REPORT

TITLE OF THE PROJECT

Email Service Automation System

CHAPTER 1

Introduction

ABSTRACT & INTRODUCTION

This project is about developing a web based mail client connecting to windows Server running a Mail Server. This Project has the following main functionality

1. Receiving/Sending/organizing mails.
2. Sending mail using send mail.
3. Performing Admin functions like managing new user, resetting passwords etc.

The software is fully integrated with **CRM** (Customer Relationship Management) as well as **CMS** (Content Management System) solution and developed in a manner that is easily manageable, time saving and relieving one from manual works.

EXISTING SYSTEM:

- Cannot Upload and Download the latest updates.
- No use of Web Services and Remoting.
- Risk of mismanagement and of data when the project is under development.
- Less Security.
- No proper coordination between different Applications and Users.
- Fewer Users - Friendly.
- Manual system need man power a lot.
- Communication between Patient and administration is a tuff job.
- Difficult to maintain each and patient information in form of files.

PROPOSED SYSTEM:

The development of the new system contains the following activities, which try to automate the entire process keeping in view of the database integration approach.

- User friendliness is provided in the application with various controls.
- The system makes the overall project management much easier and flexible.
- Readily upload the latest updates, allows user to download the alerts by clicking the URL.
- There is no risk of data mismanagement at any level while the project development is under process.
- It provides high level of security with different level of authentication.

OBJECTIVES

1. Capability to create user Email Accounts by an Administrator or by End users after registering themselves
2. Administrator functionality to Delete User Accounts, Change passwords
3. Capability for End users to login into the system using a browser
4. Capability for logged in users to send/receive/forward/reply/delete mails
5. Invalidate user login on inactive for more than 10mts
6. Address book capability
7. Mark mails as Junk
8. Apply Label to Mail
9. Organize mails in Logical Folders

CHAPTER

2

System Study

2.1 System under Study

In order to understand the need of the proposed system first we will have to understand the demand and supply. Of such system accordingly in the real world , and in nowadays context.

2.2 The manual System:

In the manual system Employee register all thing related with metro corporation but now it is easy to maintain, the manula system is to time taken.

2.3 Need for the present System:

The project "**EMAIL SERVICE AUTOMATION SYSTEM"** a system, it deals all task related with metro corporation.

2.4 Objective:

- Maintain the whole information about the Mail services.
- Maintain the whole information about client information.
- Maintain the information about the mail details.
- Generate reports.
- Easy registration (any type)
- Login dependent information retrieval and updating.

- Correct and consistent maintenance of data and its quick retrieval.
- Smooth transition and easy access to the pages not requiring much training to the end user.

CHAPTER 3

Project Planning

3.1 The Concept

The concept behind software project planning is to provide a framework that enable the manager to make reasonable estimates of the resources, cost and schedule. The appropriate estimates are formulated within a time limit and the objective is achieved through a process of information discovery, which leads to reasonable estimates.

In our Context

We studied and went through all set of tasks and activities regarding software and identified them before software project is to be implemented. With the help software project planning we constituted an identified works and resources required to complete the project and also the time required for the completion of project. Planning helped us in decision-making process while we were studying how to initiate and what steps to follow to accomplish the task. We also planned to finish the assigned task within appropriate cost, effort and resources to maintain the system feasible.

Our planning was about all kinds of resources required and estimation regarding total cost. We also finalized how much time it will take to launch the s\w with in the given constraint of the cost, effort and resources. In resources we also considered human resources, reusable software resources and environmental resources.

3.2 Software Scope:

Scope of software is the parameter under which the software is applicable or it is the unambiguous and understandable

domain of the software in which the system is applicable. The scope of the software can be determined by following factors:

The Context: By understanding the context of the software (i.e. how does the software to be built to be fit into a large system, product, or business context and what constraint are required as a result of the context?), we can easily determine the scope of the software.

The Information objectives: Here, we try to know the user visible outputs and inputs. By understanding the end-user visible data objects (inputs/outputs), we determined the range of the information on which we have to work upon and for what purpose.

Function and performance: Here, we tried to know about the different functions, which transform the input data into output. By knowing about the different functions, we came to know about the different processes of the system.

The above said factors and their analysis helped us to exploit and implement the following features of software scope:

Reusability: Reusability is possible as and when we required in this application. We can update it next version. Reusable software reduces design, coding and testing cost by amortizing effort over several designs. Reducing the amount of code also simplified understanding, which increases the likelihood that the code is correct. We followed up both types of reusability as sharing of newly written code within a project and reuse of previously written code on new projects.

Extensibility: This application software is extended in ways that its original developers may not expect. The following principles enhance extensibility like Hiding data structures, avoiding traversing multiple links or methods, avoiding case statements on object type and distinguishing public and private operations.

Robustness: its method is robust and it will not fail even if it receives improper parameters. There are some alert pages and messages is flashed out with some dialogue boxes to warn and inform the end user about the current processes going on. It also interacts with the user by alerting them about invalid parameters.

Understandability: A method is understandable if anyone other than the developer of the method can understand the code (as well as the developer after a time-span).

Cost-effectiveness: Its cost is under the budget and developed within given time period. It is always desirable to aim for a system with a minimum cost subject to the condition that it must satisfy all the requirements.

To implement Software Scope we considered it by looking out on the future scope of the software. The software is being implemented on a web site having server any were and Clint any computer on the net. The codes being embedded and installed at the root directory of the server with support for SQL and Internet Information Services. The database space taken at SQL server and connected with the site server will be main power for the resource utilization on the site. The software scope enlarges ultimately with the use of internet technologies when universities organization institutes and individual can use and extract the vast and huge possibility of this site engulfed by the net technologies.

3.3 Resource Estimation

The second task of planning is estimation of resources required to accomplish the software development effort. The development environment encapsulates hardware and software tools are the foundation of the resource and provide the infrastructure to support

the development effort. At a higher level we encounter reusable software components – software building blocks that can dramatically reduce development costs and accurate delivery. At the highest level is the primary resource – people.

3.3.1 Human Resources

Human resource is the primary resource and lies at the topmost level of the resources hierarchy. It is the resource, which says about the people who are technically sound and develops the software. We started by evaluating scope and studying the skills required to complete the development. Both organization position and specialty are specified. Since I am only in my team with one project coordinator so we evaluated our skills and updated ourselves with the required skills and divided the work among us and so calculated the estimated schedule.

3.3.2 Reusable Software Resources

It lies at the second priority level of resources but any discussion would be incomplete without recognition of reusability. Since the task we are assigned is a new one and is to be developed from the very bottom level so we do not perform the planning and estimation regarding reusable software resources.

3.3.3 Environmental Resources

The environment that supports the software project often called a software engineering environment (**SEE**) incorporates hardware and software. Hardware provides a platform that supports the tools (software) required to produce the work products that are an outcome of good software engineering practices. Because most software organizations have multiple constituencies that require access to the **SEE**, a project planner must prescribe the time window required for hardware

and software and verify that these resources will be available. When a computer-based system (incorporating specialized hardware and software) is to be engineered, the software team may require access to hardware elements being developed by other engineering teams. The software project planner must specify each hardware element.

3.3.4 Project Estimation

Software is most expansive element of any computer-based information system. So, the estimation of cost and effort applied to the project is crucial factor in system development. The major problem with software project estimation is that there are so many variables that affect the estimation and these too are difficult to quantify. Some of the variables that affect the estimation are – human, technical, environmental, and political. This does not mean we don't have the solution. There are a number of techniques, which has been developed for the estimation purpose. Some of the important techniques we adopted are as follows:

a) Decomposition Techniques
b) Empirical Estimation Models and
c) Automated Estimation Tools

3.4 Decomposition

Software project estimation is a form of problem solving, and in most cases, developing a cost and effort estimate for a software project is too complex to be considered in one piece. So, we decompose the problem, and recharacterize it as a set of smaller problems. But no matter which technique we apply for our project, the first primary step in software project estimation is "The determination of the size of the software project ".

The accuracy of estimation of software project is dependent on the degree to which the size of the software has been estimated. The direct method to find the size of the software is to find the LOC (Lines of Code) of the software but this method is very difficult at the very beginning of the project. Because even the most experienced people will find it next to impossible to predict LOC before hand. There are various approaches to major the size of the software. Some of the important approaches are as follows:

a) "Fuzzy logic " Sizing
b) Function Point Sizing
c) Standard Component Sizing and
d) Change Sizing

Here, in our case, we have used the functional point sizing of the project. Let us have a brief review of the Function Point sizing concept.

Function Point Sizing: Albrecht first proposed the concept of functional point. Function Points are derived using an empirical relationship based on countable (direct) measures of software's information domain and assessment of software complexity. Functional points are computed by the formula

$$FP = \text{Count Total} \times [\, 0.65 + 0.01 \times \textstyle\sum (Fi) \,]$$

Where, Fi is the value in the range of zero to five for various types of complexity factors in the project.

CHAPTER 4

Project Scheduling

4.1 The Concept:

Software project scheduling is an activity that distributes estimated effort across the planned project duration by allocating the effort to specific software engineering tasks. Every project has a deadline to complete. So, software projects also have a deadline to meet. To complete a software project within a time frame, proper scheduling should be done. In other words, we can say that successful software projects are those that have been successfully placed a schedule. A software project scheduling involves plotting project activities against a time frame. So, the scheduling is about developing a road map structure or a network based on analysis of the tasks that must be performed to complete the projects. The schedule evolves over time. During early stages of project planning, a microscopic schedule is developed. This type of schedule identifies all major software-engineering activities and the product functions to which they are applied. As the project gets under way, each entry on the macroscopic schedule is refined into a detailed schedule. Here, specific software tasks are identified and scheduled. The scheduling objective is to define all project tasks, build a network that depicts their interdependencies, identity the tasks that are critical within the network, ant then track their progress to ensure that delay is recognized " one Day at a time". To accomplish this, the manager must have a schedule that has been defined at a degree of resolution that enables the manager to monitor progress and control the project.

4.2 Pert & Gantt Chart

Program evaluation and review technique (PERT) and *critical path method* are two project scheduling methods that are applied to software development. Both techniques are driven by information already developed in earlier project planning activities. Gantt charts are a project control technique that can be used for several purposes, including scheduling, budgeting and resources planning. A Gantt chart is a bar chart, with each bar is proportional to the length of time planned for the activity.

Inter task dependencies cannot be shown using these techniques. PERT charts show tasks inter dependencies directly. It is organized by events and activities or tasks. PERT controls time and cost during the project and also facilitate finding the right balance between completing a project on time and completing it within the budget. In my project a set of tasks was performed for the project development. The major identified Software Engineering Tasks at software project development are as follows:

1. System Analysis
2. System Design
3. Software Development
4. Software Testing
5. Software Installation
6. Software Documentation
7. System Evaluation

The timeline chart of the above task set (Time Frame: 3 months 15 days) has been prepared. The PERT Chart of the software development phase is also prepared and is depicted in figure 1. It is based on the different modules identified in the software.

Project Development Scheduling

PERT (Program Evaluation & Review Technique) Chart

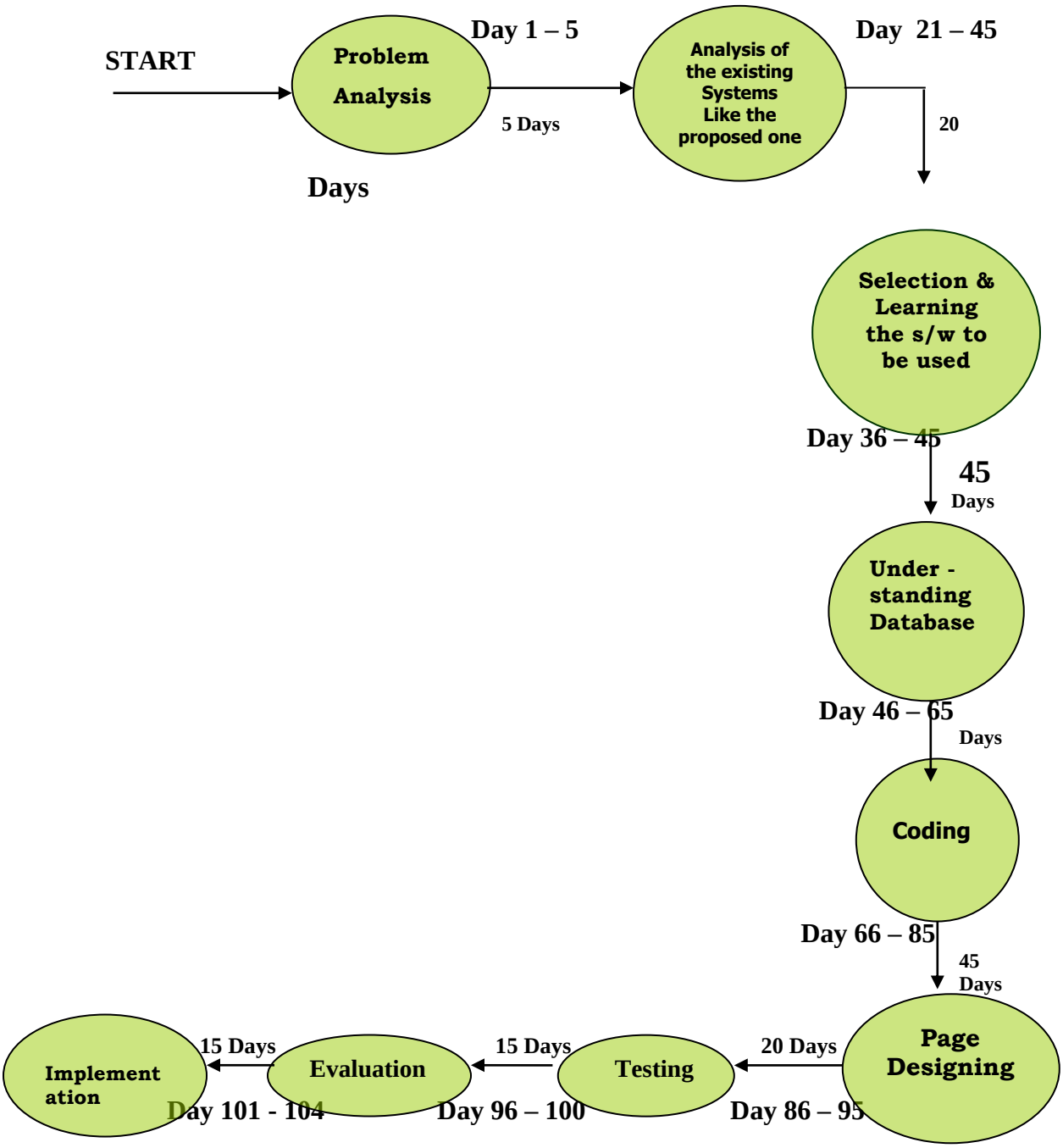


Figure 1: PERT Chart

CHAPTER 5

System Analysis

5.1 Problem Identification

Problem identification is very first step of system analysis. The analyst meets with the candidate and the company's placement. Here in our case both candidates and the company placement and end-user are the former and candidate . In this step, we are interested to understand the product's objective and defined the goals required to meet the objective. After identifying overall goals, we moved towards evaluation of supplementary information during preliminary investigation. We also focused on "does the technology exit to build the system?" and "what bounds have been placed on costs and schedule". To perform it we scanned followings:

- The performance of the system
- The information being supplied and its form
- The economy of processing
- The control of the information processing
- The efficiency of the existing system
- The security of the data & software

5.2 Preliminary Investigation

Once the request for the information system development is received, the first system activity " *preliminary investigation* " begins. This activity is independent of the software-engineering paradigm. That means whether a system will be developed by means of the system development life cycle (SDLC), a prototyping strategy, or the structured analysis method, or a combination of these methods, preliminary investigation must be performed first. This activity has three parts:

- Request Clarification
- Study and
- Approval

It may be possible that the request that has been received is not clearly defined and it is difficult to figure out what for the request is. So, in that case, the person who has made the request is contacted and asked to explain the request. If the request is appropriate enough then in that case feasibility study is carried out for that request. I.e. before the feasibility study, the request must be determined and defined precisely. After request clarification the feasibility study is performed. If the request for the information system is feasible in the request is approved and then starts the real analysis phase called "*Requirement study*". The data that are collected from the organization as the part of preliminary investigation is follows three mechanisms also called " *Fact Finding Techniques* ":

- Reviewing existing Documents material and existing systems (web sites)
- Conducting Interviews and
- On-Site Observation

To conduct preliminary investigation, we contacted with the perfect placement like noida , employment exchange, mrg placement etc. The very first impression validates the request for the automation (i.e. Information System development). I talked with head of the company and the subordinates about the current working system and about the tasks they were performing. We used all the three fact-finding techniques for data collection. We collected various papers & documents related to the activities like the objective and subjective exams. And conducted marketing surveys to know the interest among the people associated and attached with different organization .It was surprising to know that most of them were very interested in computers and has the better understanding of Computer & Information Systems. Both were very helpful in passing the exact information we were looking for. Reviewing the

traditional one and having own experience in the field helped us immensely in the knowing the data items being processed by the system. It also helped in understanding the complexities of the working process of the web site.

Latter on, we examined the various kinds of reports and result the company were producing. The above said fact finding techniques not only helped in the understanding of the system but also proposed some solutions to the new information systems.

5.2.1 Information Content:

Information content represents the individual data and control objects that constitute some larger collection of information transformed by the software. In our case the object, which is composite of a number of important pieces of data as *project*, *package* and item with its category. So the contents is defined by the necessary attributes required to create it. During the analysis of the information domain the inter relationship between objects is also defined.

5.2.2 Information Flow:

Information flow represents the manner in which data and control changes as each moves through a system. Input objects are transformed into intermediate information, which is further transformed to output. Here, additional information is also supplied and transformations are some engineering functions/formulae.

5.2.3 Information Structure:

Information structure represents the internal organization of various data and control items. Some queries are answered like “how does information in one information structure relate to information in another structure?”, “is all information contained within a single structure or are distinct structure to be used?”.

5.2.4 Feasibility Study:

5.3.1 Introduction:

The feasibility study of any system is mainly intended to study and analyze the proposed system and to decide whether the system under consideration will be viable or not after implementation. That is it determines the usability of the project after deployment. To come to result a set of query is answered keeping the efficiency of the software and its impact on the domain for which it was developed. It main emphasis is on the following three questions elucidated below as:

What are the user's requirements and how does a candidate system meet them?

What resources are available for the proposed systems? Is it worth solving the problem?

What is the likely impact of the proposed system on the organization? I.e. how does the proposed system fit with in the organization?

Thus since the feasibility study may lead to commitment of large resources, it becomes necessary that it should be conducted competently and no fundamental errors of judgment are made. Different types of feasibility study and the way we performed on our project "**EMAIL SERVICE AUTOMATION SYSTEM"**

5.2.5 Technical Feasibility:

In technical feasibility, we study all technical issues regarding the proposed system. It is mainly concerned with the specifications of the equipments and the software, which successfully satisfies the end-user's requirement. The technical needs of the system may vary accordingly but include:

- The feasibility to produce outputs in a given time.
- Response time under certain conditions.

- Ability to process a certain volume of the transaction at a particular speed.
- Facility to communicate data.

Under this analysis process questions like (i) does the compatible platform exist within our domain or can we procure it? (ii) Does the proposed equipment have the technical capacity to hold the data required using the new system? Both at the development site and at server where we will be hiring the space for the website, and also the database (iii) would it be possible to upgrade the system after it is developed and implemented, if necessary? And (iv) would the recommended technology guarantee the reliability, accuracy and data security? This analysis process requires more emphasis on system configuration given more importance rather than the actual hardware specifications.

. The configuration of the existing systems is:

- Processor : Intel Pentium-IV
- Memory : 256 MB,
- Network adapter : Ethernet adaptor
- Modem : 56kbps voice fax data
- Secondary storage : Samsung/Seagate hard disk (40 GB)
- Printers: 2 Inkjet and one HP Laser Printer
- Internet: 128 kbps cable internet

The data will reside at server root directory installed with SQL server 7 and IIS. For Software there are following alternatives:

- Front End: ASP.net
- Back End: SQL server 2010
- Editor: Micro media dreamweaver
- Documentation tool: MS-Word

Features of SQL SERVER 2010

SQL SERVER 2010 provides one of the best security features among all RDBMS. With low cost and common availability in spite of having oracle as an option it was better to use SQL server 2010. As we know that the database is a repository for stored, operational data in a database environment and common data are available and used by several users. Instead of each program (or user) to manage its own data, the data across applications are shared by all authorized users with the help of database software managing the data as an entity.

The general concept behind a database is to handle information as an integrated whole. A database is a collection of interrelated data stored with minimum redundancy to serve many users quickly and effectively. The general objective is to make information easy, quick, expressive, and flexible for the user. In database design specific objectives are considered: -

- Ease of learning and use.
- More information at low cost.
- Accuracy and integrity.
- Recovery from failure.
- Performance.

In this way, *S.Q.L server 2010* is one of the leading R.D.B.M.S. software in the world. It is characterized by the quick retrieval of information from huge tables. This quality allows it to cater to the ever-changing business needs of the present age. It supports fourth generation language, SQL, thereby making it easier for the customers to grasp it, a development language where complicated procedures, functions etc. can be used.

