CORPORATE EMPLOYEE SYSTEM

Sonali Samanta
B.tech: Information Technology, Dronacharya College of Engineering, Gurgaon , India

***Abstract*- The Corporate Employee is a Java application that aims to provide a sync between the employees of a company and the administration of that company. In addition to that, this application also acts as an employee management system. It makes the system easy to monitor and manage employees from different location. This system helps in supervising employees work report and productivity. Employees are the backbone of any company, their management pays a major role in deciding the success of the organization. Employee Management Software makes easy for the employer to keep a track all the records.**

***Index Terms*- corporate, employee, java, company, administrator**

I. INTRODUCTION

Employees are the backbone of any company, their management pays a major role in deciding the success of the organization. The application makes it easy for the employer to keep a track all the records. The Corporate Employee structure is used within a company where administrator maintains the system and manage all the operations like generating log in number for employees, add or delete new vacancies, etc. On the other hand, employees are the second important part of this system. They apply for jobs, promotion and apply if they fit into requirement; and apply for leave as needed. This application works in favor of employees. A company can maintain all the records of employees like jobs, leave etc. Also, administrator can update the status of leave of employee.

II. SCOPE OF APPLICATION

The application has a very vast scope in future. It can be implemented on the Internet 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 Web 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: -

1. The number of levels that the software is handling can be made unlimited in future from the current status of handling up to N levels as currently laid down by the software. Efficiency can be further enhanced and boosted up to a great extent by normalizing and de-normalizing the database tables used in the project as well as taking the kind of the alternative set of data structures and advanced calculation algorithms available.
2. We can in future generalize the application from its current customized status wherein other vendors developing and working on similar applications can utilize this software and make changes to it according to their business needs.
3. Faster processing of information as compared to the current system with high accuracy and reliability.
4. Automatic and error free report generation as per the specified format with ease.
5. Automatic calculation and generation of correct and precise Bills thus reducing much of the workload on the accounting staff and the errors arising due to manual calculations.
6. With a fully automated solution, lesser staff, better space utilization and peaceful work environment, the company is bound to experience high turnover.

III. SYSTEM ANALYSIS

Software requirements analysis is a process of discovery, refinement, modeling, and specification. Requirement analysis proves the software designer with a representation of information, function, and behavior that can be translated to data, architectural interface, and component -level designs. To perform the job properly we need to follow as set of underlying concepts and principles of analysis.

1. INFORMATION DOMAIN

All software applications can be collectively called data processing. Software is built to process data, to transform data from one form to another; that is, to accept input, manipulate it in some way, and produce output. This fundamental statement of objective is true whether we build batch software for a payroll system or real-time embedded software to control fuel flow to an automobile engine.

The first operational analysis principle requires an examination of the information domain and the creation of a data model. The information domain contains three different views of the data and control as each is processed by a computer program:

(1) information contend and relationships (the data model)
(2) information flow, and
(3) Information structure.

To fully understand the information domain, each of these views should be considered.

2. MODELLING

(i) **Functional models**: Software transforms information, and in order to accomplish this, it must perform at least three generic functions:

- Input
- Processing
- And output

The functional model begins with a single context level model (i.e., the name of the software to be built). Over a series of iterations, more and more functional detail is gathered, until a through delineation of all system functionality is represented.

(ii) Behavioral models: Most software responds to events from the outside world. This stimulus/response characteristic forms the basis of the behavioral model. A computer program always exists in some state- an externally observable mode of behavior (e.g., waiting, computing, printing, and polling) that is changed only when some even occurs. For example, in our case the project will remain in the wait state until:

- We click OK command button when first window appears
- An external event like mouse click cause an interrupt and consequently main window appears by asking the username and password.
- This external system (providing password and username) signals the project to act in desired manner as per need.

A behavioral model creates a representation of the states of the software and the events that cause software to change state.

