

**Add**

**EMS**

Main Screen

**Admin**

**0-level DFD**

Verifying credentials

Entering credentials

EMS

Admin

MYSql Database

(login table)

On failure

Redirecting to login page

On Success

Invalid login message

Main Screen

Update

View

Add

Employee not found

Remove

Removing employee

Enter

EmployeeID

Adding employee

Searching employee

Updating details

E-mail sent if the post is manager

MYSql Database

(Employee\_detail table)

**2-level DFD**

**Database Design**

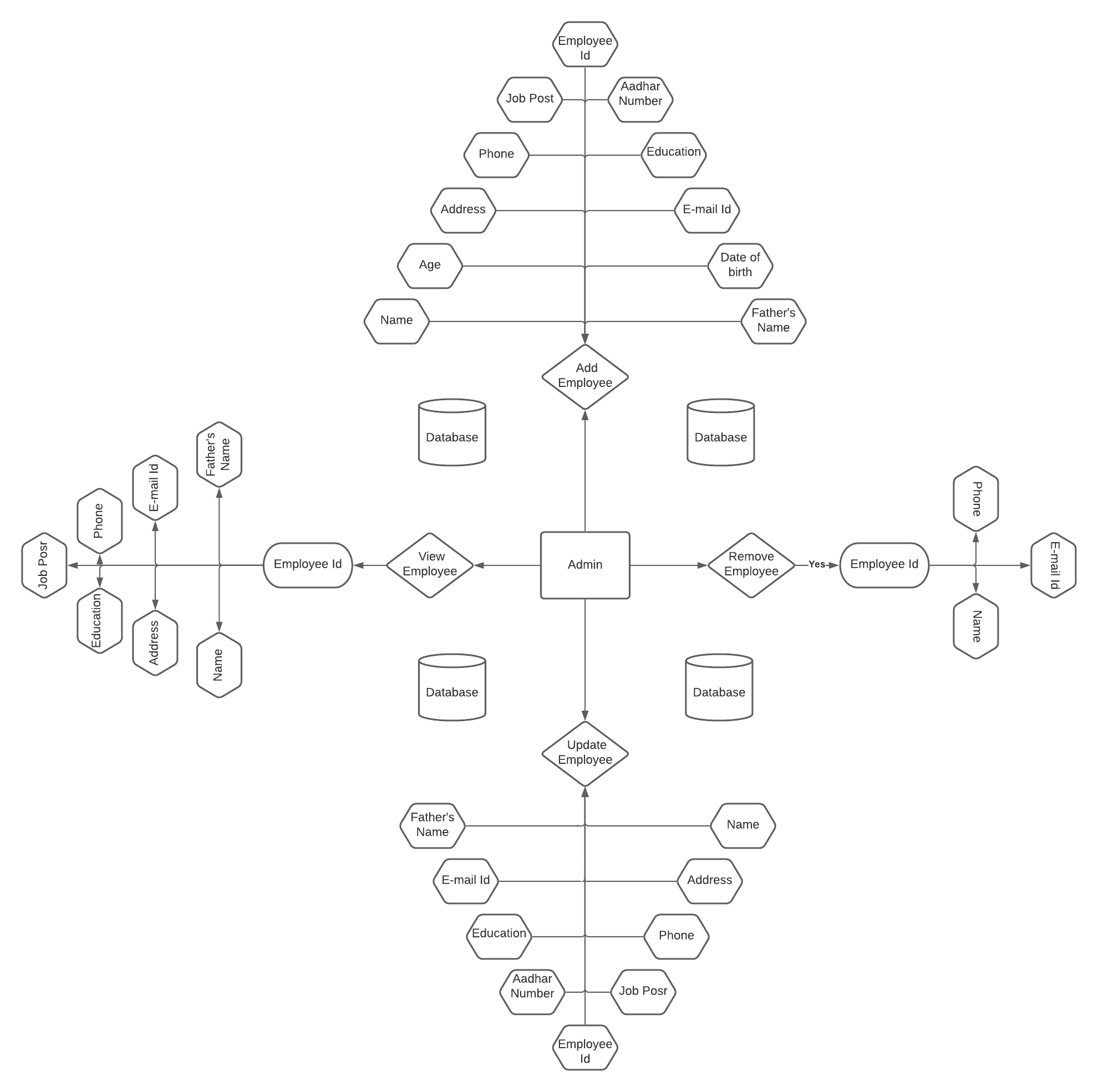
**Introduction**

After getting user’s requirements a close eye was put on designing the database. The main emphasis was given to a fully normalized database with least redundancy keeping all these things in mind database was designed.

The Database (collection of data) is at the core of any information system. The nature of data stored in the database as well as the organization of the data has wide implications on the performance of the system. A good database design should hold all the data, which may be required by the users of the system, but at the same time, avoid any harmful or unnecessary redundancy. The database should also be flexible so that any future changes in the requirements can be incorporated with minimum changes.

The data in Manpower Requisition system is organized in the form of 2-D relational tables. Organization of data into tables allows easy categorization, retrieval and storage of data. Different tables have been created for different types of information. The rules of normalization have also been applied while designing thetables.

**Entity relationship diagram**



## **Technical Requirements**

The technical requirements for the proposed software are outlined below:

##### Hardware Requirements

1. 16 MB RAM memory minimum, 32 MB RAM memory recommended
2. IBM compatible PC with AMD/Intel Pentium P200 MHz processor orabove.
3. Approx. 50 MB hard disk spacerequired
4. SVGA Display running at 1024 x 768resolution
5. Mouse
6. CD-ROM drive for softwareinstallation