The *S.Q.L server* 2010 include following features**:**

- Queries
- Constraints
- Procedures
- Triggers
- batch implementation
- functions
- bulk copy utility
- cursores
- stored procedures

5.3.3 Economical Feasibility:

Economic feasibility is the most frequently used technique for evaluating the effectiveness of the proposed system. More commonly known as cost/benefit analysis, the procedure is to determine the benefits and savings that are expected from the proposed system and compare them with the costs. If the benefits outweigh costs, a decision is

taken to design and implement the system otherwise further justification or the alternative in the proposed system will have to be made if it is to have a chance of being approved. This outgoing effort improves in accuracy at each phase of the system development life cycle.

Classification of Costs & Benefits: The various cost expected to incur & benefit expected to realize are given below.

Tangible or Intangible Costs & Benefits: Tangibility refers to the ease with which costs or benefits can be measured. An outlay of cash for a specific item or activity is referred to as a tangible cost. They are shown as disbursements on the books. The purchases of hardware or software, personnel training are examples of tangible costs. They are readily identified and measured.

Costs that are known to exist but whose financial value cannot be accurately measured are referred to as an intangible costs. Benefits are also classified as tangible or intangible. Like costs, they are often difficult to specify accurately. Tangible benefits such as saving of material cost are quantifiable. Intangible benefits such as improved institution image is also taken into account in the system evaluation process.

Direct or Indirect Costs & Benefits: Direct costs are those with which a monetary figure can be directly associated in a project. They are applied directly to the operation. Direct benefits can also be specified to the given project. Indirect costs are the results of operation that are not directly associated with the given system or activity. They are often referred to as overhead. Light, air conditioning, maintenance, protection of the computer center is all tangible costs, but it is difficult to determine the proportion of each attribute to a specific activity. Indirect benefits are realized as a by-product of another activity or system.

Fixed and Variable Costs and Benefits: Some costs and benefits are constant, regardless of how well a system is used. They are constant and

do not change. Once encountered they will not reappear. In contrast, variable costs are incurred on a regular basis. They are proportion to work volume and continue as long as is in operation. For example, the cost of printer paper and cartridge. Fixed benefits are also constant and do not change. The variable benefits, on the other hand, are realized on a regular basis. For example, the system will save several hours of employees in preparing each report, which otherwise was prepared manually. Another examples are error free data entry, help about different topics and ease of use.

5.3.4 Operational Feasibility:

Proposed projects are beneficial only if they can be turned into information system that will meet the operating requirements for which they are designed and developed. It is mainly related to human organizational and political aspects. People are resistant to change, and computers have been known to facilitate change. Findings of operational feasibility analysis can be summarized as given below:

- Since the audiences are million of netigen who use it for information retrieval. So their will be many who will be benefited with this project
- End users are always welcome to have the technical support on the site.

CHAPTER 6

Software Engineering Paradigms Applied

6.1 Introduction

Win engineering as software engineering is a layered technology, which comprises of four independent layers as a quality focus layer, the process layer, the methods layer, and the tools layer. The foundation for the software engineering is the process layer and together with the technology layer helps the rational and timely development of computer software. The method layer provides a technical way to perform a broad array of tasks regarding requirements analysis, design, program construction, testing, and maintenance. The tools layer encompasses the software engineering tools to provide automated or semi automated support for the process and the methods.

In real world, the software development is a teamwork that incorporates a development strategy that encompasses the process, methods, and tools layers and so the strategy is termed as a process model or Software Engineering Paradigms.

According to Fritz Bauer “Software engineering is the establishment and use of sound engineering principles in order to obtain economically software that is reliable and works efficiently on real machines”. There are several models suggested to go through the software process. The one most suitable & we opted the linear sequential model to develop the software.

6.2 System Engineering

The software is always part of a large system and is built using modular approach. The very scratch work begins by establishing requirements for all system elements and then allocating some subset of these requirements for all system elements and then allocating some subset of these requirements to the software. This system view is essential whenever the software is intended to interact with other system elements such as hardware, people, and database. In due course

the system engineering and the analysis encompass requirements gathering at the system level with a small amount of top-level design and analysis. Information engineering encompasses requirements gathering at the strategic business level and at the business area level.

6.3 Software Requirement Analysis

In the very first approach of software development, the requirements gathering process is performed and later intensified and focused on the software. To understand the various characteristics of the software to be built, first we must have to understand the information domain for the software, and the required function, behavior, performance, and interface. Requirements for both the system and the software are documented and must be reviewed with the customer. Keeping this software engineering paradigm we are having a proper documentation on our web site so that Customer's requirement should be reviewed with the user. It helped us to minimize the distance between the actual requirement and the software required and hence kept and will keep the customer's satisfaction very high.

6.4 Design

The design is almost and always a multi step process and focuses on four distinct attributes of a program:

- Data Structure
- Software Architecture
- Interface Representations and
- Procedural Detail.

The design process translates requirements into a representation of the software that can be assessed for quality before coding begins. Like requirements, the design is documented and becomes part of the software configuration.

6.5 Code Generation

The design is translated into a machine-readable form using some programming tools.

6.6 Testing

Once the design is converted into machine-readable form the program testing phase starts. The testing process on our software mainly focused on the logical internals of the software, ensuring that all statements have been tested properly, and on the functional externals. Through testing we mainly intended to uncover errors and ensured that the defined input produces the desired results that agree with required results in the specification.

6.7 Maintenance

Software will undoubtedly undergo change after it is delivered to the customer. Change will occur due to either errors have been encountered or the software have been adapted to accommodate in its external environment or the customer requires functional or performance enhancements. Software support/ maintenance reapplies each of the preceding phases to an existing program rather than a new one. The linear nature of the classic life cycle worked perfect without any "blocking states" as the project is to be developed by me under the guidance of the project coordinator only & hence we started the next phase as soon as the current phase is finished without waiting for any one else.

6.8 The Linear Sequential Model

The Linear Sequential Model, the Classic life cycle, or the Waterfall model suggests a systematic and sequential approach to the software development. The very first phase starts from the

system level and progresses through analysis, design, coding, testing, and support. A pictorial view of opted model to develop the software is depicted in figure 1.

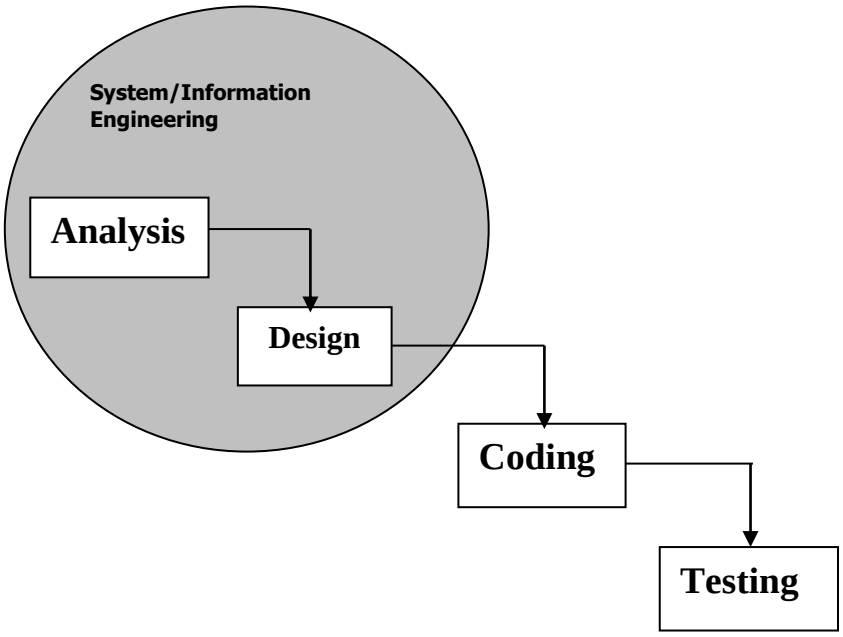


Figure 2: Linear Sequential Model

CHAPTER 7

Software Requirement Specification

7.1 Overview

The *software requirement specification (SRS)* is very important part of the software building process, which describes the actual user level requirement from technical point of view. i.e. what the user exactly wants ?or for what purpose we are making every thing The objective of preparing the software requirement specification is to represent the requirements of the software in such a manner that ultimately leads to successful software implementation. It is the result of the analysis process of the software development. It should contain all the data the software is going to process, the function it will provide, and the behavior it will exhibit. This Software Requirements Specifications (SRS) is defined in IEEE Std. 830-1993, IEEE Recommended Practice for Software Requirements Specifications. The document is organized in the following structure:

- Introduction
- Information Description
- Functional Description
- Behavior Description
- Validation Criteria
- Bibliography
- Appendix

7.1.1 Introduction

The introduction section describes the goals and objective the software under going development in context of computer based system. It mainly deals with the software scope. i.e. it will bind the software application domain.

7.1.2 Information Description

This section of the SRS provides a detailed of the problem that the software must solve. It should describe the core of the software – i.e. *The Data*. The data or information the software is going to work on is the most basic part of the software. The description of each data or information entity is described here. It also gives details of the relationships between the data elements of the software. The information description helps the software designers in their designing purpose.

7.1.3 Functional Description

This section of the SRS describes the each functions required to solve the problem. It emphasizes on the core of the software on which the data will be processed – i.e. *The Function*. It also includes the process specification of each function, design constraints, and performance characteristics. The DFD or any other graphical diagram can also be added to describe the functionality of the system.

7.1.4 Behavioral Description

This section of the SRS describes the behavior of the software will exhibit. It is based on definition of the events and the operations that it will perform because of events.

7.1.5 Validation Criteria

This section of the SRS contains the details of the tests that should be performed to validate functions, performance, and behavior of the software. It is one of the most important aspect of the software which decides how much robust our software is.

7.1.6 Bibliography

This section contains references to all the related documents related with the software. This may include any technical document, standards document or software engineering paper.

7.1.7 **Appendix**

This section is supplementary and can include the matters that are important for the software development. It may include the statistical data, graphs or algorithm details.

7.1.8 **SRS Document**

Software Requirements Specification Document

Author : Chetan

Affiliation : Alchemy software tchnologies

7.2.1 Introduction

This is a Software Requirement Specification for **" "EMAIL SERVICE AUTOMATION SYSTEM"** While this document will be in reasonable detail, it will be a good base for expansion of detail to be easily updated in the future.

7.2.2 Audience

EMAIL SERVICE AUTOMATION SYSTEM for all is apart of the web site so audiences are the person around the glob who uses the Internet to retrieve information from the site whether they are an independent end user or related to an institute whose data is customized to use with the institution assigned user id and password an updated copy of SRS is always available on the site.

7.2.3 Purpose

The purpose of this Software Requirement Specification document is to define the requirements, which will enable the development of a win application *" with the facility provided by the*

embedded approach in the software titled **EMAIL SERVICE AUTOMATION SYSTEM***"* using software engineering approach.

7.2.4 Scope

This document will form the basis for validating the final uploading of the site with the facility provided by the education for all. Any changes made to the requirements in future will have to go through a formal change approval process. The developer is responsible for asking for clarifications, where necessary, and will not make any alterations without the permission of the client.

7.2.5 Case Study

"EMAIL SERVICE AUTOMATION SYSTEM

Introduction:

Software Scope of Proposed software: the website get request from audience around the globe and act accordingly with that, login information, login dependent data transaction. Major task of the software is to build a robust flexible sustainable site, which fulfills all the criteria of an Ogr. With the facility of advertising package, match making and much more (the later are not the part of the project).

Information Description:

The most basic element of any software system is the information items that the system accepts, processes, and outputs. This is also called the information domain of the software. Our software consists of various types of entity sets, which are nothing but the set, the attributes (or information). The grouping of information items helps in logical mapping of the software. The following chart describes the different kinds of entities and the attributes that are the intrinsic part of the software.

Product sell and purchase-

This part of the site is still to be developed as a better option to get better items at low cost.

Behavioral Description

In our context, the proposed system should fully automate the education system; automation to improve the work efficiency by using Internet technologies will benefit the masses. As the proposed system is based on internet technologies (client server model) right to maintain the database and functionality which are strictly confidential within the control of administrator master and partially with the client master. Therefore, as per the need this is a unique and a new approach with the centralized database system in the field of online education. And needs more input and better improvement on every upload.

Validation Constraints

As our project is a database dependent for the information extraction and input so for such a project a minute error in the database will be drastic. So validation constraints has very important role here, because putting any wrong value in the database or in any fields on the form can give tremendous error. So, each field on the form level is validated before saving it into the database. We are and have maintained the validation checks using constraints to maintain the referential integrity at database level too.

CHAPTER 8

Analysis Modeling

8.1 Data Modeling

Data modeling is very important part of the analysis it mainly deals with the data object and their relationship. With the help of data model one can easily identify the primary objects to be process by the system. Data model makes Entity-Relationship (ER) diagram to describe the relationship between entities. The data model considers the data independent of the processing that transform the data.

8.1.1 Data Objects and Attributes

The data model consists the data objects and attributes that describe the data object. A data object is the representation of any composite information that is processed by computer software. A data object can be an external entity, a thing, an occurrence or event, a role, a structure etc. A data object encapsulates the data only-there is no reference within a data object to operations that act on the data. So the data object can be a table.

Attributes define the properties of a data object and take on one of three different characteristics.

1. Name an instance of the data object
2. Describe the instance
3. Make reference to another instance in another table.

Also one or more attribute can be defined as identifier i.e.- **Key**. The key is used to make the relationship with the attribute of another data object. It may or may not be unique. In our context, we can say that as our project is RDBMS, so here relationship between various data object is essential to get required information and result the user want. So key has very important role here to make references as well as referential integrity and to prevent from data redundancy.

8.1.2 Cardinality and Modality

Cardinality: Cardinality defines the maximum number of objects that can be participating in a relationship. According to Tillmann, *the cardinality is the specification of the number of occurrences of one object that can be related to the number of occurrences of another object*. Taking into consideration all combinations of 'one' and 'many' two objects can be related as:

1. One-to-One (1:1)
2. One-to-Many (1:M)
3. Many-to-One (M:1)
4. Many-to-Many (M:N)

Modality: Modality normally says that whether the occurrence of a relationship is necessary or not. The modality of a relationship is 0 if there is no explicit need for the relationship to occur or the relationship is optional and 1 if the occurrence of the relationship is mandatory.

8.1.3 Entity Relationship Diagram

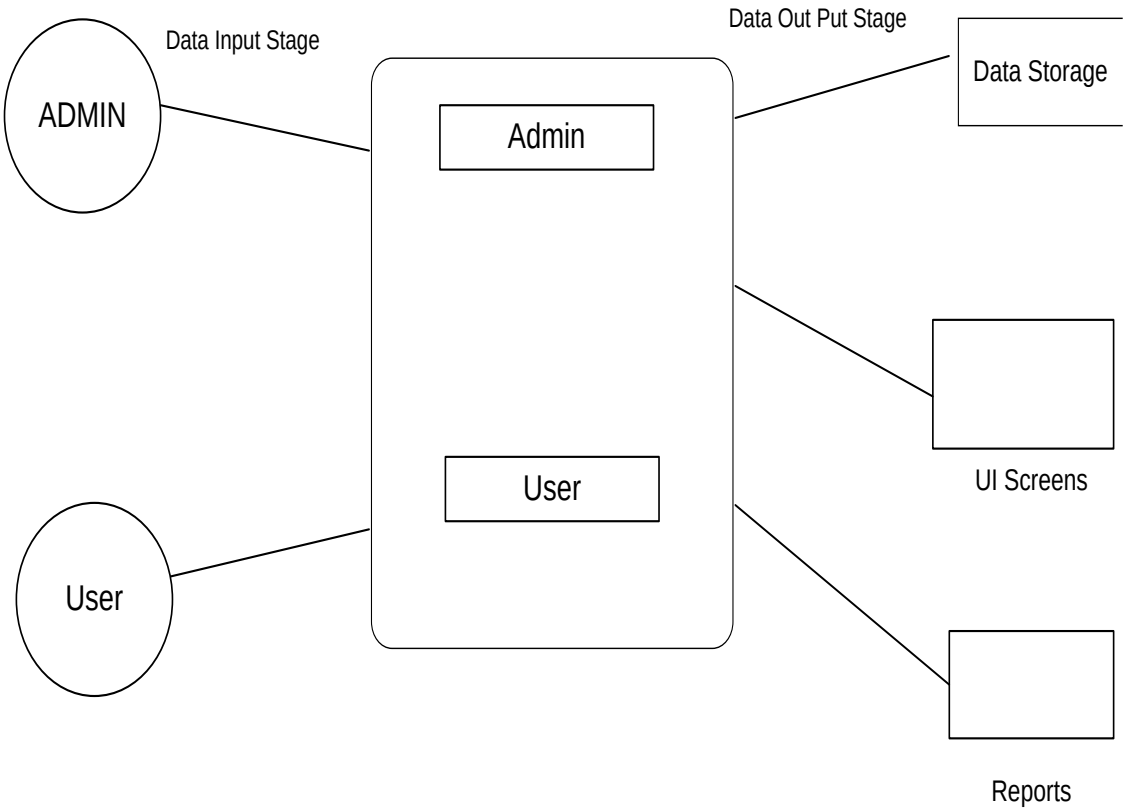
The Entity Relationship Diagrams (ERD) is the graphical notation of relationship between data object and attributes. The ERD was originally proposed by *Peter Chen* for the design of relational database systems and has been extended by others. Sets of primary components are identified for the ERD: data objects, attributes, relationship, and various type indicators. The primary purpose of the ERD is to represent data objects and their relationship.

The ER diagram for "**EMAIL SERVICE AUTOMATION SYSTEM**"