3. FEASIBILITY STUDY

A feasibility analysis usually involves a thorough assessment of the operational, financial and technical aspects of a proposal. Feasibility study is the test of the system proposal made to identify whether the user needs may be satisfied using the current software and hardware technologies, whether the system will be cost effective from a business point of view and whether it can be developed with the given budgetary constraints. A feasibility study should be relatively cheap and done at the earliest possible time. Depending on the study, the decision is made whether to go ahead with a more detailed analysis. When a new project is proposed, it normally goes through feasibility assessment. Feasibility study is carried out to determine whether the proposed system is possible to develop with available resources and what should be the cost consideration. Facts considered in the feasibility analysis were:

- Technical Feasibility
- Economic Feasibility
- Behavioral Feasibility

(i) Technical Feasibility: Technical Feasibility deals with the hardware as well as software requirements. Technology is not a constraint to type system development. We have to find out whether the necessary technology, the proposed equipments have the capacity to hold the data, which is used in the project, should be checked to carry out this technical feasibility.

(ii) Economical Feasibility: This feasibility study present tangible and intangible benefits from the prefect by comparing the development and operational cost. The technique of cost benefit analysis is often used as a basis for assessing economic feasibility. This system needs some more initial investment than the existing system, but it can be justifiable that it will improve quality of service.

Thus feasibility study should center along the following points:

- Improvement resulting over the existing method in terms of accuracy, timeliness.
- Cost comparison
- Estimate on the life expectancy of the hardware
- Overall objective

Our project is economically feasible. It does not require much cost to be involved in the overall process. The overall objectives are in easing out the requirement processes.

(iii) Behavioral/ Operational Feasibility: This analysis involves how it will work when it is installed and the assessment of political and managerial environment in which it is implemented. People are inherently resistant to change and computers have been known to facilitate change. The new proposed system is very much useful to the useful to the users and there for it will accept broad audience from around the world.

IV. OPERATING ENVIRONMENT

Hardware Specification:

(1) Server Side:

- Core 2 Due 2.4GHz and Above
- 2 GB of Random Access Memory and Above
- GB 160 Hard Disk

(2) Client Side:

- Pentium-IV 1.5MHs and Above
- 512 MB of Random Access Memory and Above
- 80 GB Hard Disk

Software Specification:

- Environment: Netbeans 8.1
- Technologies: JSP,CSS,HTML
- Database: MY SQL 5.5
- Software: Netbeans 8.1, Notepad ++
- OS: Windows 10, Windows 8, Windows 7
- Browser: IE7, IE8, FF 3.5, GOOGLE CHROME

V. SYSTEM DESIGN

1. USE CASE DIAGRAM

Use case diagrams are used to gather the requirements of a system including internal and external influences. These requirements are mostly design requirements. So when a system is analyzed to gather its functionalities use cases are prepared and actors are identified. Use case diagrams are considered for high level requirement analysis of a system. So when the requirements of a system are analyzed the functionalities are captured in use cases. So we can say that uses cases are nothing but the system functionalities written in an organized manner. Now the second things which are relevant to the use cases are the actors. Actors can be defined as something that interacts with the system. The actors can be human user, some internal applications or may be some external applications. So in a brief when we are planning to draw a use case diagram we should have the following items identified.

- Functionalities to be represented as an use case
- Actors
- Relationships among the use cases and actors.