##### Software Requirements

1. Windows 9x, 2000 or Xpplatform
2. VB. Net2005
3. Ms-Access

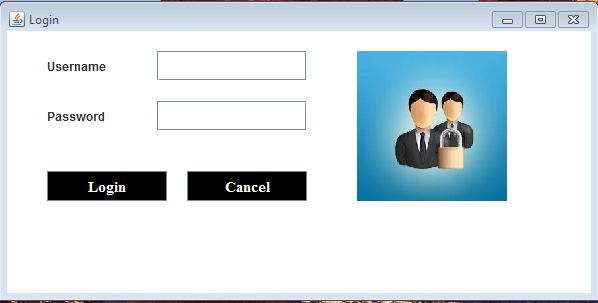
**Snapshots**

#### Splash Screen

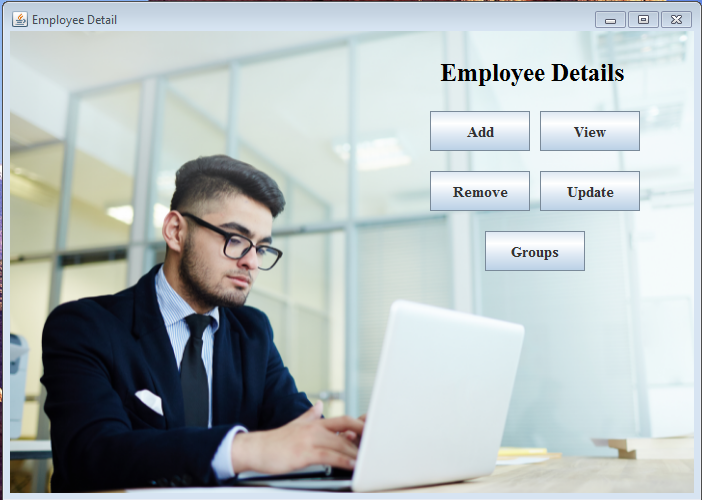


**Login page:**

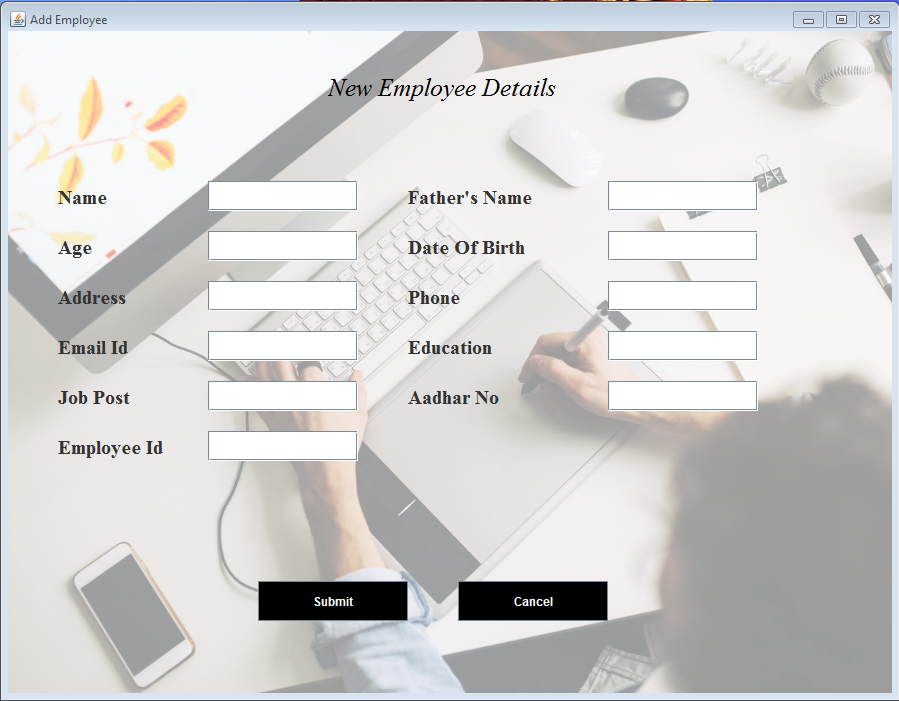
Admin Login by username and password



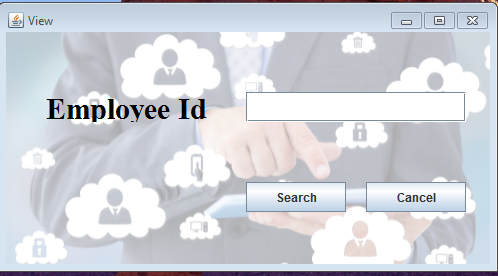
Main Page :