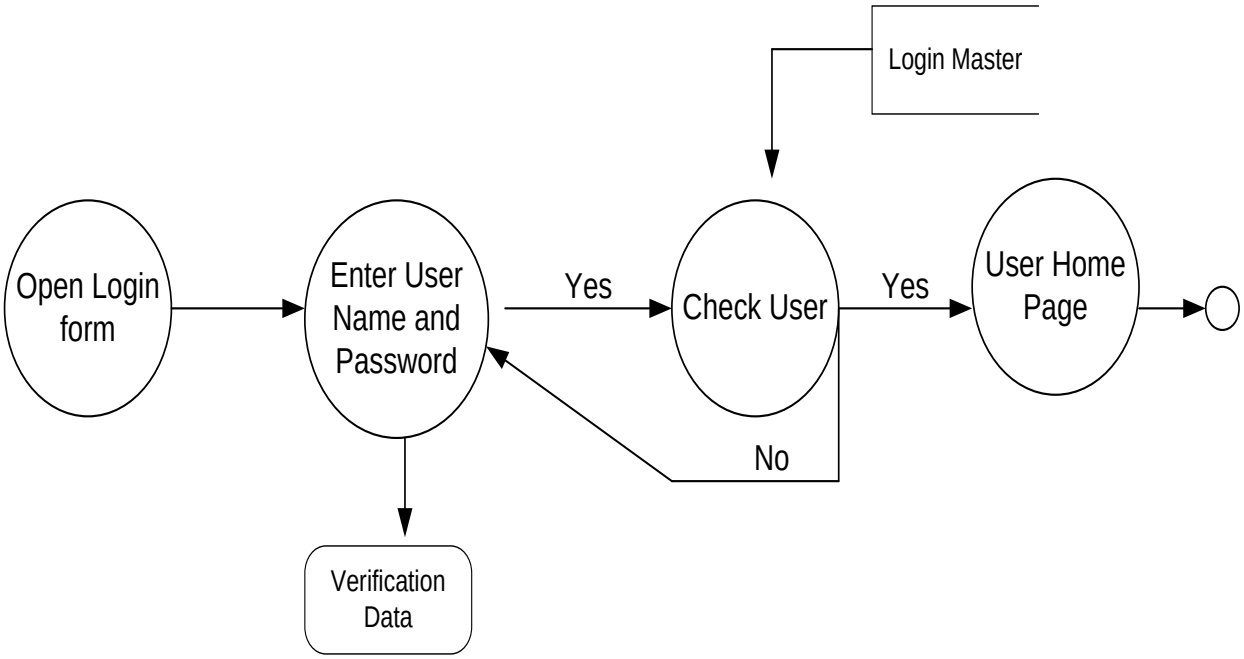
Analysis and Design

Mail Client DFD's

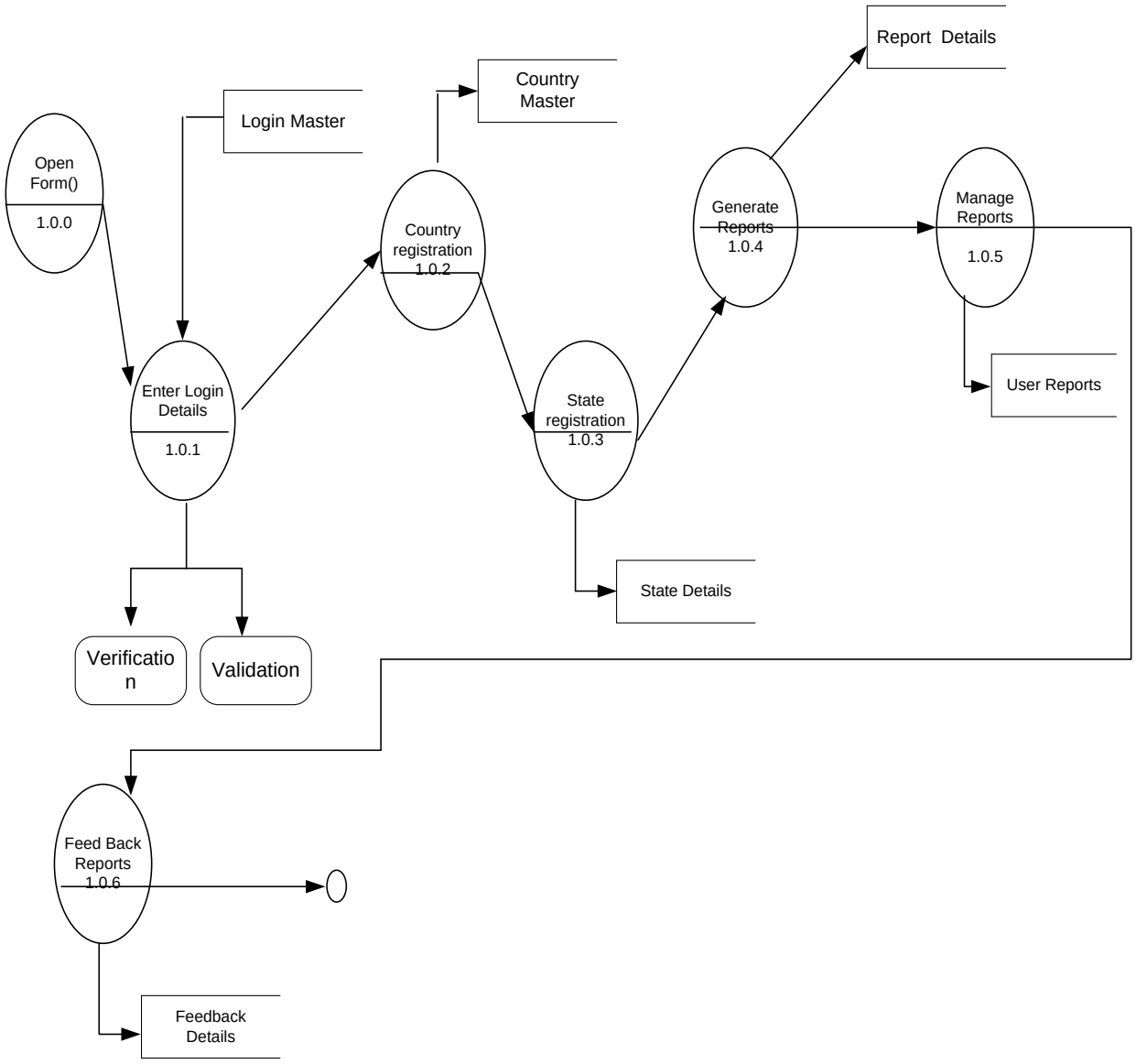
Context Level DFD



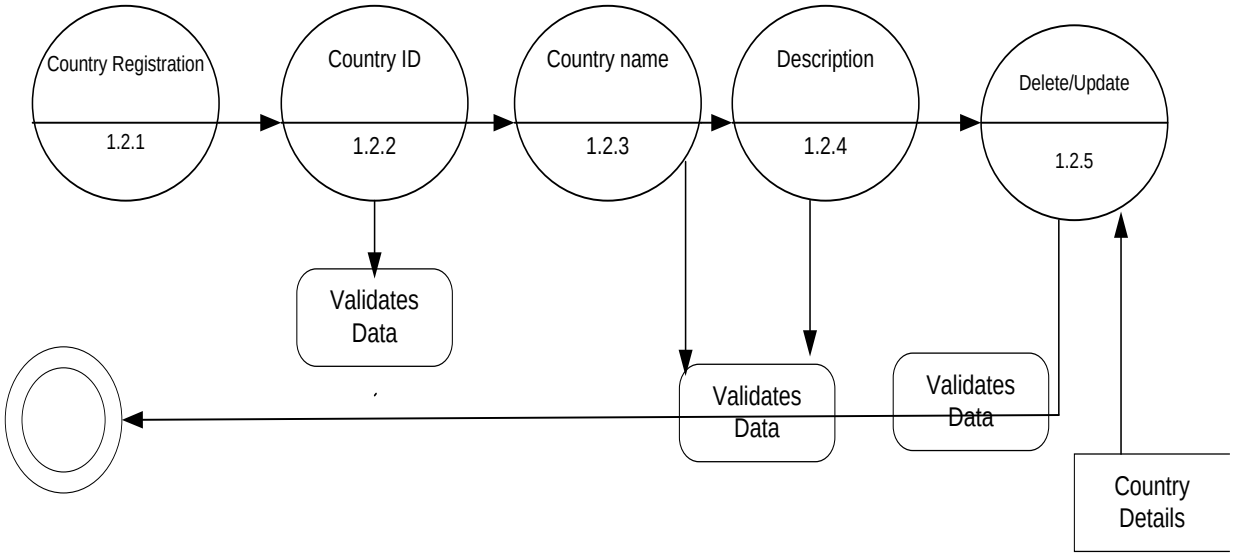
Login DFD


Login Master
Open Login form
Enter User Name and Password
Yes
Check User
Yes
User Home Page
No
Verification Data

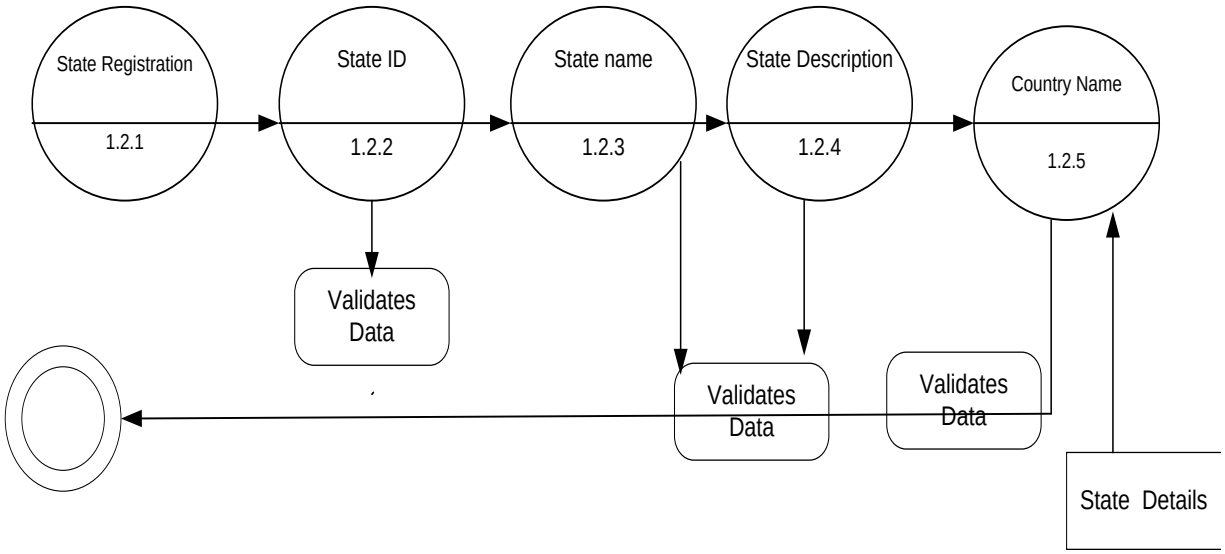
Admin Activities (1st Level)



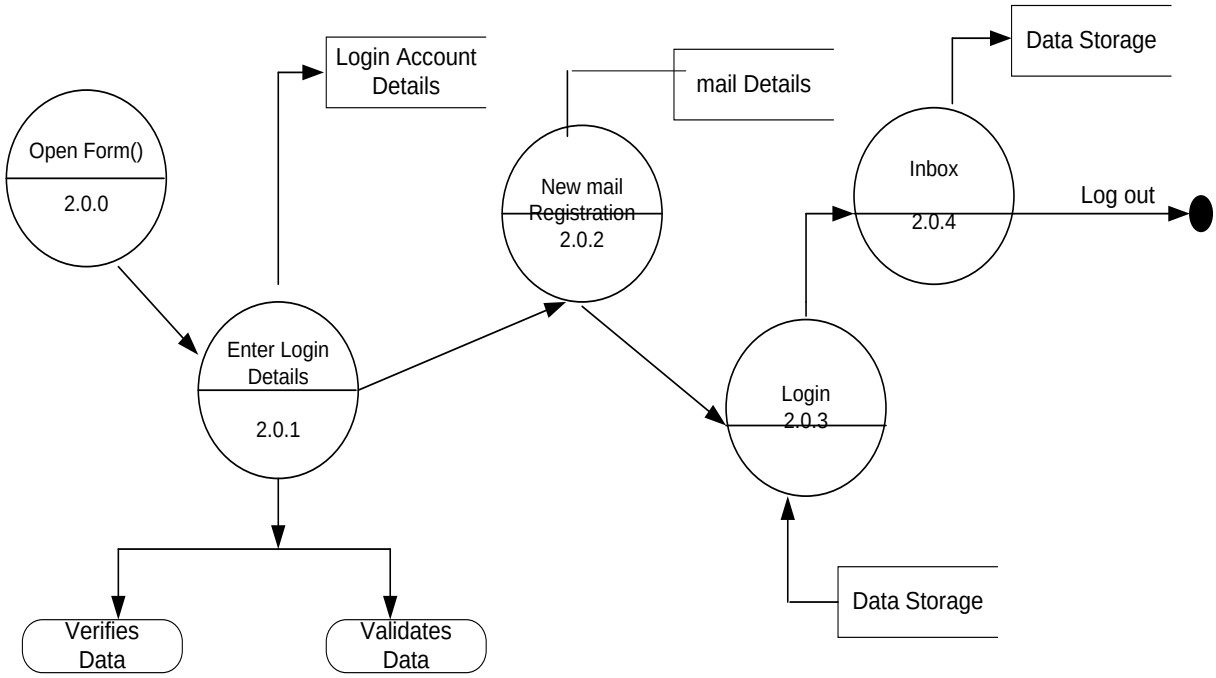
Admin Register Country



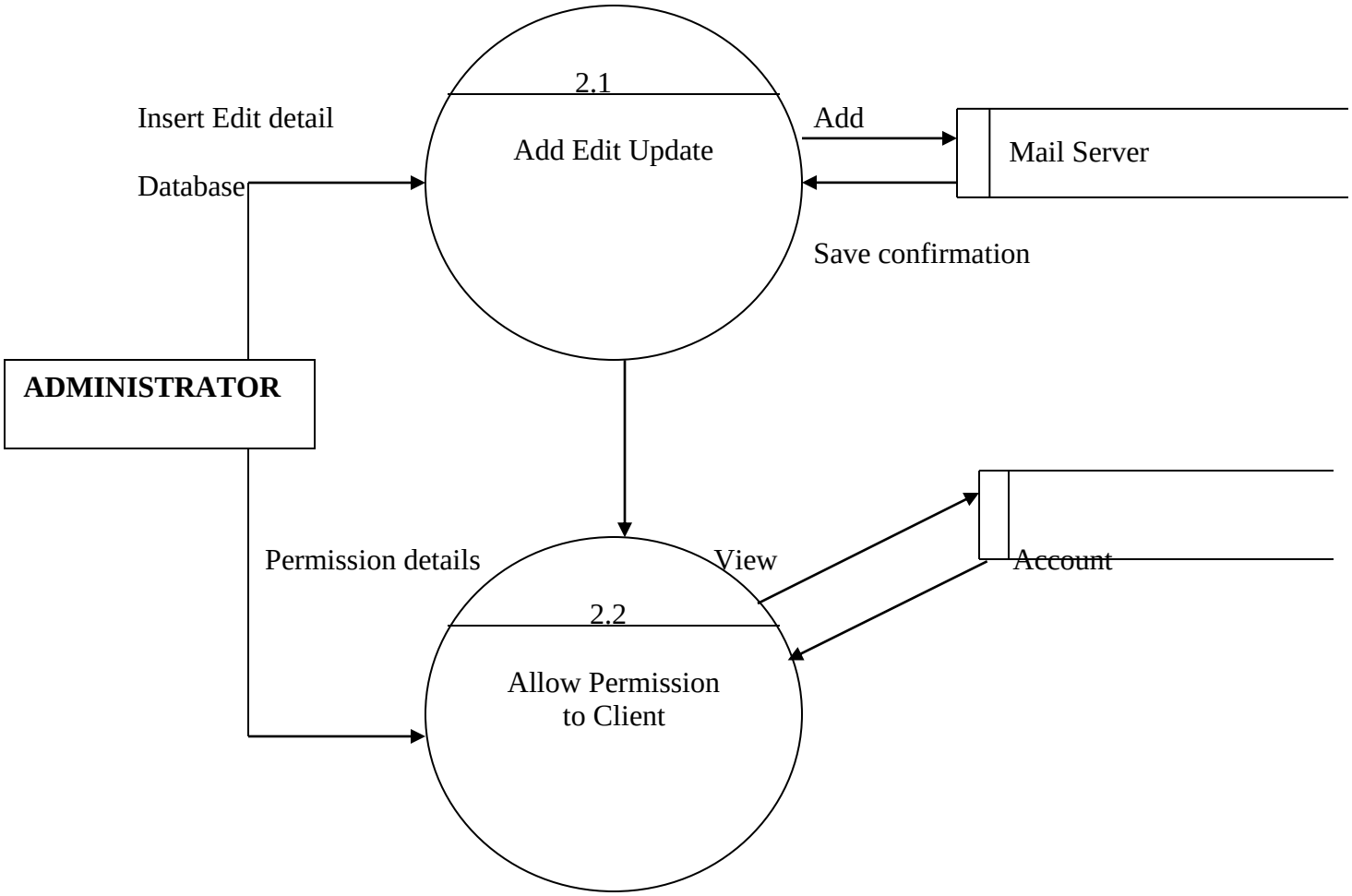
Admin Register State



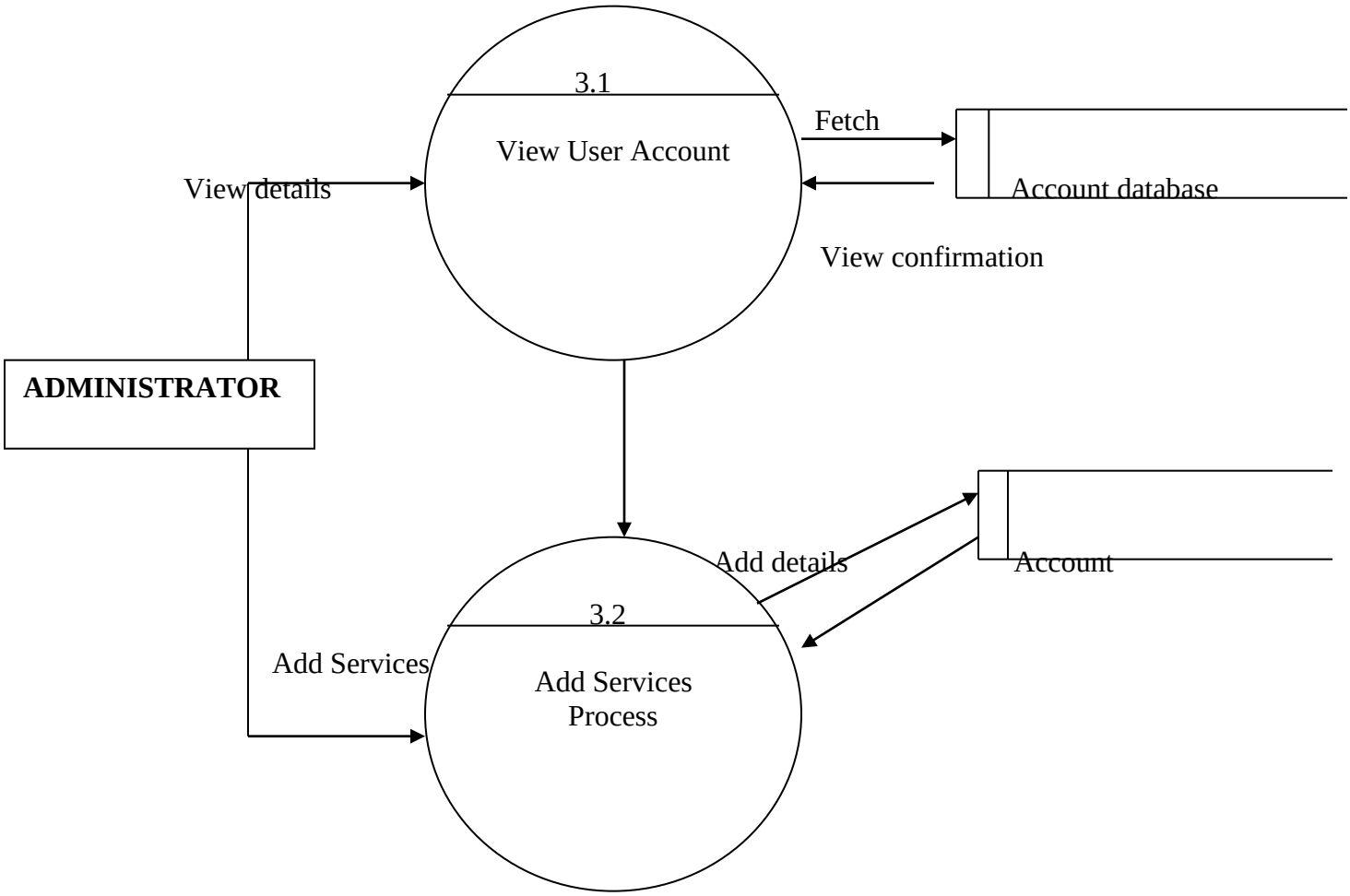
User (Employee) Activities (2nd Level):