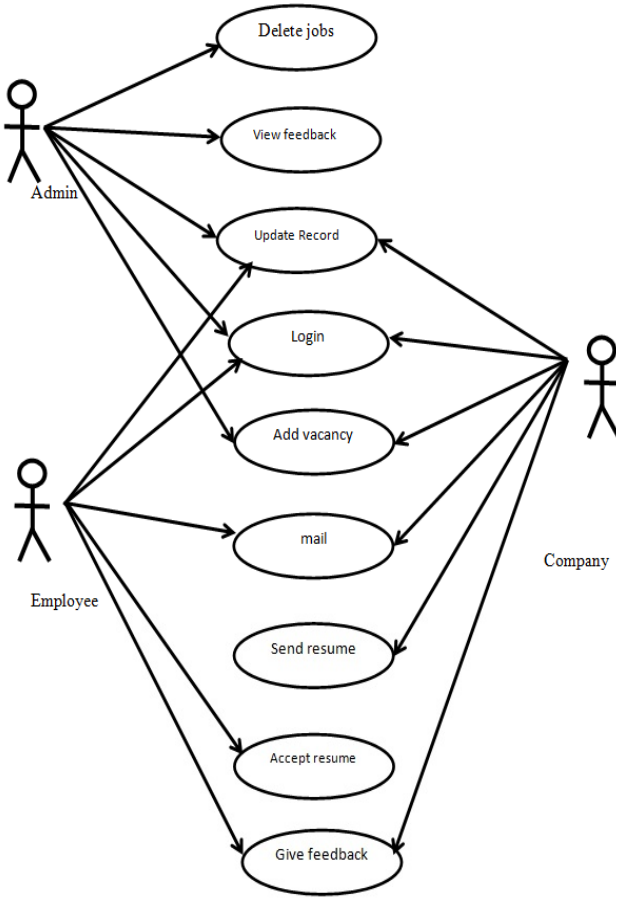

Fig: Case Diagram for Corporate Employee

2. STATE DIAGRAM

A state diagram, also called a state machine diagram or state chart diagram, is an illustration of the states an object can attain as well as the transitions between those states in the Unified Modeling Language (UML). In this context, a state defines a stage in the evolution or behavior of an object, which is a specific entity in a program or the unit of code representing that entity. State diagrams are useful in all forms of object-oriented programming (OOP). The concept is more than a decade old but has been refined as OOP modeling paradigms have evolved

A state diagram resembles a flowchart in which the initial state is represented by a large black dot and subsequent states are portrayed as boxes with rounded corners. There may be one or two horizontal lines through a box, dividing it into stacked sections. In that case, the upper section contains the name of the state, the middle section (if any) contains the state variables and the lower section contains the actions performed in that state. If there are no horizontal lines through a box, only the name of the state is written inside it. External straight lines, each with an arrow at one end, connect various pairs of boxes. These lines define the transitions between states. The final state is portrayed as a large black dot with a circle around it. Historical states are denoted as circles with the letter H inside.

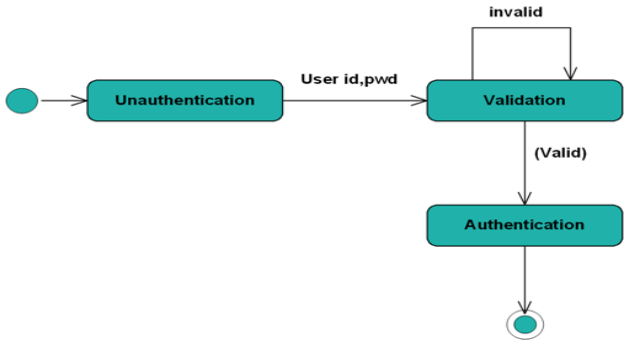

Fig: State Diagram for Administrator

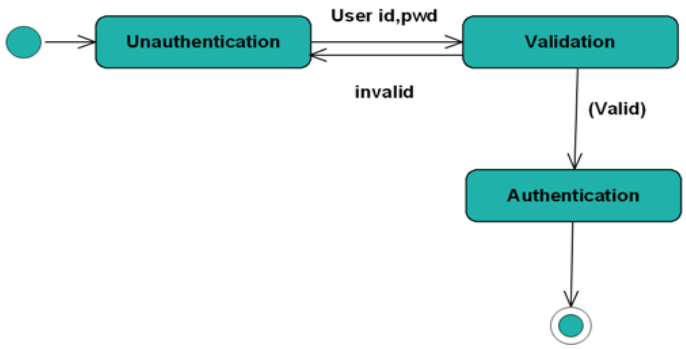

Fig: State Diagram for User

3. DATA FLOW DIAGRAM (DFD)

A data flow diagram (DFD) illustrates how data is processed by a system in terms of inputs and outputs. As its name indicates its focus is on the flow of information, where data comes from, where it goes and how it gets stored.