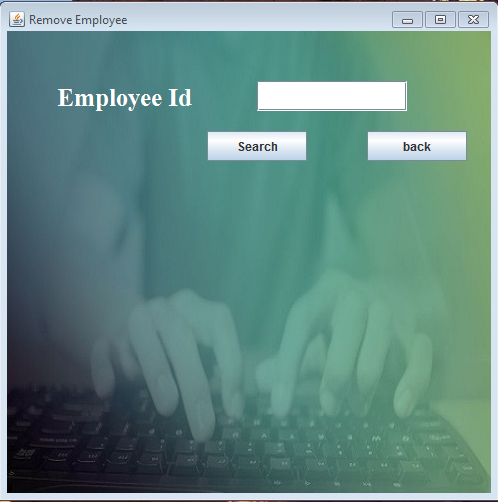
Add Employee Details

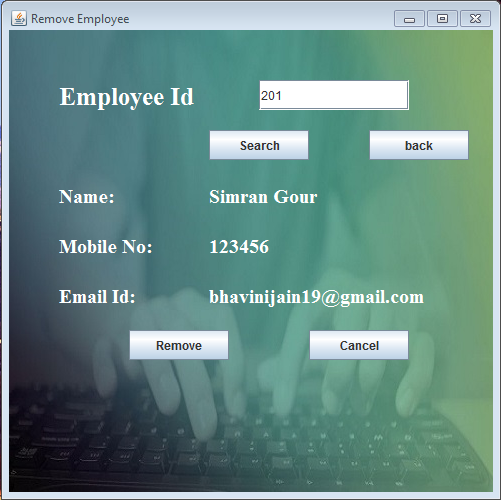


Search Employee by its ID



Remove Employee





Update Employee Details



Employee Groups



Sendin Mail to Employees

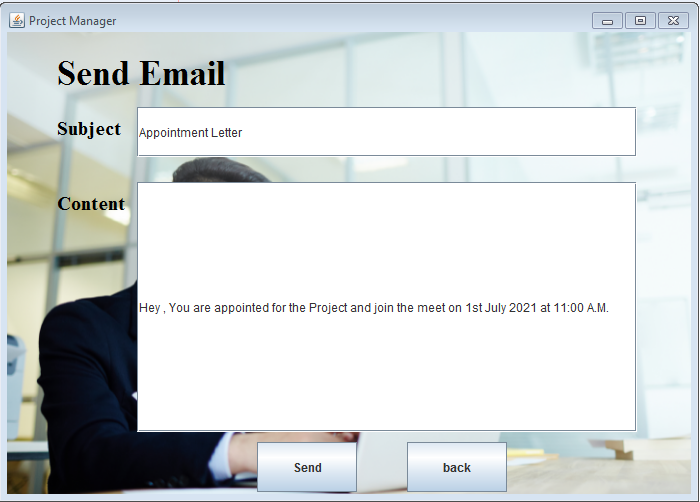
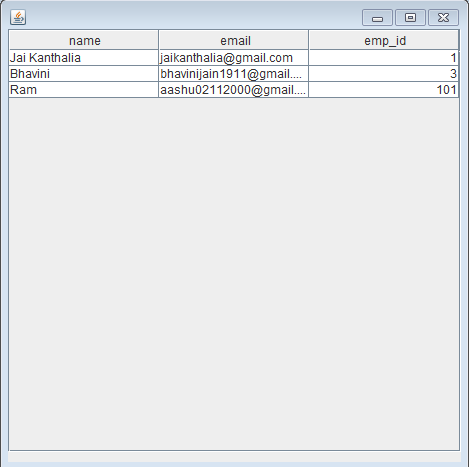


Table of Employees in particular group



## **Suggestion for future work**

* Migrate to **MS SQL** from **MSACCESS**

The reasons for the suggestion are

* 1. Atomicity in access todata
  2. Reliability
  3. Performance
  4. Concurrency
  5. DataSecurity
  6. IndustryStandard

## **Conclusion**

The “Employee Management System” has been developed to solve the various problems that an organization faces when it is working manually. At the same time this software meets the objective to maintain database for futureinterest.

The database was designed with the consultation of the user of the software. The tables were normalized to reduce redundancy and inconsistency; all the data are stored in the database in the form of 2-D relational tables. To perform operations on this database, different programs have been designed and developed. To efficiently carry out the functionality of the “Employee Management System”, a number of user- friendly forms and reports have been developed. All these forms and reports are linked and invoked by the main form. A full-fledged Help system is also provided with theapplication.

The software has been developed keeping in mind the convenience of users. It is developed in such a way that any user without any detailed knowledge of the system can handle with a little training. The user interacts with the software of the system through the messages displayed at various steps. Any illegal action is prompted by an appropriate error message.

## **Bibliography**

* 1. CORE JAVA(Vol-I) by SUN MICROSYSTEMS
  2. [WWW.GOOGLE.COM](http://WWW.GOOGLE.COM)
  3. WWW.JAVA.SUN.COM
  4. [WWW.ONLY4GURUS.COM](http://WWW.ONLY4GURUS.COM)
  5. [WWW.SMARTDRAW.COM](http://WWW.SMARTDRAW.COM)
  6. HEAD FIRST BY KATHY SIERRA & BERT BATES