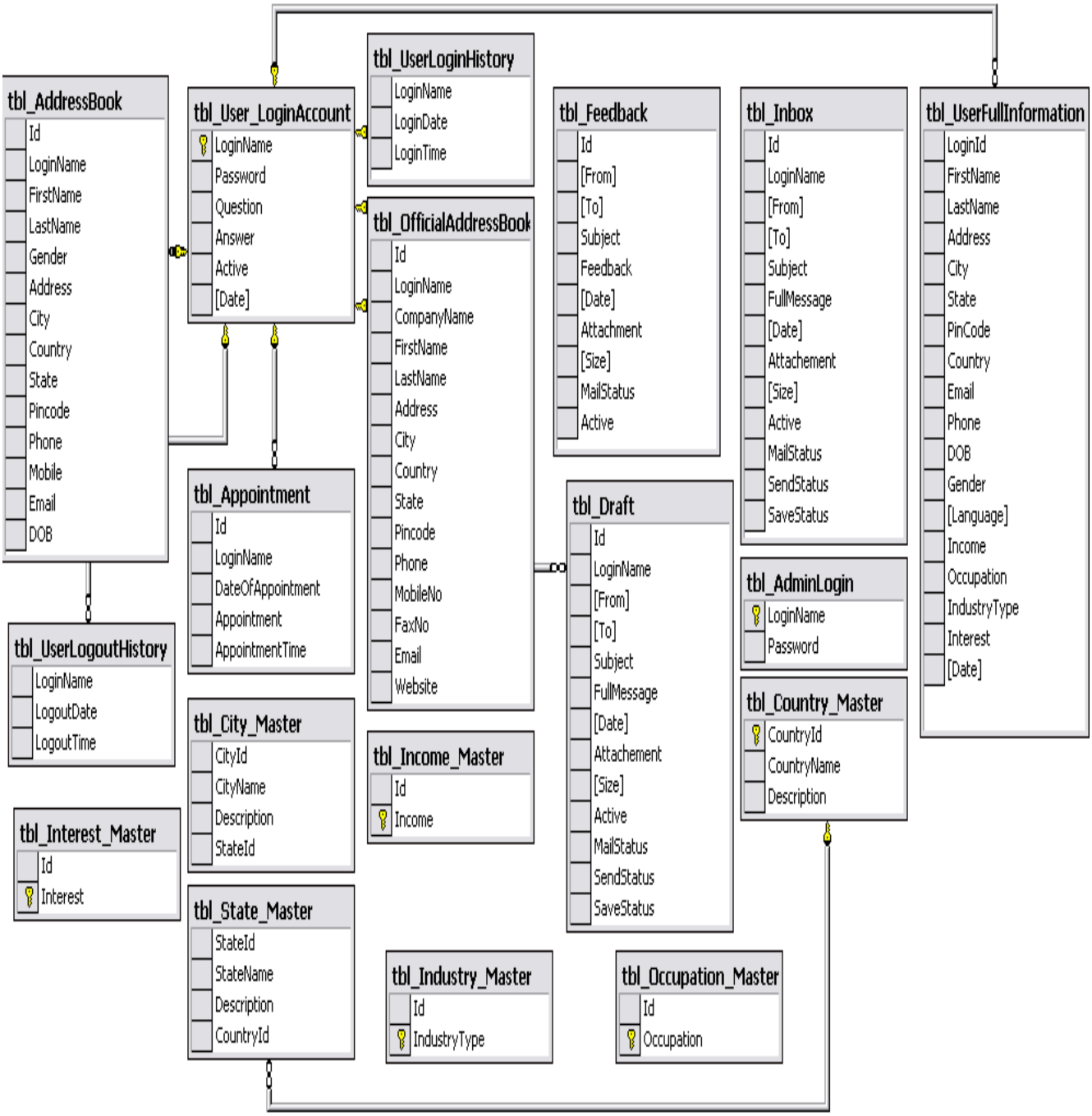


2nd level DFD



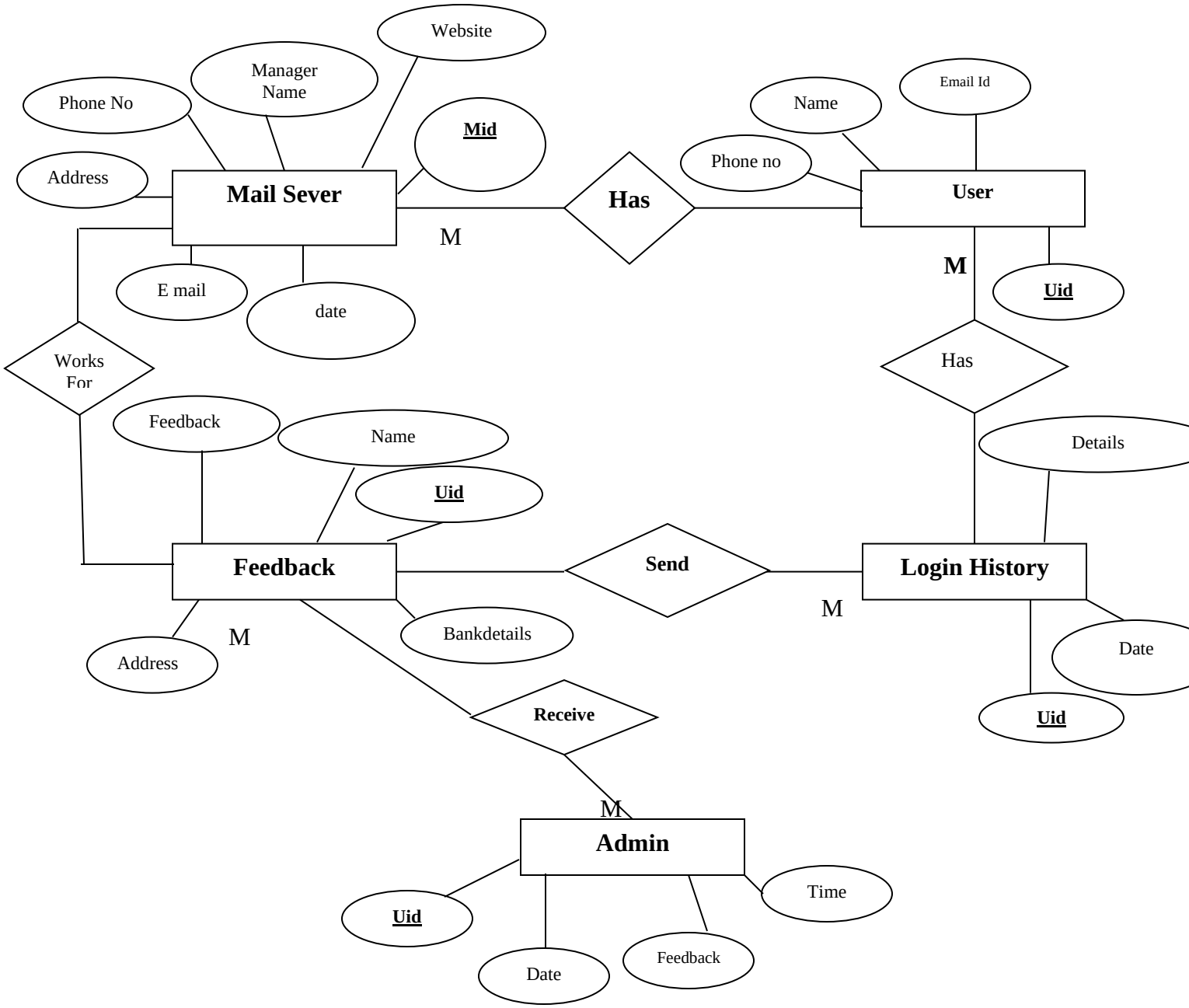
3rd level DFD



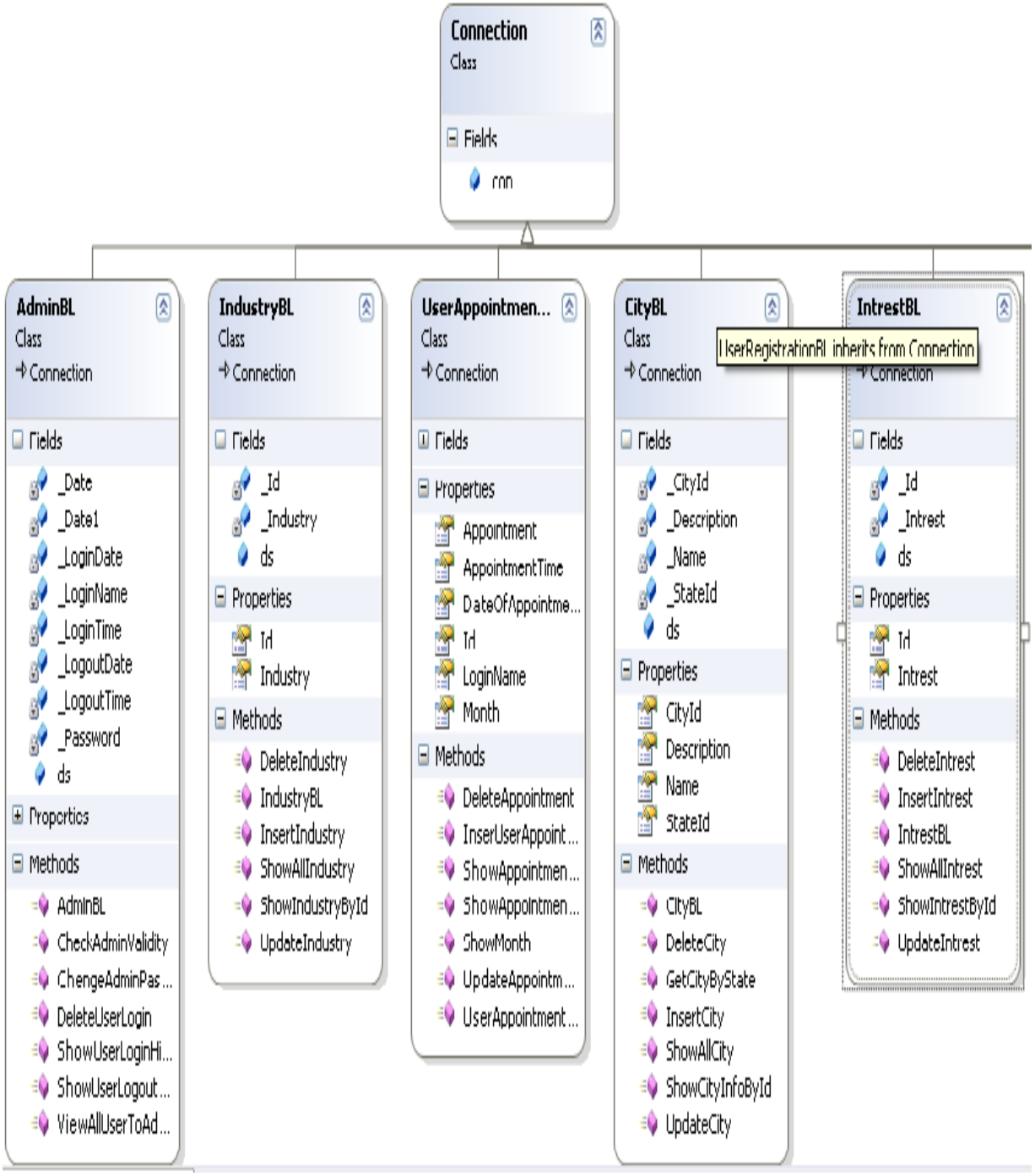

tbl_AddressBook
Id
LoginName
FirstName
LastName
Gender
Address
City
Country
State
Pincode
Phone
Mobile
Email
DOB
tbl_User_LoginAccount
LoginName
Password
Question
Answer
Active
[Date]
tbl_UserLoginHistory
LoginName
LoginDate
LoginTime
tbl_OfficialAddressBook
Id
LoginName
CompanyName
FirstName
LastName
Address
City
Country
State
Pincode
Phone
MobileNo
FaxNo
Email
Website
tbl_Feedback
Id
[From]
[To]
Subject
Feedback
[Date]
Attachment
[Size]
MailStatus
Active
tbl_Inbox
Id
LoginName
[From]
[To]
Subject
FullMessage
[Date]
Attachement
[Size]
Active
MailStatus
SendStatus
SaveStatus
tbl_UserFullInformation
LoginId
FirstName
LastName
Address
City
State
PinCode
Country
Email
Phone
DOB
Gender
[Language]
Income
Occupation
IndustryType
Interest
[Date]
tbl_Appointment
Id
LoginName
DateOfAppointment
Appointment
AppointmentTime
tbl_Draft
Id
LoginName
[From]
[To]
Subject
FullMessage
[Date]
Attachement
[Size]
Active
MailStatus
SendStatus
SaveStatus
tbl_AdminLogin
LoginName
Password
tbl_Country_Master
CountryId
CountryName
Description
tbl_UserLogoutHistory
LoginName
LogoutDate
LogoutTime
tbl_City_Master
CityId
CityName
Description
StateId
tbl_Income_Master
Id
Income
tbl_Interest_Master
Id
Interest
tbl_State_Master
StateId
StateName
Description
CountryId
tbl_Industry_Master
Id
IndustryType
tbl_Occupation_Master
Id
Occupation

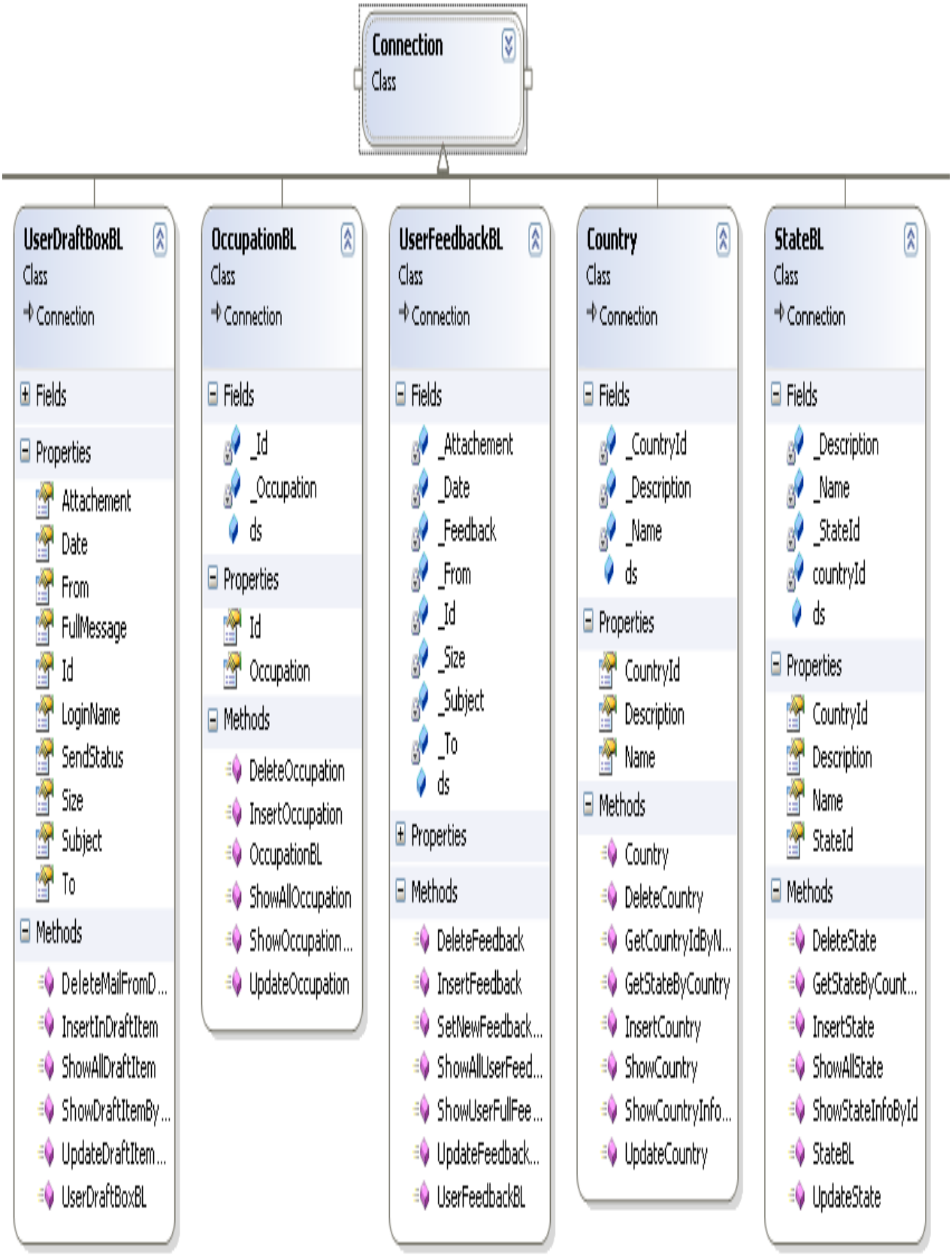
E-R DIAGRAM

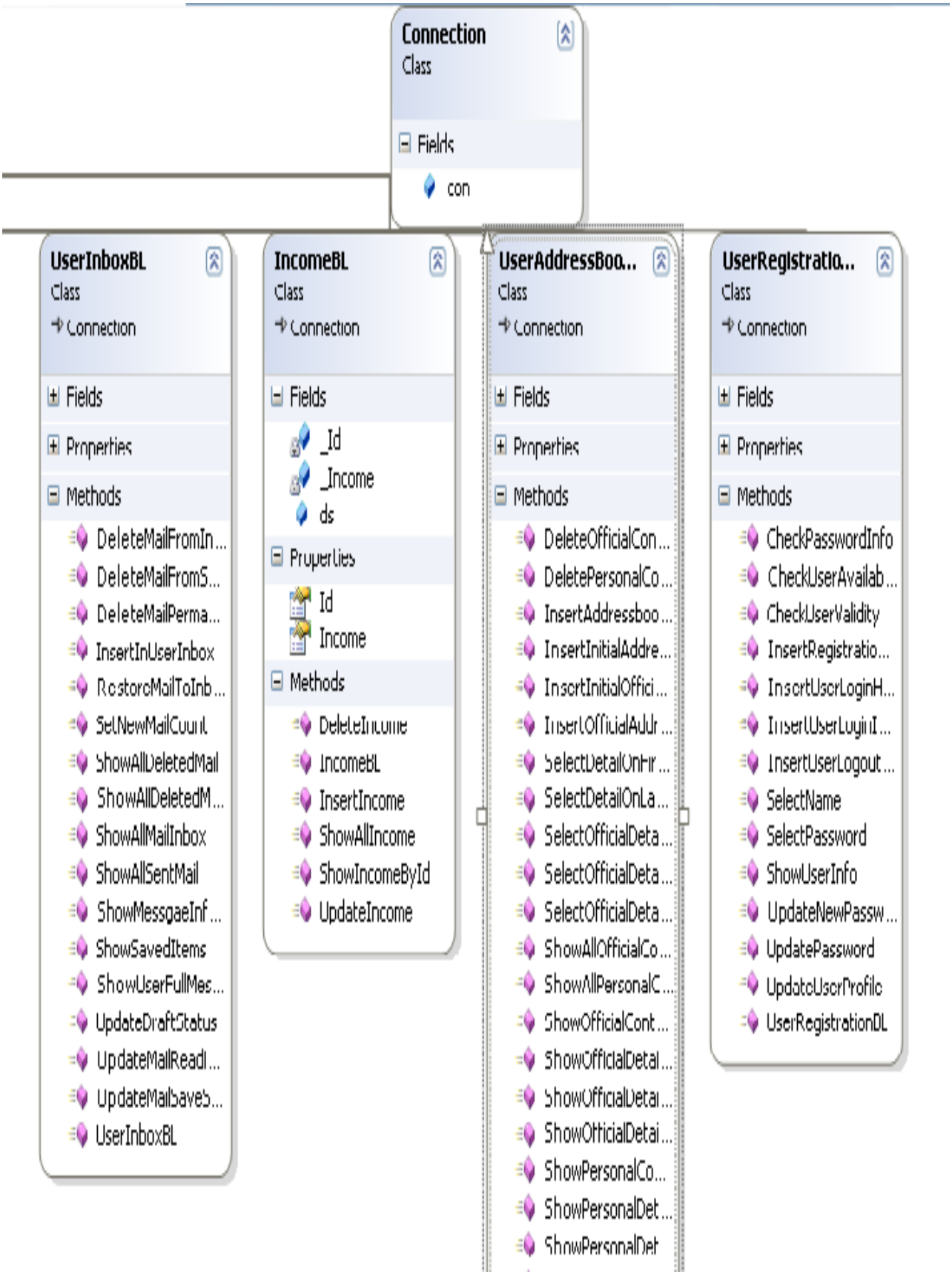
R- R Diagram

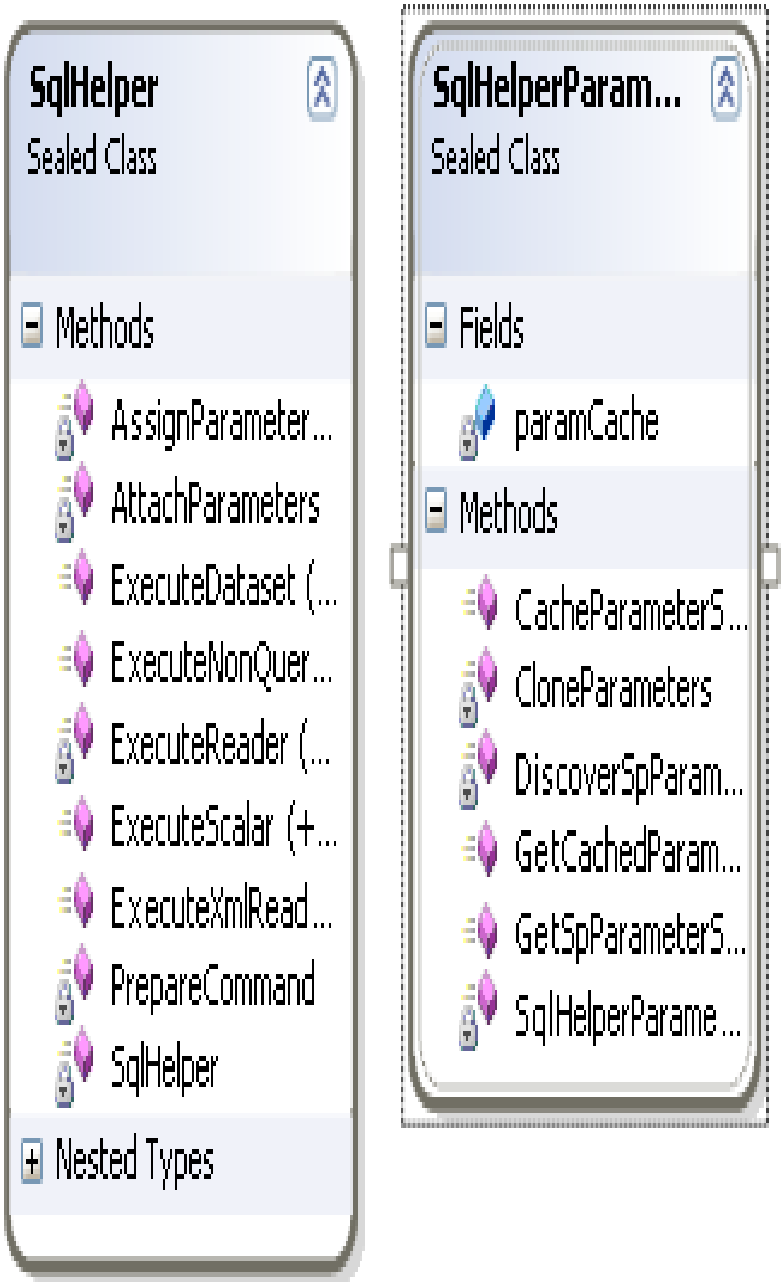

Website
Manager Name
Phone No
Mid
Address
Mail Sever
Has
Name
Email Id
Phone no
User
M
M
E mail
date
Uid
Works For
Has
Feedback
Name
Details
Uid
Feedback
Send
Login History
M
Address
M
Bankdetails
Date
Receive
Uid
M
Admin
Time
Uid
Date
Feedback