DFD contains a shape that represents the system to model. It also shows the participants who will interact with the system, called the external entities. In this example, Company, New Individual, Registered Individual and Admin are the entities who will interact with the system. In between the process and the external entities, there are data flow

(connectors) that indicate the existence of information exchange between the entities and the system.

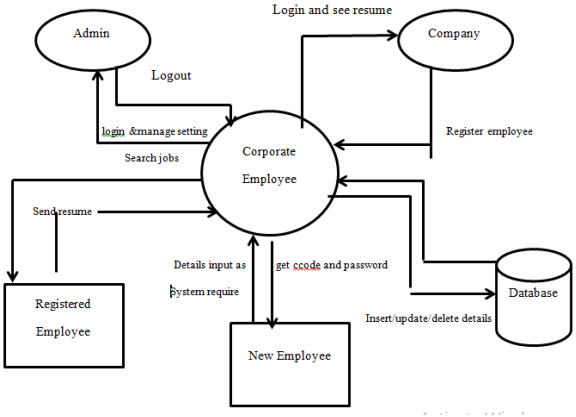

Fig: Zero Level DFD

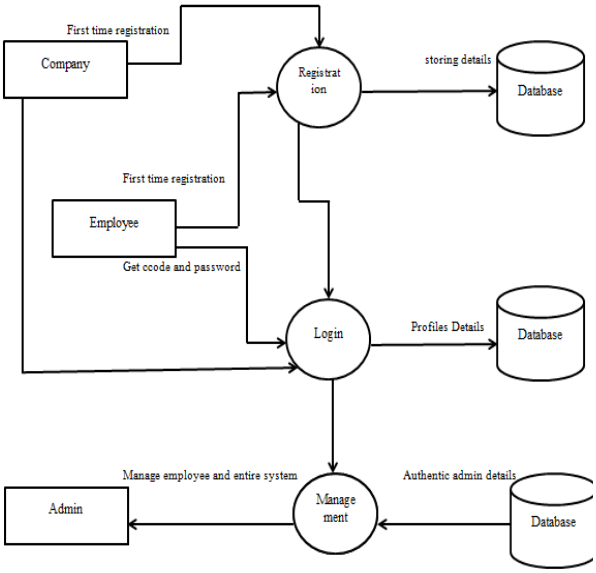

Fig: First Level DFD

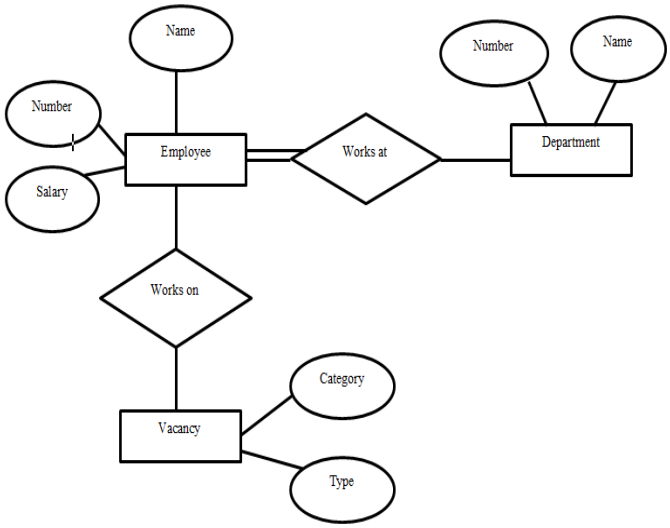

Fig: ER Diagram

VI. TESTING

1. SYSTEM TESTING

Once source code has been generated, software must be tested to uncover (and correct) as many errors as possible before delivery to customer. Our goal is to design a series of test cases that have a high likelihood of finding errors. To uncover the errors software techniques are used. These techniques provide systematic guidance for designing test that

(1) Exercise the internal logic of software components, and

(2) Exercise the input and output domains of the program to uncover errors in program function, behavior and performance.

Software is tested from two different perspectives:

(1) Internal program logic is exercised using “White box” test case design techniques.

(2) Software requirements are exercised using “black box” test case design techniques.