CLASS DIAGRAM




Connection
Class
UserDraftBoxBL
Class
Connection
Fields
Properties
Attachement
Date
From
FullMessage
Id
LoginName
SendStatus
Size
Subject
To
Methods
DeleteMailFromD...
InsertInDraftItem
ShowAllDraftItem
ShowDraftItemBy...
UpdateDraftItem...
UserDraftBoxBL
OccupationBL
Class
Connection
Fields
_Id
_Occupation
ds
Properties
Id
Occupation
Methods
DeleteOccupation
InsertOccupation
OccupationBL
ShowAllOccupation
ShowOccupation...
UpdateOccupation
UserFeedbackBL
Class
Connection
Fields
_Attachement
_Date
_Feedback
_From
_Id
_Size
_Subject
_To
ds
Properties
Methods
DeleteFeedback
InsertFeedback
SetNewFeedback...
ShowAllUserFeed...
ShowUserFullFee...
UpdateFeedback...
UserFeedbackBL
Country
Class
Connection
Fields
_CountryId
_Description
_Name
ds
Properties
CountryId
Description
Name
Methods
Country
DeleteCountry
GetCountryIdByN...
GetStateByCountry
InsertCountry
ShowCountry
ShowCountryInfo...
UpdateCountry
StateBL
Class
Connection
Fields
_Description
_Name
_StateId
countryId
ds
Properties
CountryId
Description
Name
StateId
Methods
DeleteState
GetStateByCount...
InsertState
ShowAllState
ShowStateInfoById
StateBL
UpdateState


Connection
Class
Fields
con
UserInboxBL
Class
Connection
Fields
Properties
Methods
DeleteMailFromIn...
DeleteMailFromS...
DeleteMailPerma...
InsertInUserInbox
RestoreMailToInb...
SetNewMailCount
ShowAllDeletedMail
ShowAllDeletedM...
ShowAllMailInbox
ShowAllSentMail
ShowMessgaeInf...
ShowSavedItems
ShowUserFullMes...
UpdateDraftStatus
UpdateMailReadI...
UpdateMailSaveS...
UserInboxBL
IncomeBL
Class
Connection
Fields
_Id
_Income
ds
Properties
Id
Income
Methods
DeleteIncome
IncomeBL
InsertIncome
ShowAllIncome
ShowIncomeById
UpdateIncome
UserAddressBoo...
Class
Connection
Fields
Properties
Methods
DeleteOfficialCon...
DeletePersonalCo...
InsertAddressboo...
InsertInitialAddre...
InsertInitialOffici...
InsertOfficialAdd...
SelectDetailOnHr...
SelectDetailOnLa...
SelectOfficialDeta...
SelectOfficialDeta...
SelectOfficialDeta...
ShowAllOfficialCo...
ShowAllPersonalC...
ShowOfficialCont...
ShowOfficialDetai...
ShowOfficialDetai...
ShowOfficialDetai...
ShowPersonalCo...
ShowPersonalDet...
ShowPersonalDet
UserRegistratio...
Class
Connection
Fields
Properties
Methods
CheckPasswordInfo
CheckUserAvailab...
CheckUserValidity
InsertRegistratio...
InsertUserLoginH...
InsertUserLoginI...
InsertUserLogout...
SelectName
SelectPassword
ShowUserInfo
UpdateNewPassw...
UpdatePassword
UpdateUserProfile
UserRegistrationDL


SqlHelper
Sealed Class
Methods
AssignParameter...
AttachParameters
ExecuteDataset (...
ExecuteNonQuer...
ExecuteReader (...
ExecuteScalar (+...
ExecuteXmlRead...
PrepareCommand
SqlHelper
Nested Types
SqlHelperParam...
Sealed Class
Fields
paramCache
Methods
CacheParameterS...
CloneParameters
DiscoverSpParam...
GetCachedParam...
GetSpParameterS...
SqlHelperParame...

CHAPTER 9

Software Design

9.1 Introduction

Design is the meaningful engineering representation of something that is to be built. In the software engineering context, design focuses on four major area of concern: data, architecture, interfaces and component. Once the software requirements have been analyzed and specified, software design is the first of three technical activities—design, code generation, and test—are required to build the and verify the software. After making Software Requirement Specification of the education for all, now we are in position to design the software. As discussed earlier data design is the backbone of any RDBMS product. So firstly, we are emphasizing on database design. The data design transforms the information domain model created during analysis into data structures that will be required to implement the software.

9.2 Database Design

As we have discussed earlier about the under developing system, which is based on .internet techonologies So, our database will be centralized database which will run on sql server .

9.2.1 Data Structure

Data structure is a representation of the logical relationship among individual elements of data. Because the structure of information will invariably affect the final procedural design, data structure is as important as program structure of the representation of the software architecture. Data structure dictates the organization, method of access, degree of associativity and processing alternatives of the information.

9.2.2 Functional Dependencies

The Functional dependencies (FD) are the consequence of the interrelationship among attributes of an entity

represented by a relation or due to the relationship between entities that is also represented by a relation.

9.2.3 Normalization

Normalization is a technique used in RDBMS to decompose into table into two or more different tables to remove functional dependency as well as redundancy. The normalization has different form namely—First NF, Second NF, Third NF, BCNF and Fourth NF. The decomposition of the table should be in the manner that data should not be lost i.e. the union of the decomposed table should must give the actual table. Normalization is also necessary to maintain referential integrity. In our context, we used normalization technique up to a three NF as our project had much number of FDs and case of data redundancy.

9.2.4 Table Description

To keep record without any redundancy each entity is represented by an independent table in the database. A list of tables is depicted below:

A complete structure which includes

Number of Modules and their description

Number of Modules

The system after careful analysis has been identified to be presented with the following modules:

The modules involved are:

1. **Member registration Module:**
   - Member can register and sign in here. For registration, member has to provide personal details, address details, employment details, account details and they have to agree with policies.
   - Member can sign in by providing their account details (Username and password).
2. **Sending and Receiving mails:**
   - By making use of this module, Members can send/receive/view mails. Many features have been provided to members so that they can 1) manage (view/ edit/ delete) their mails, 2) forward mails, 3) send attachments, 4) send group mail, manage mails in folders etc.
3. **Integrated Security Module:**
   - This module is made is provide security features to the application.
4. **Admin Module:**
   - Admin is a super user and hence responsible for a) Site Maintenance, b) Members Management, c) Mails management and d) Generate various reports.
5. **Login/Logout Date & Time Tracking Module:** Admin can view the Login/Logout time of User. Whenever the user login/logout then Current Date & Time will be stored to view for Admin.
6. **Address book Maintenance:** Here user can maintain the address book for own friend with all Address Contact Info, Birthday, and Marriage Anniversary etc.
7. **CMS (content Management System) Integration:** Using CMS tool we can customize the mail the message with all formatting features.

Data Structure of Each Module

Address Book

Column Name	Data Type	Length	Allow Nulls
Id	int	4	
LoginName	varchar	50	
FirstName	varchar	50	✓
LastName	varchar	50	✓
Gender	varchar	20	✓
Address	varchar	50	✓
City	varchar	50	✓
Country	varchar	50	✓
State	varchar	50	✓
Pincode	varchar	20	✓
Phone	varchar	20	✓
Mobile	varchar	20	✓
Email	varchar	50	✓
DOB	datetime	8	✓

Admin Login

Column Name	Data Type	Length	Allow Nulls
LoginName	varchar	50	
Password	varchar	50	✓

Appointment

Column Name	Data Type	Length	Allow Nulls
Id	int	4	
LoginName	varchar	50	✓
DateOfAppointment	datetime	8	✓
Appointment	varchar	100	✓
AppointmentTime	varchar	30	✓

City Master

Column Name	Data Type	Length	Allow Nulls
CityId	int	4	
CityName	varchar	50	✓
Description	varchar	80	✓
StateId	int	4	✓

Country Master

Column Name	Data Type	Length	Allow Nulls
CountryId	int	4	
CountryName	varchar	50	✓
Description	varchar	80	✓

Draft

Column Name	Data Type	Length	Allow Nulls
Id	int	4	
LoginName	varchar	50	✓
[From]	varchar	100	✓
[To]	varchar	100	✓
Subject	varchar	100	✓
FullMessage	varchar	500	✓
[Date]	datetime	8	✓
Attachement	varchar	50	✓
[Size]	varchar	20	✓
Active	tinyint	1	✓
MailStatus	char	10	✓
SendStatus	char	15	✓
SaveStatus	char	20	✓

Feedback

Column Name	Data Type	Length	Allow Nulls
Id	int	4	
[From]	varchar	50	✓
[To]	varchar	50	✓
Subject	varchar	50	✓
Feedback	varchar	200	✓
[Date]	datetime	8	✓
Attachment	varchar	50	✓
[Size]	varchar	20	✓
MailStatus	char	10	✓
Active	tinyint	1	✓

Inbox

Column Name	Data Type	Length	Allow Nulls
Id	int	4	
LoginName	varchar	50	✓
[From]	varchar	100	✓
[To]	varchar	100	✓
Subject	varchar	100	✓
FullMessage	varchar	500	✓
[Date]	datetime	8	✓
Attachement	varchar	50	✓
[Size]	varchar	20	✓
Active	tinyint	1	✓
MailStatus	char	10	✓
SendStatus	char	10	✓
SaveStatus	char	20	✓

Income

Column Name	Data Type	Length	Allow Nulls
Id	int	4	
Income	varchar	50	

Industry

Column Name	Data Type	Length	Allow Nulls
Id	int	4	
IndustryType	varchar	80	

Interest

Column Name	Data Type	Length	Allow Nulls
Id	int	4	
Interest	varchar	80	

Occupation Master

Column Name	Data Type	Length	Allow Nulls
Id	int	4	
Occupation	varchar	80	

Official Address Book

Column Name	Data Type	Length	Allow Nulls
Id	int	4	
LoginName	varchar	50	
CompanyName	varchar	50	✓
FirstName	varchar	50	✓
LastName	varchar	50	✓
Address	varchar	50	✓
City	varchar	50	✓
Country	varchar	50	✓
State	varchar	50	✓
Pincode	varchar	20	✓
Phone	varchar	20	✓
MobileNo	varchar	20	✓
FaxNo	varchar	20	✓
Email	varchar	50	✓
Website	varchar	50	✓

State Master

Column Name	Data Type	Length	Allow Nulls
StateId	int	4	
StateName	varchar	50	✓
Description	varchar	80	✓
CountryId	int	4	✓

Login Account

Column Name	Data Type	Length	Allow Nulls
LoginName	varchar	50	
Password	varchar	50	✓
Question	varchar	100	✓
Answer	varchar	80	✓
Active	tinyint	1	✓
[Date]	datetime	8	✓

User Full Information

Column Name	Data Type	Length	Allow Nulls
LoginId	varchar	50	✓
FirstName	varchar	50	✓
LastName	varchar	50	✓
Address	varchar	100	✓
City	varchar	50	✓
State	varchar	50	✓
PinCode	varchar	10	✓
Country	varchar	50	✓
Email	varchar	50	✓
Phone	varchar	20	✓
DOB	varchar	50	✓
Gender	varchar	10	✓
[Language]	varchar	50	✓
Income	varchar	50	✓
Occupation	varchar	50	✓
IndustryType	varchar	100	✓
Interest	varchar	300	✓
[Date]	datetime	8	✓

User Login History

Column Name	Data Type	Length	Allow Nulls
LoginName	varchar	50	✓
LoginDate	datetime	8	✓
LoginTime	varchar	20	✓

User Logout History

	Column Name	Data Type	Length	Allow Nulls
▶	LoginName	varchar	50	✓
	LogoutDate	datetime	8	✓
	LogoutTime	varchar	20	✓

Process Logic of each Module

Login

This is a very first module in my project:

1. Enter username and password.
2. If username and password will exists in the table.
3. The software will open.
4. Else you got an error message.

New User

This is new user module:

1. Enter the username and password.
2. Save the records
3. Data will save in the login table permanently.
4. Second time you can enter with your own username and password.

Member Registration details:

Registration module is responsible for member registration and login. While registration, member will be prompted for his 1) login account details (username, password, hint question, answer), 2) his personal details, and 3) his contact address

At time of sign in, Member has to provide username and password.

In Message compose box, Member has to provide Message to send with Email-ID (to whom message has to be sent.).

Outputs:

- On successful registration, member will be provided confirmation mail.
- On successful signing in, member will be placed to My Account page.

The following commands specify access control identifiers and they are typically used to authorize and authenticate the user (command codes are shown in parentheses)

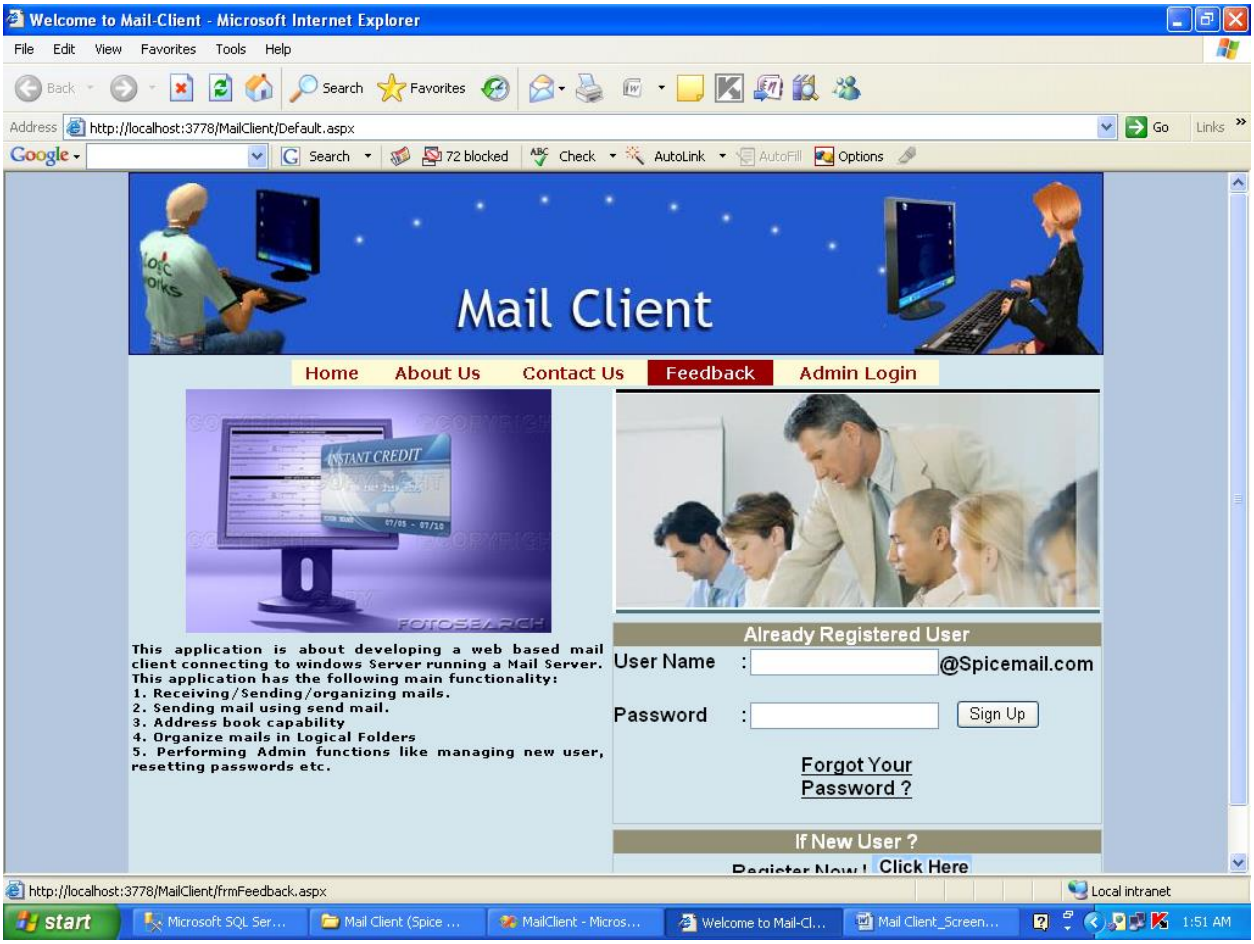
USER NAME (USER)

The user identification is that which is required by the server for access to its file system. This command will normally be the first command transmitted by the user after the control connections are made (some servers may require this).

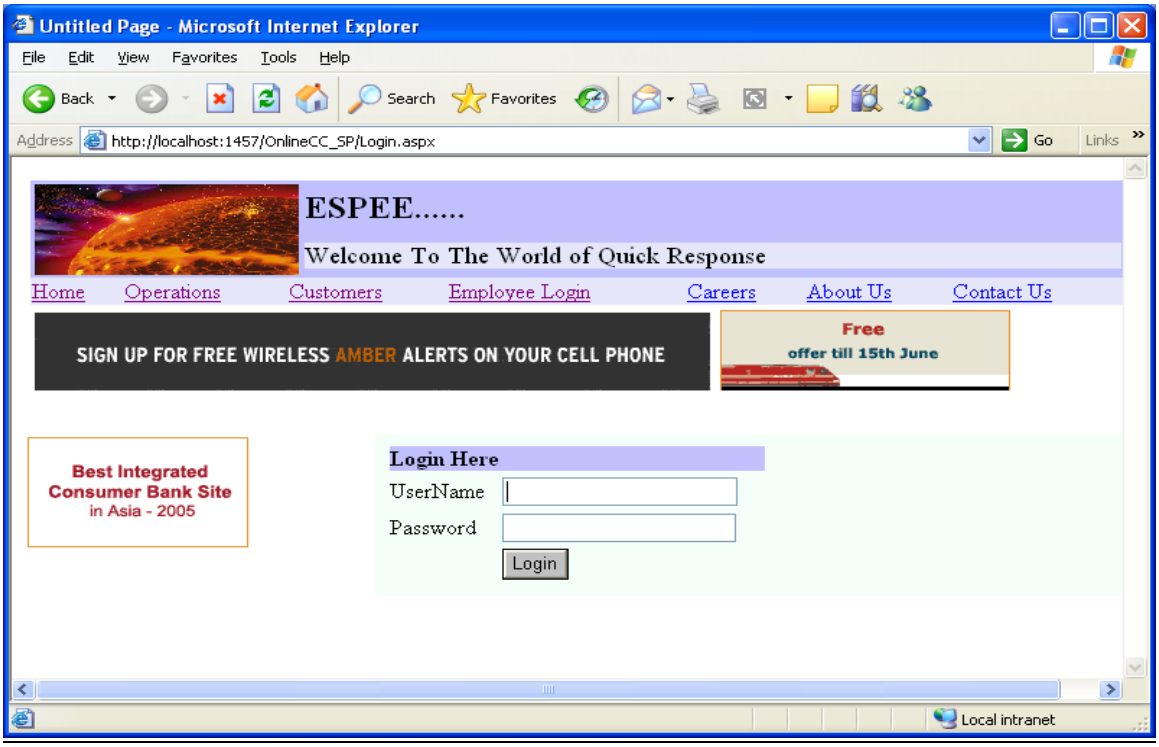
PASSWORD (PASS)

This command must be immediately preceded by the user name command, and, for some sites, completes the user's identification for access control. Since password information is quite sensitive, it is desirable in general to "mask" it or suppress type out.

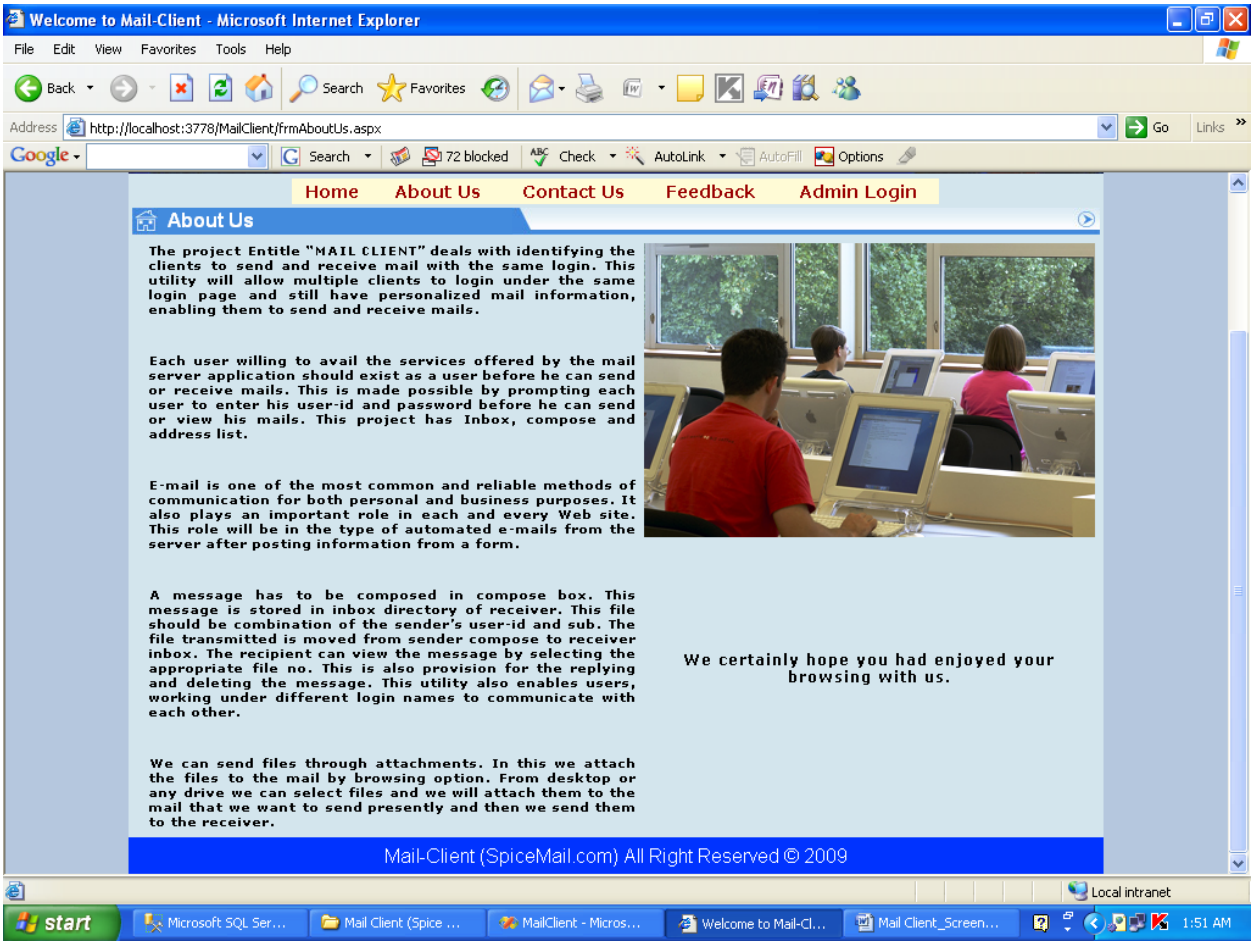
OUTPUTS OF THE PROJECT



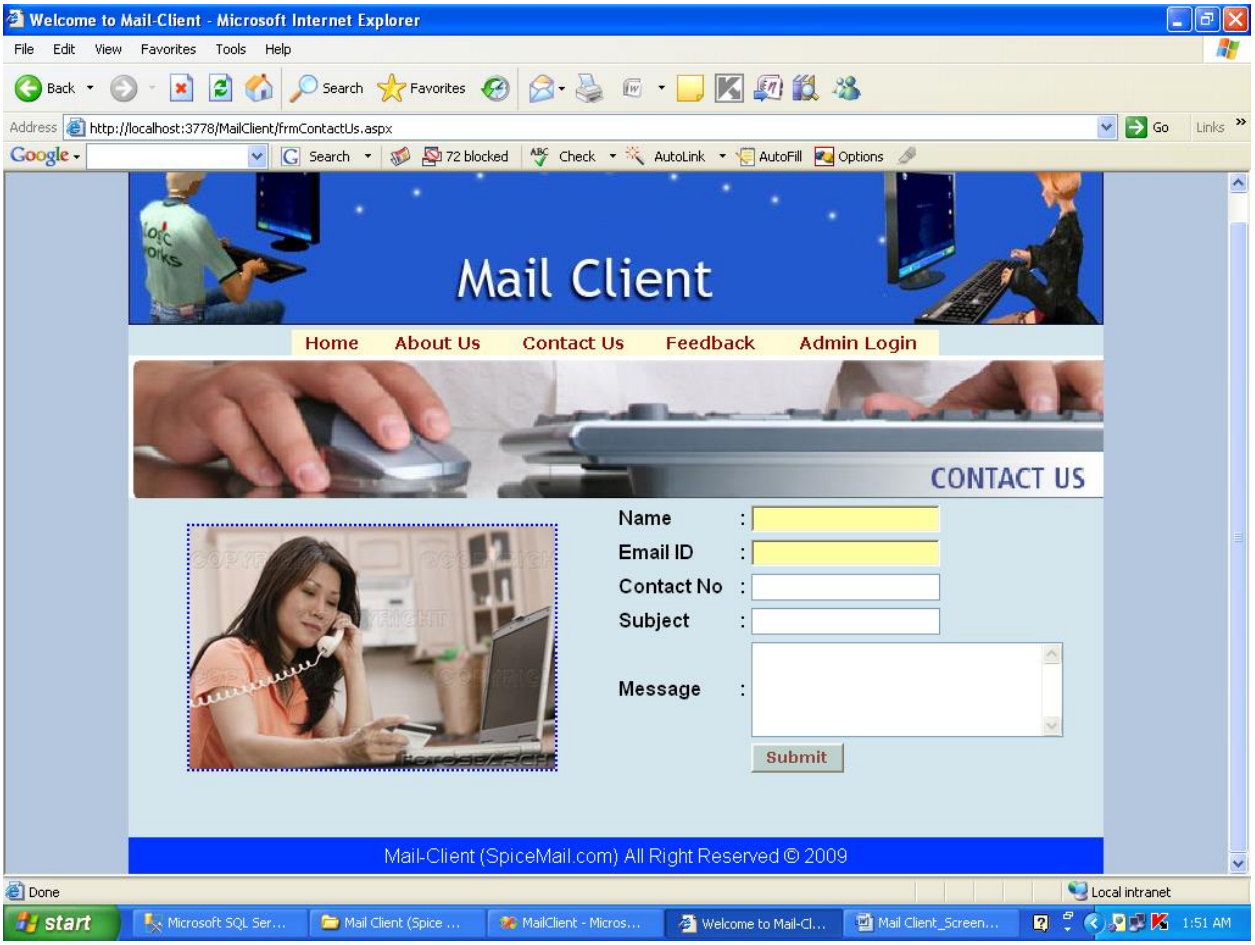
Pagename: Home page
Fields: User name, password,forget your password,sign up
Process: user have to enter login details
Table: tbl_UserLoginHistory
Stored procedure: yes
Remarks: if new user have to sign up
Error: no



Pagename: emp login
Fields: user name,password,login
Process: enter login details
Table: login
Stored procedure: sp_userlogin
Remarks: login
Error: no



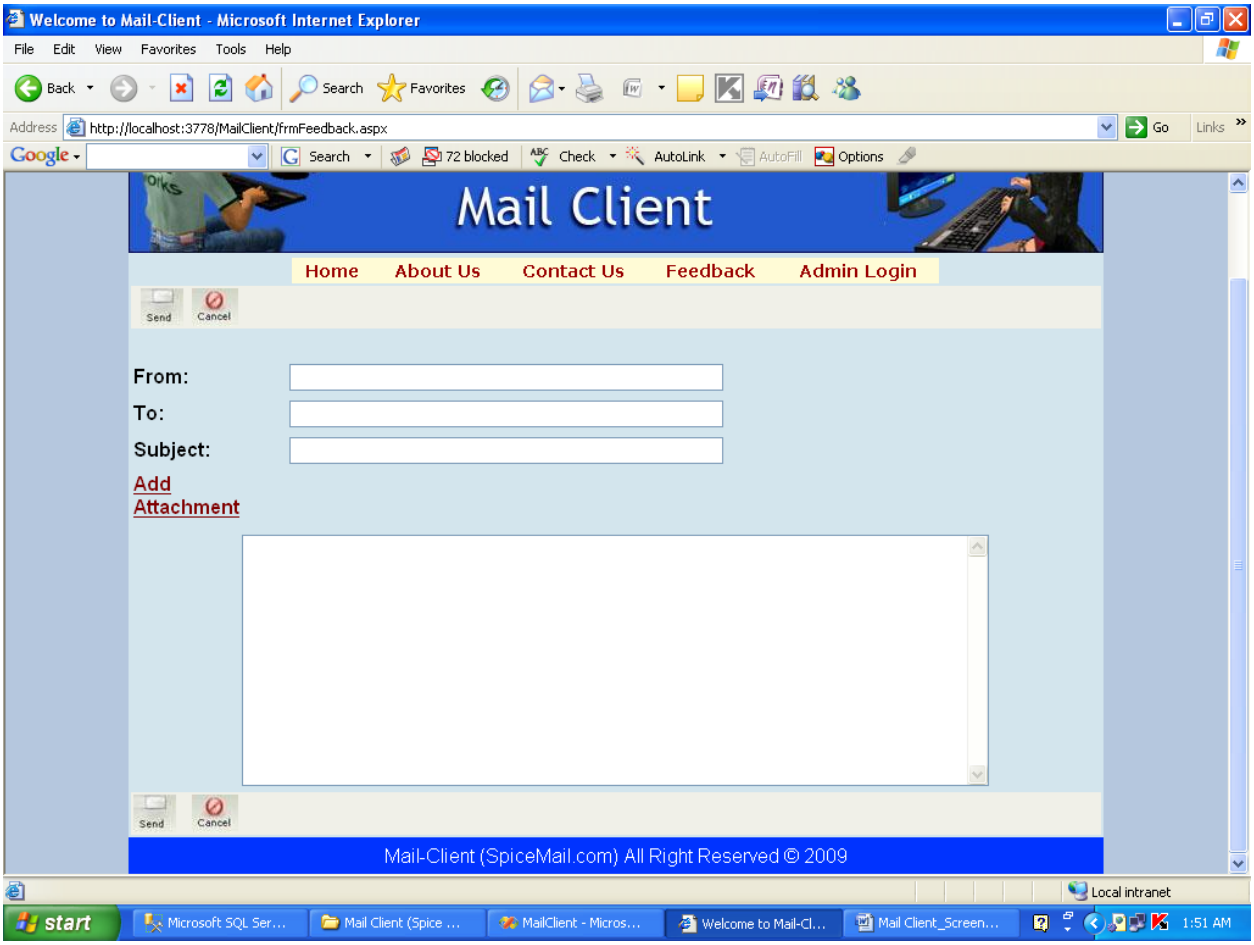
Pagename: About Us
Fields: no
Process: static page
Table: no
Stored procedure: no
Remarks:statis



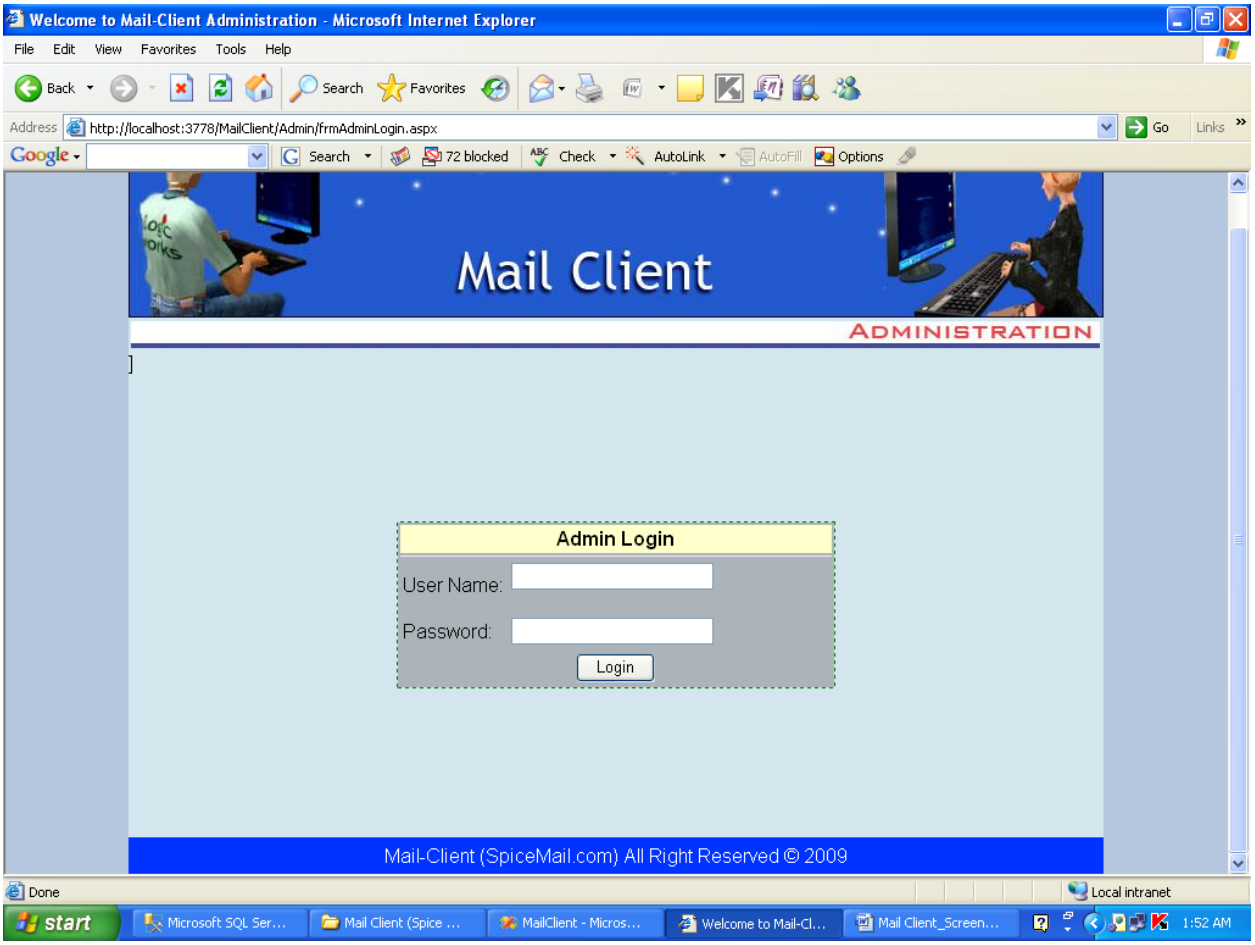
Pagename: contact us

Fields: name, email id,contact no,subject,message

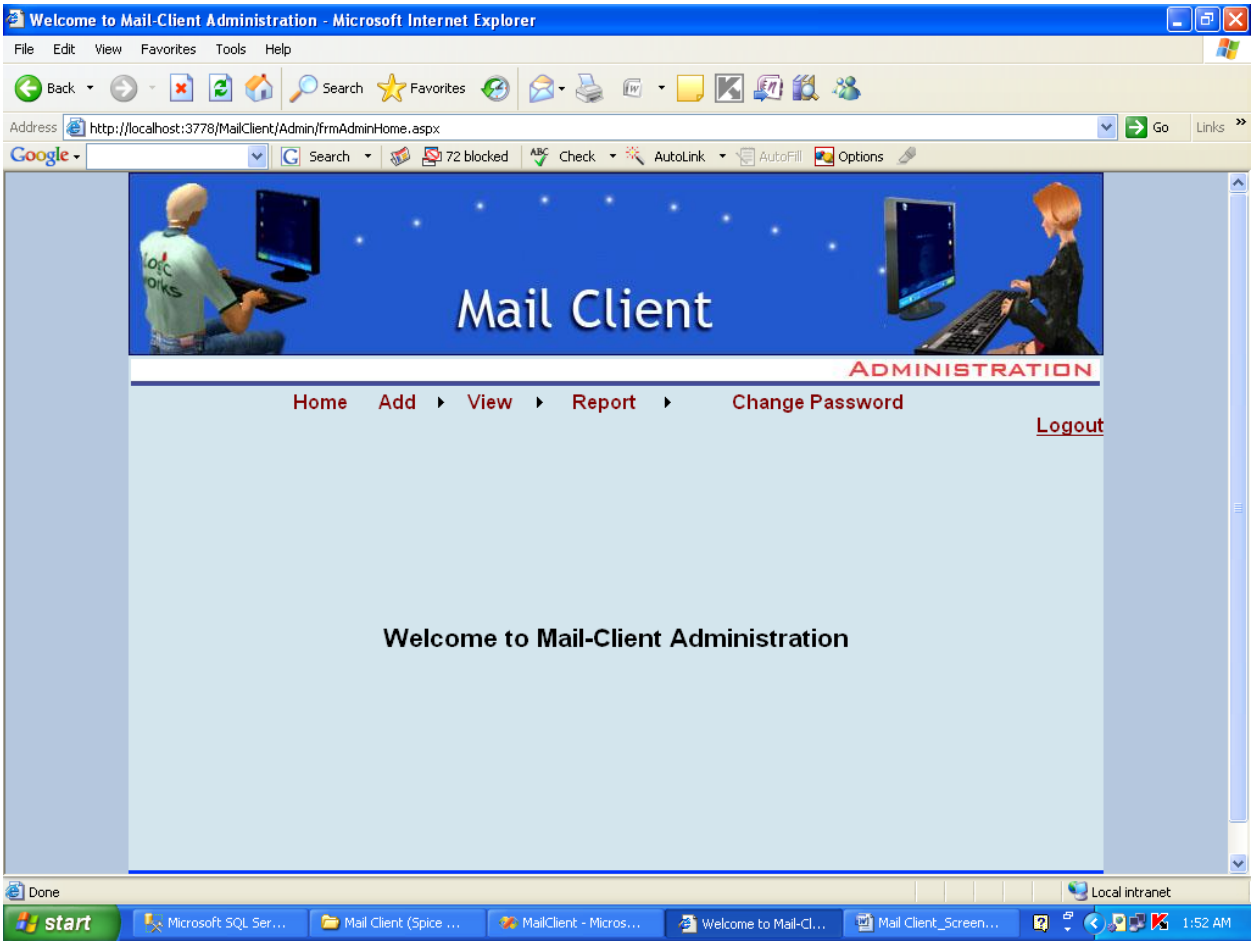
Process: enter contact details



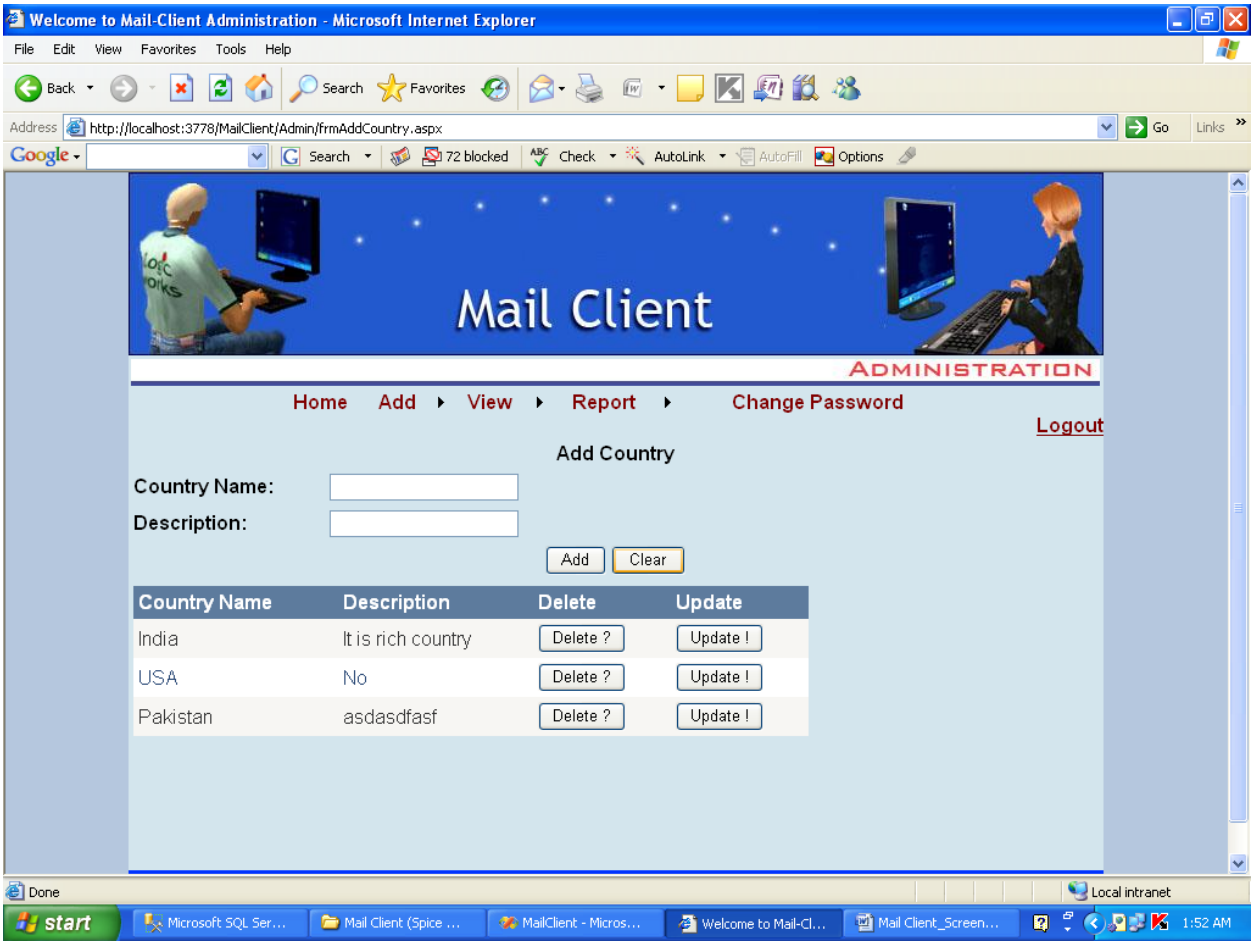
Pagename: mail
Fields:from,to,subject,add attachment,body,send,cancel
Process: enter respective details
Table: mail
Stored procedure: sp_mail