In both cases, the intent is to find the maximum number of errors with the minimum amount of effort and time.

2. STRATEGIES

A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high-level tests that validate major system functions against customer requirements. A strategy must provide guidance for

the practitioner and a set of milestones for the manager. Because the steps of the test strategy occur at a time when deadline pressure begins to rise, progress must be measurable and problems must surface as early as possible.

Following testing techniques are well known and the same strategy is adopted during this project testing.

2.1 Unit testing: Unit testing focuses verification effort on the smallest unit of software design- the software component or module. The unit test is white-box oriented. The module interface is tested to ensure that information properly flows into and of the program unit under test the local data structure has been examined to ensure that data stored temporarily maintains its integrity during all steps in an algorithm's execution. Boundary conditions are tested to ensure that the module operated properly at boundaries established to limit or restrict processing. All independent paths through the control structure are exercised to ensure that all statements in a module haven executed at least once.

2.2 Integration testing: Integration testing is a systematic technique for constructing the program structure while at the same time conducting tests to uncover errors associated with interfacing. The objective of this test is to take unit tested components and build a program structure that has been dictated by design.

2.3 Validation testing: At the culmination of integration testing, software is completely assembled as a package, interfacing errors have been uncovered and corrected, and a final series of software tests—validation testing-may begin. Validation can be defined in many ways, but a simple definition is that validation succeeds when software functions in a manner that can be reasonably expected by the customer.

2.4 System testing: System testing is actually a series of different tests whose primary purpose is to fully exercise the computer-based system. Below we have described the two types of testing which have been taken for this project.

2.4.1 Security testing

Any computer-based system that manages sensitive information causes actions that can improperly harm (or benefit) individuals is a target for improper or illegal penetration. Penetration spans a broad range of activities: hackers who attempt to penetrate system for sport; disgruntled employees who attempt to penetrate for revenge; dishonest individuals who attempt to penetrate for illicit personal gain. For security purposes, when anyone who is not authorized user cannot penetrate this system. When programs first load it check for correct username and password. If any fails to act according will be simply ignored by the system.

2.4.2. Performance Testing

Performance testing is designed to test the run-time performance of software within the context of an integrated system. Performance testing occurs throughout all steps in the testing process. Even at the unit level, the performance of an individual module may be assessed as white-box tests are conducted.

2.5 Validation Checks

Software testing is one element of broader topic that is often referred to as verification and validation. Verification refers to the set of activities that ensure that software correctly implements a specific function. Validation refers to a different set of activities that ensure that the software that has been built is traceable to customer requirements. Boehm state this another way:

Verification: "Are we building the product right?"

Validation: "Are we building the right product?"

Validation checks are useful when we specify the nature of data input. Let us elaborate what I mean. In this project while entering the data to many text box you will find the use of validation checks. When you try to input wrong data. Your entry will be automatically abandoned.

In the very beginning of the project when user wishes to enter into the project, he has to supply the password. This password is validated to certain string, till user won't supply correct word of string for password he cannot succeed. When you try to edit the record for the trainee in Operation division you will find the validation checks. If you supply the number (digits) for name text box, you won't get the entry; similarly if you data for trainee code in text (string) format it will be simply abandoned.

A validation check facilitates us to work in a greater way. It become necessary for certain applications like this.

VII. CONCLUSION

The aim of this project is to help the employees or the people who are connected to a company directly or indirectly. This is web based project so it can modify in future if needed. In this report all the needs of a corporate employee are considered which means any employee can use this by logging with his c code which is equals to user id and password and complete his profile after getting the permission of admin.

REFERENCES

1. www.google.co.in
2. www.youtube.com
3. http://wikipedia.org/
4. www.webopedia.com
5. www.mysql.com
6. www.javatutorial.com
7. www.sun.com
8. http://www.java2s.com
9. www.w3schools.com
10. Material provided by the HCL-CDC
11. Herbert Scheldt – Java 2: The Complete Reference, Fifth Edition. This book covers all aspect of java programming language.
12. Head First Java-Second Edition Authors: Bert Bates, Kathy Sierra