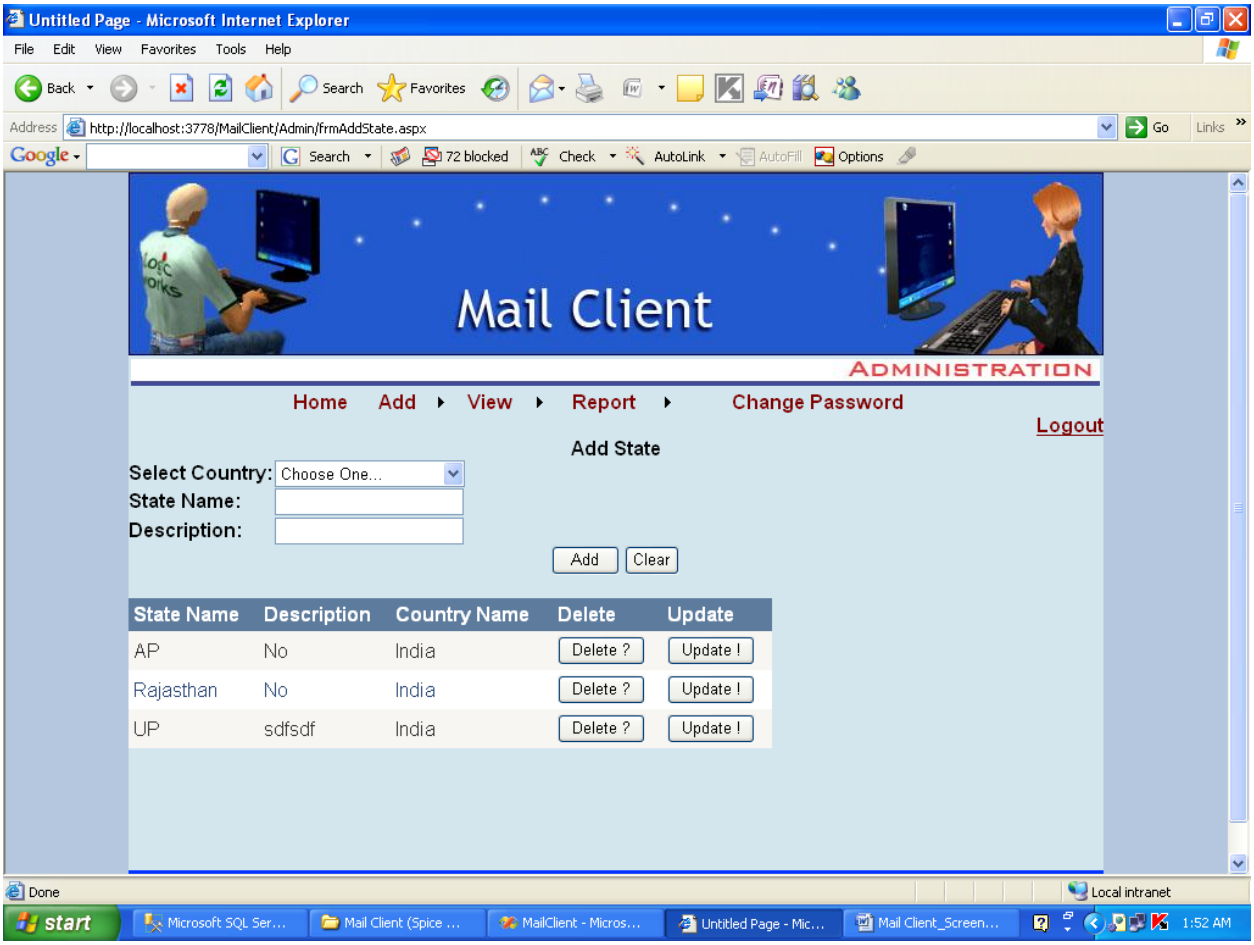
Pagename: admin
Fields: username,password
Process: neter admin credentials
Table: admin
Stored procedure: sp_admin



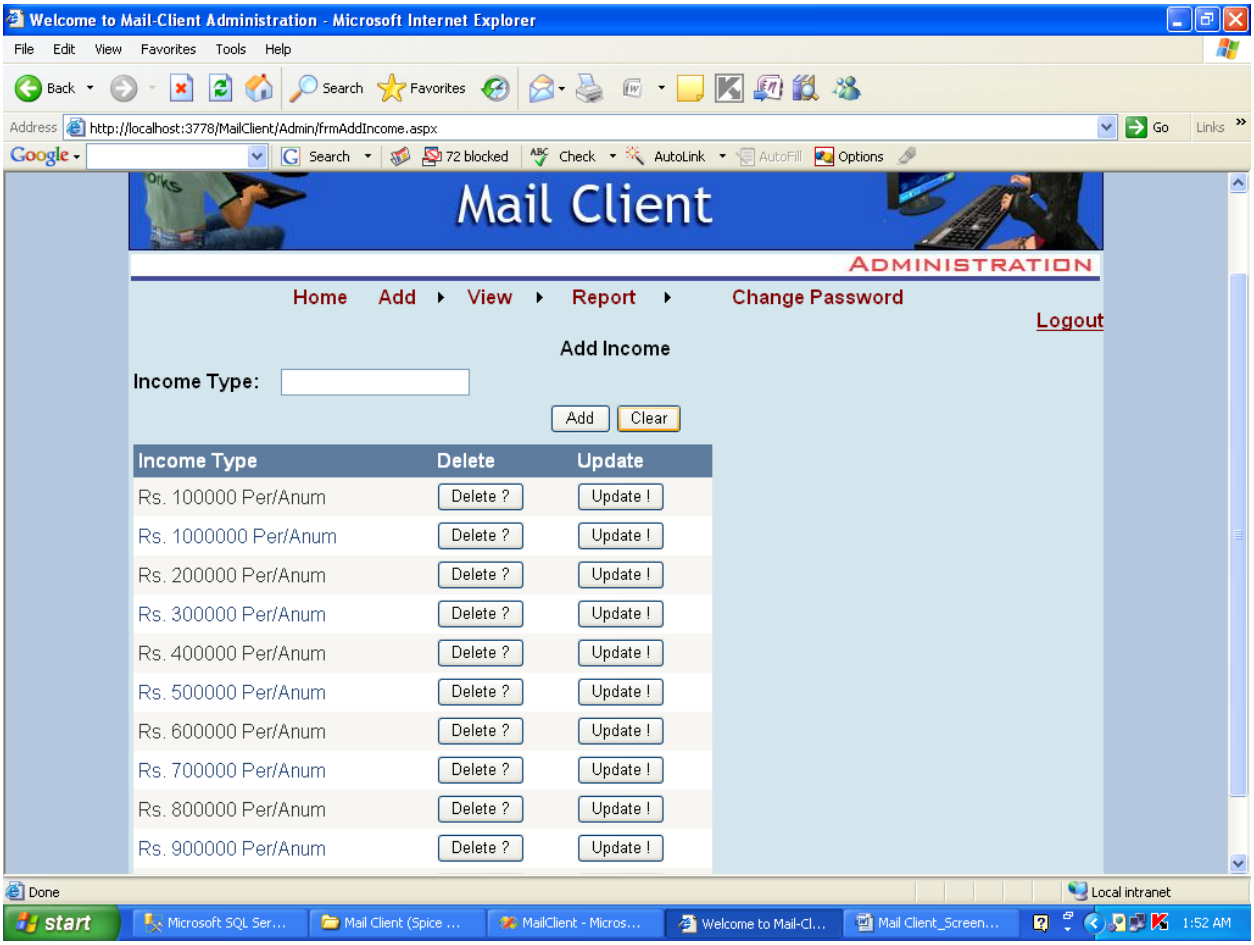
Pagename: admin home
Fields:logout
Process: home page



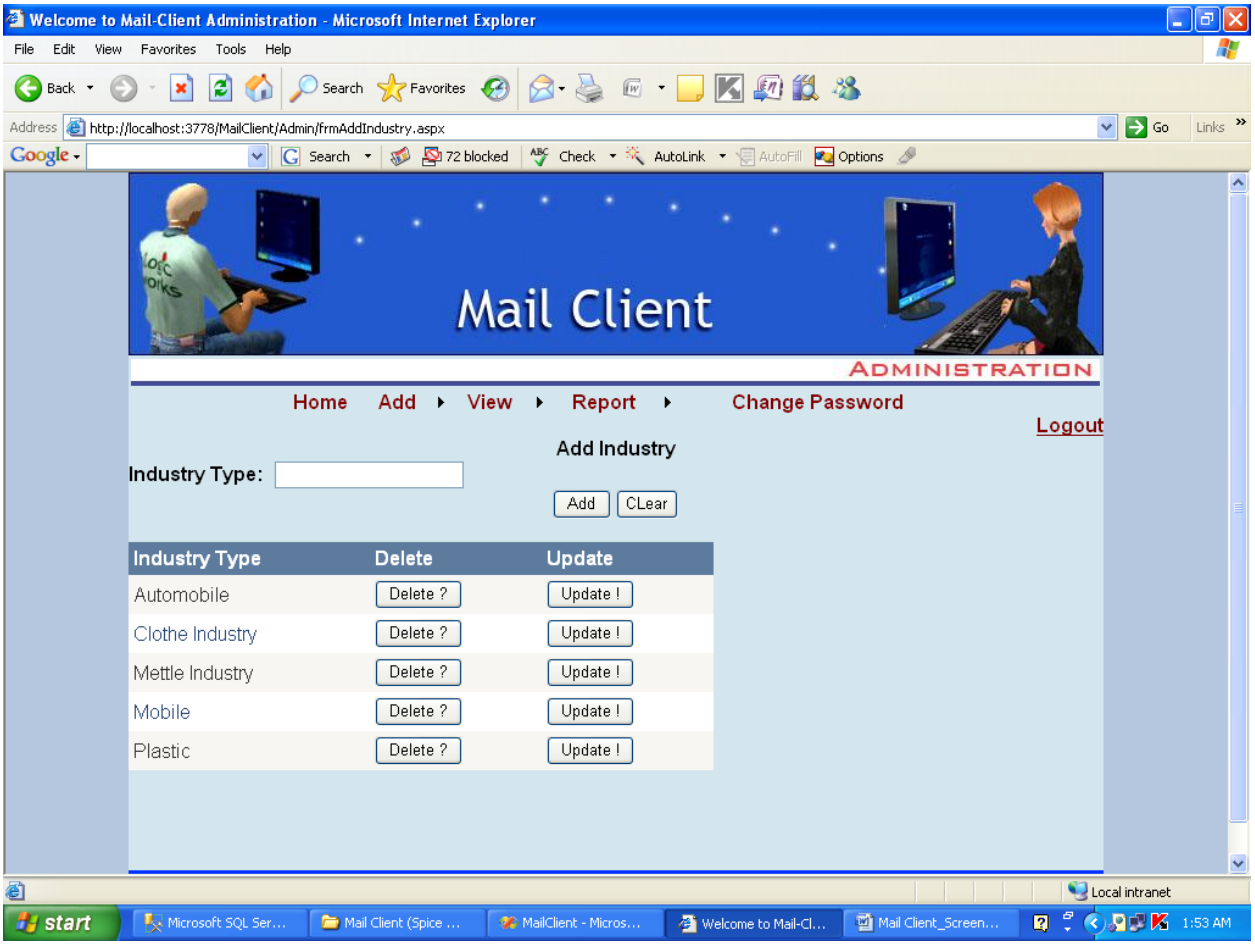
Pagename: add country
Fields: country name,description,add,clear
Process: adds country details
Table: country
Stored procedure: sp_insert_country
Remarks:adds details
Error: yes rectified



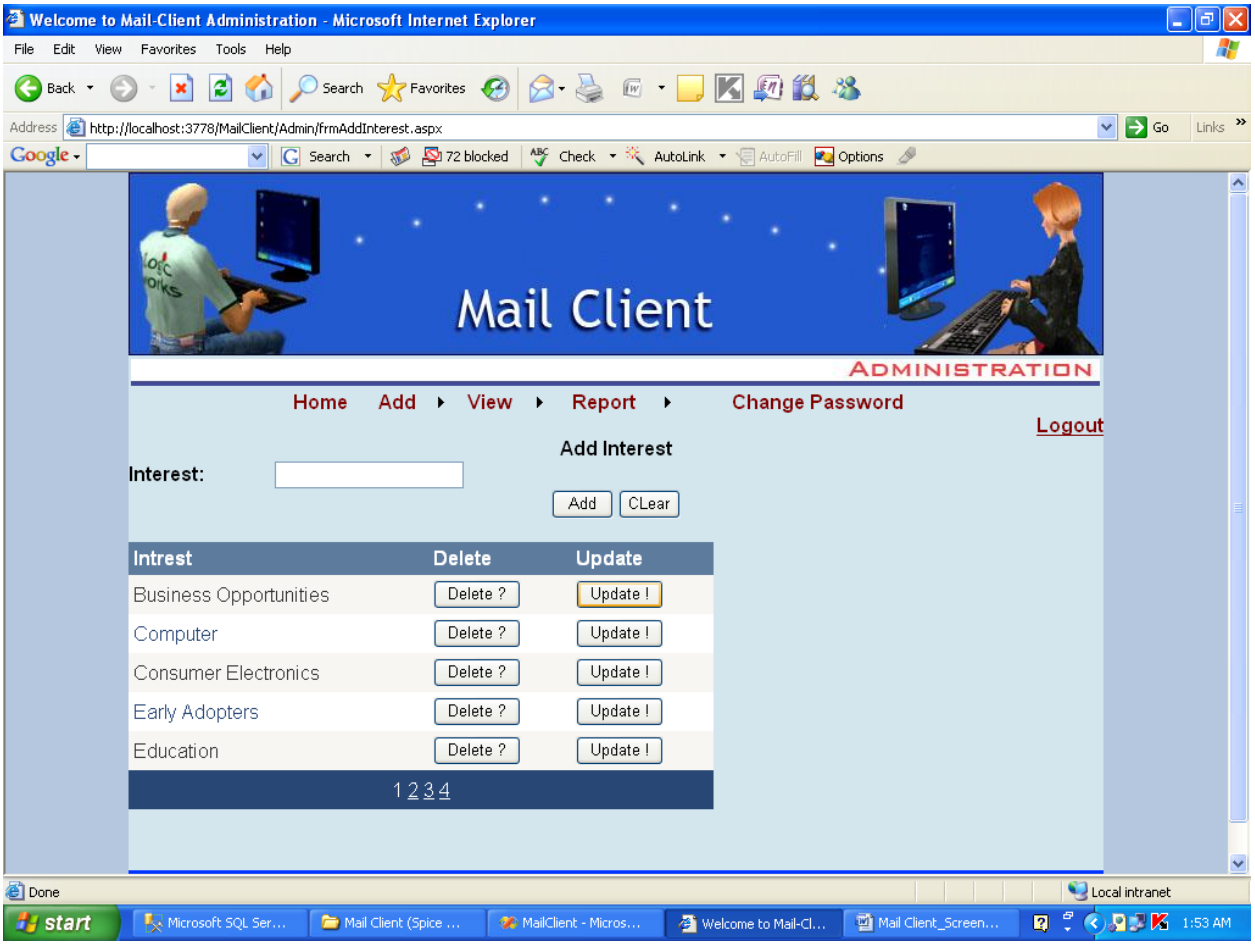
Pagename: add state
Fields: state name,description,add,clear
Process: adds state details
Table: state
Stored procedure: sp_insert_ state
Remarks:adds details
Error: yes rectified



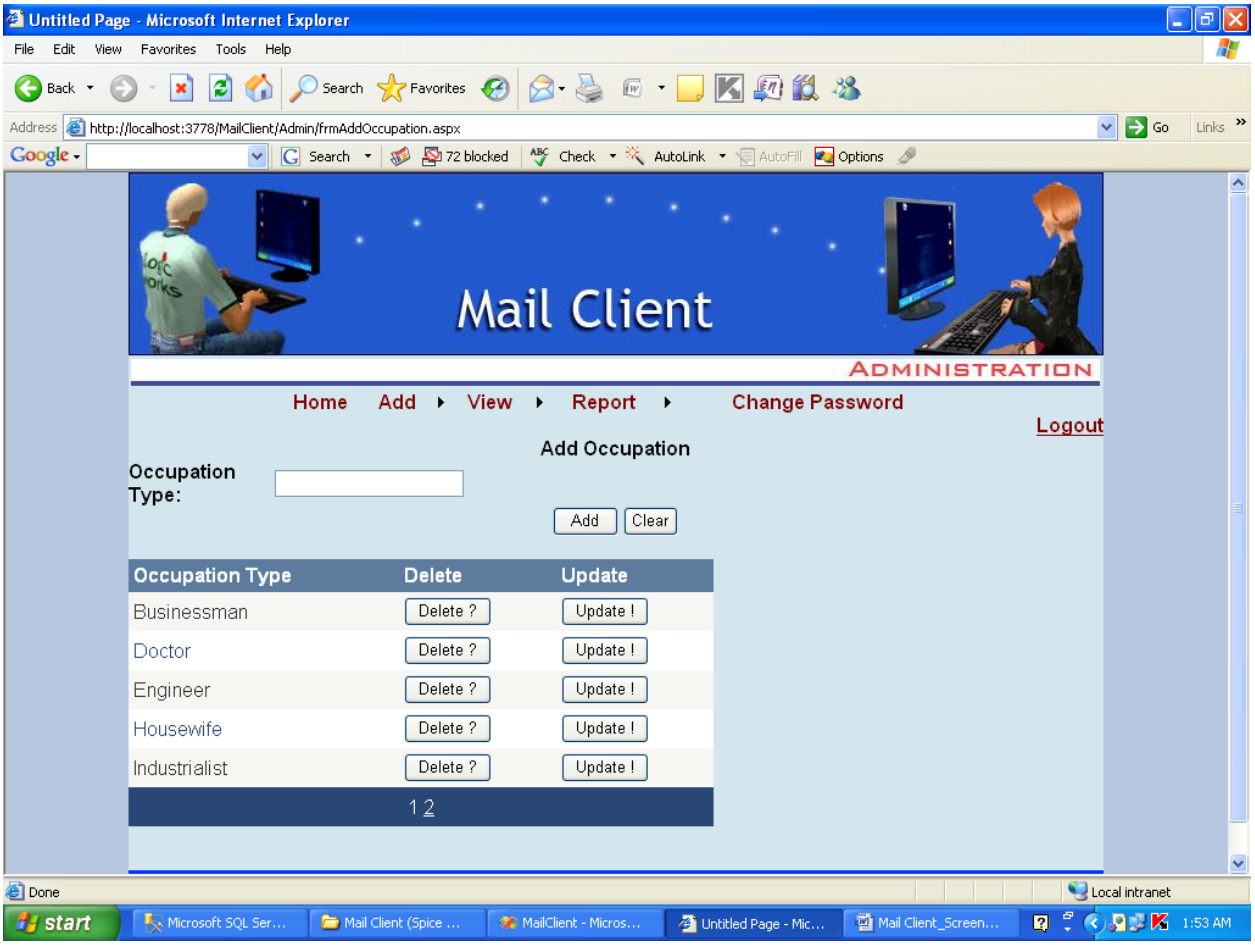
Pagename: Add income
Fields: income type
Process: adds income in gridview
Table: income
Stored procedure: sp_income
Remarks: add income
Error: binding error and rectified



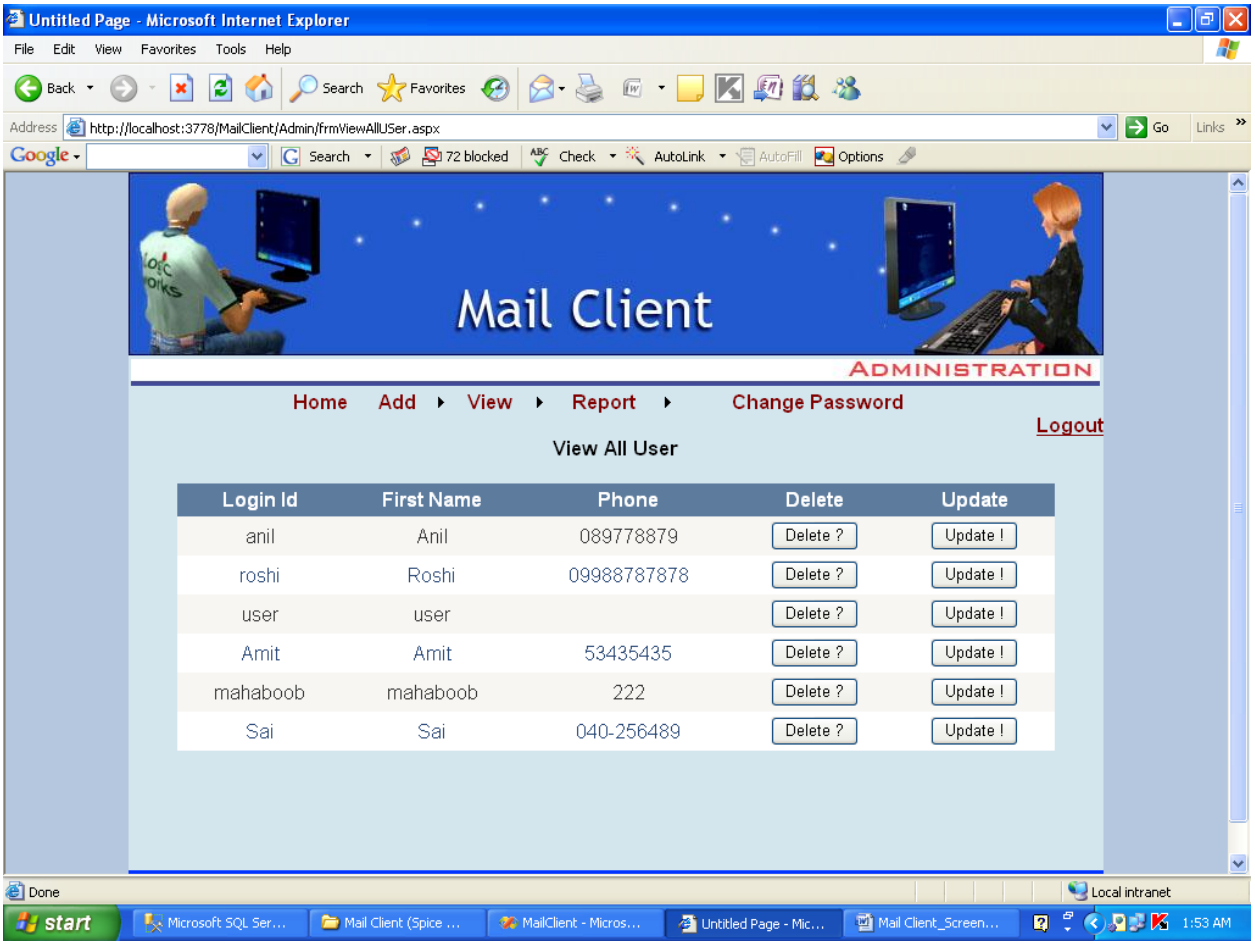
Pagename: add industry
Fields: industry name,description,add,clear
Process: adds industry details
Table: industry
Stored procedure: sp_insert_ industry
Remarks:adds details
Error: yes rectified



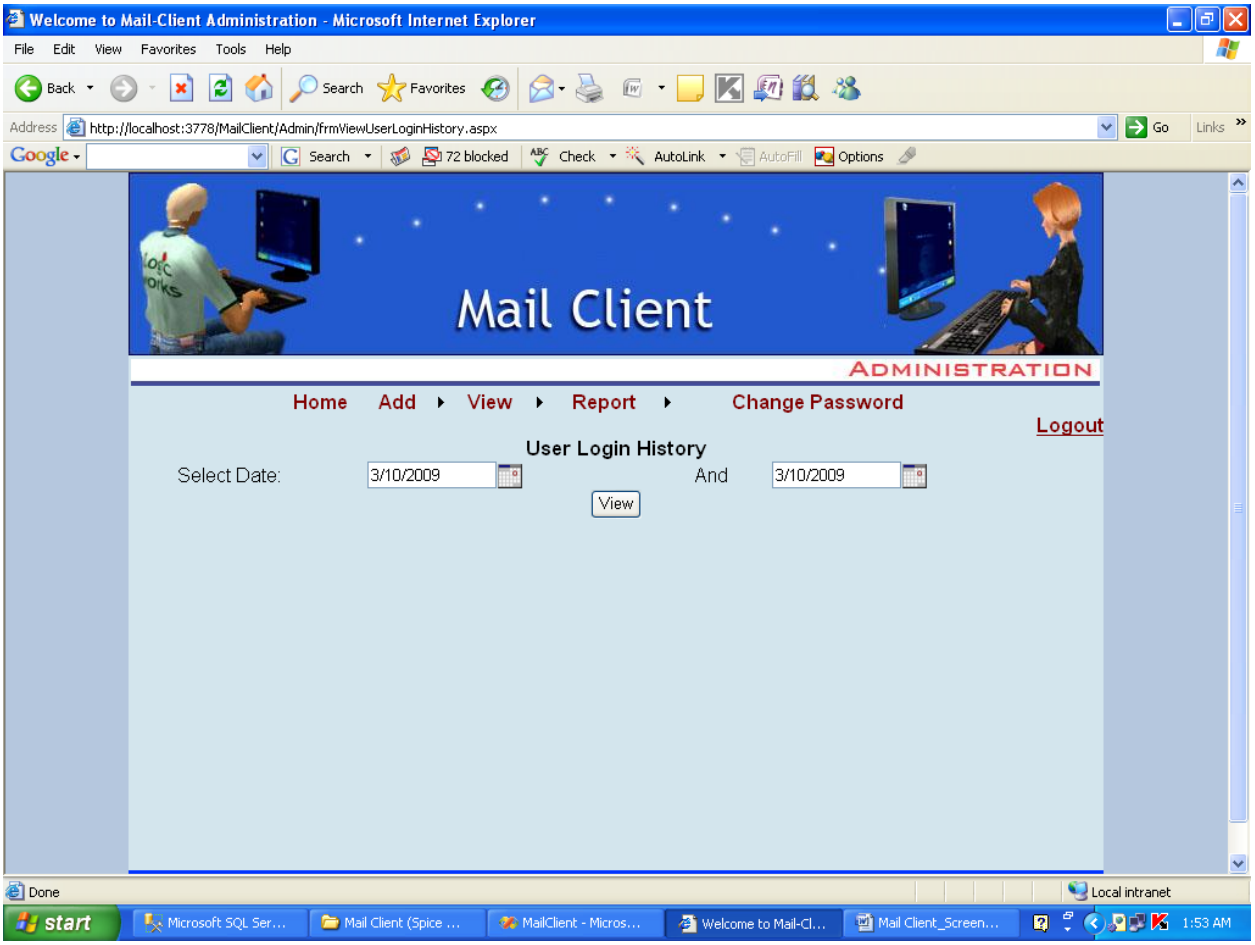
Pagename: add interest
Fields: interest name,description,add,clear
Process: adds interest details
Table: interest
Stored procedure: sp_insert_ interest
Remarks:adds details
Error: no



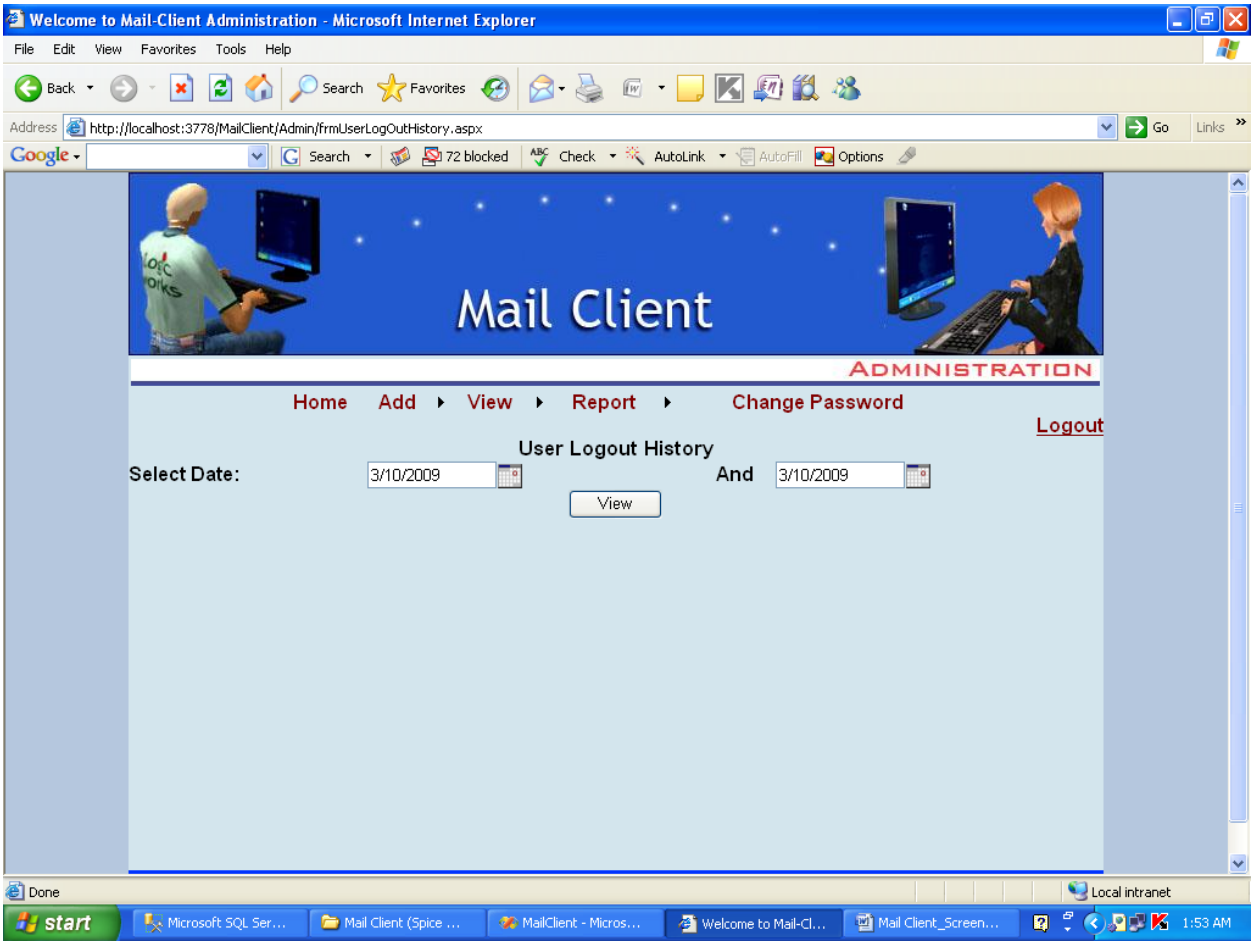
Pagename: add occupation
Fields: occupation name,description,add,clear
Process: adds occupation details
Table: occupation
Stored procedure: sp_insert_ occupation
Remarks:adds details
Error: no



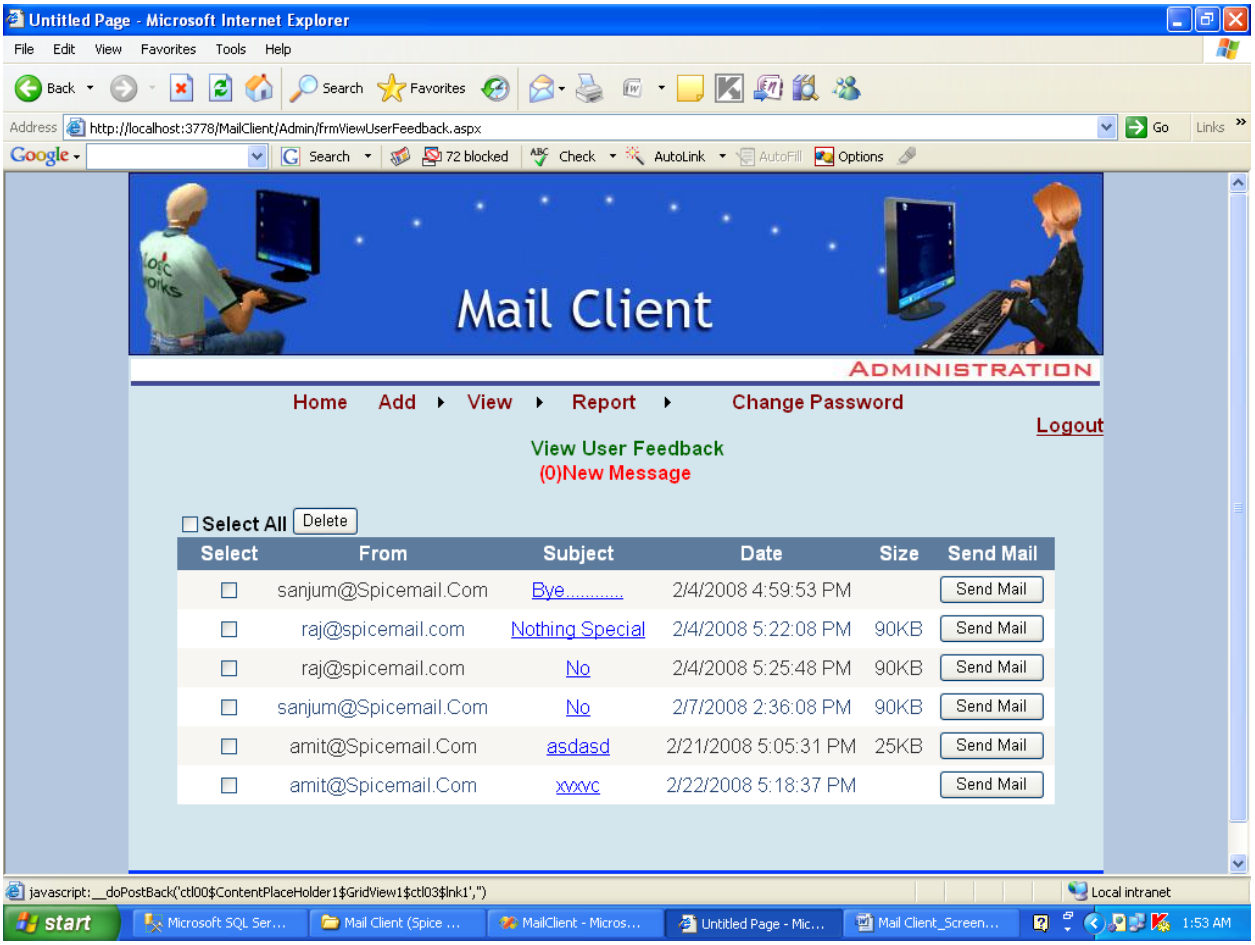
Pagename: view all user
Fields:grid view
Process: show user
Table: user details
Stored procedure:
Remarks:user details
Error: no



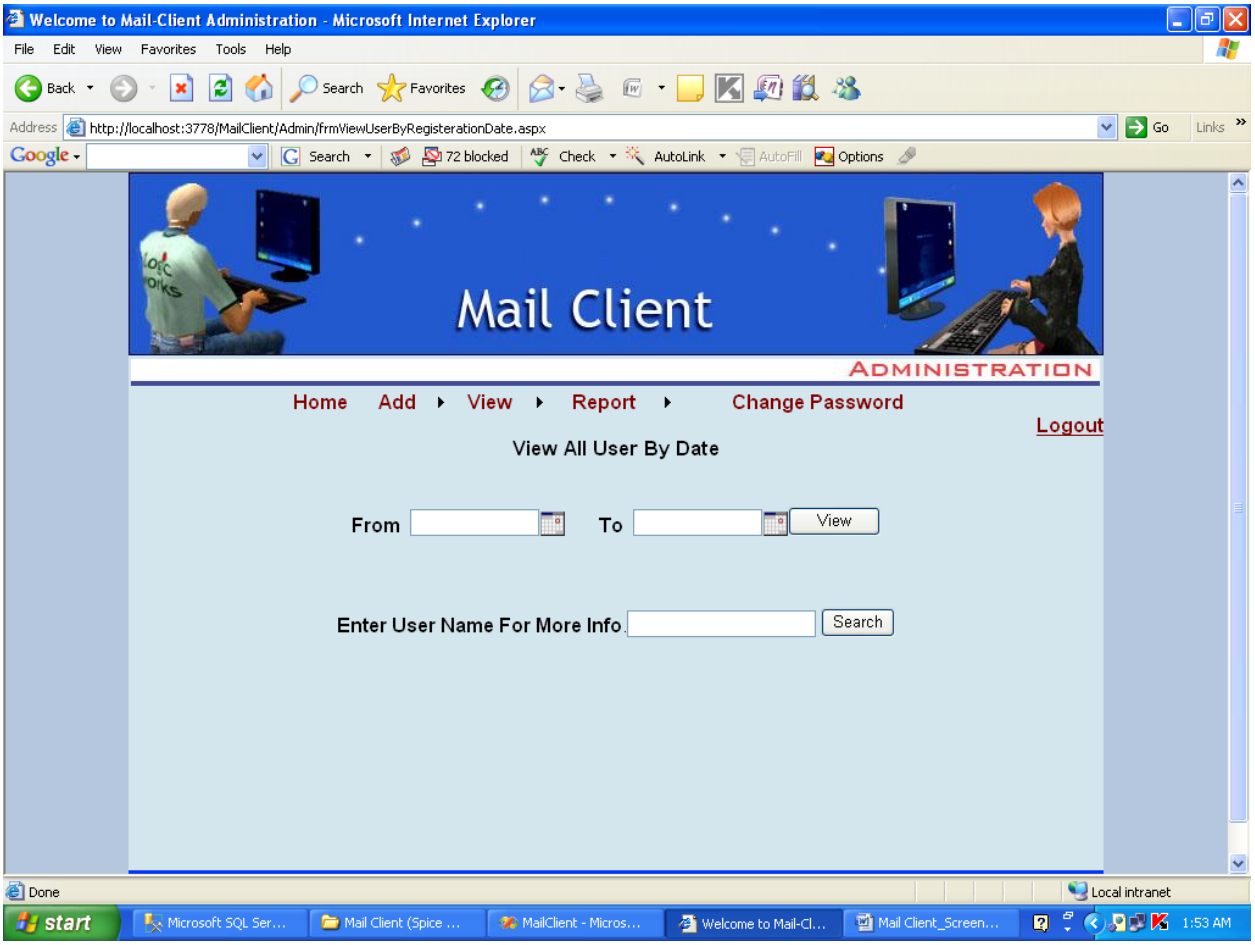
Pagename: user login histoy
Fields:date
Process: show who are login
Table: user login
Stored procedure:
Remarks: show who are login
Error: no



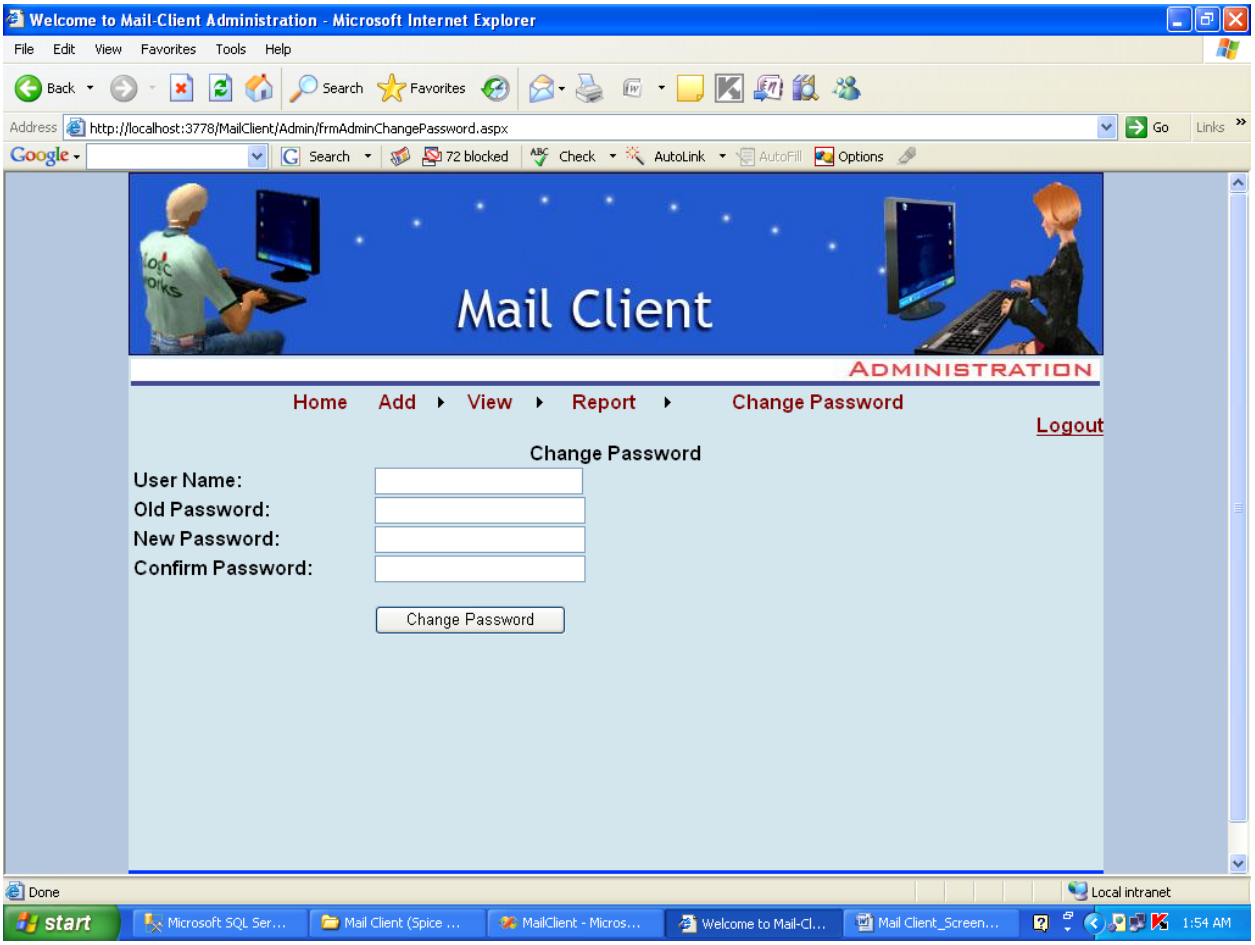
Pagename: user logout histoy
Fields:date
Process: show who are logout
Table: user logout
Stored procedure:
Remarks: show who are logout
Error: no



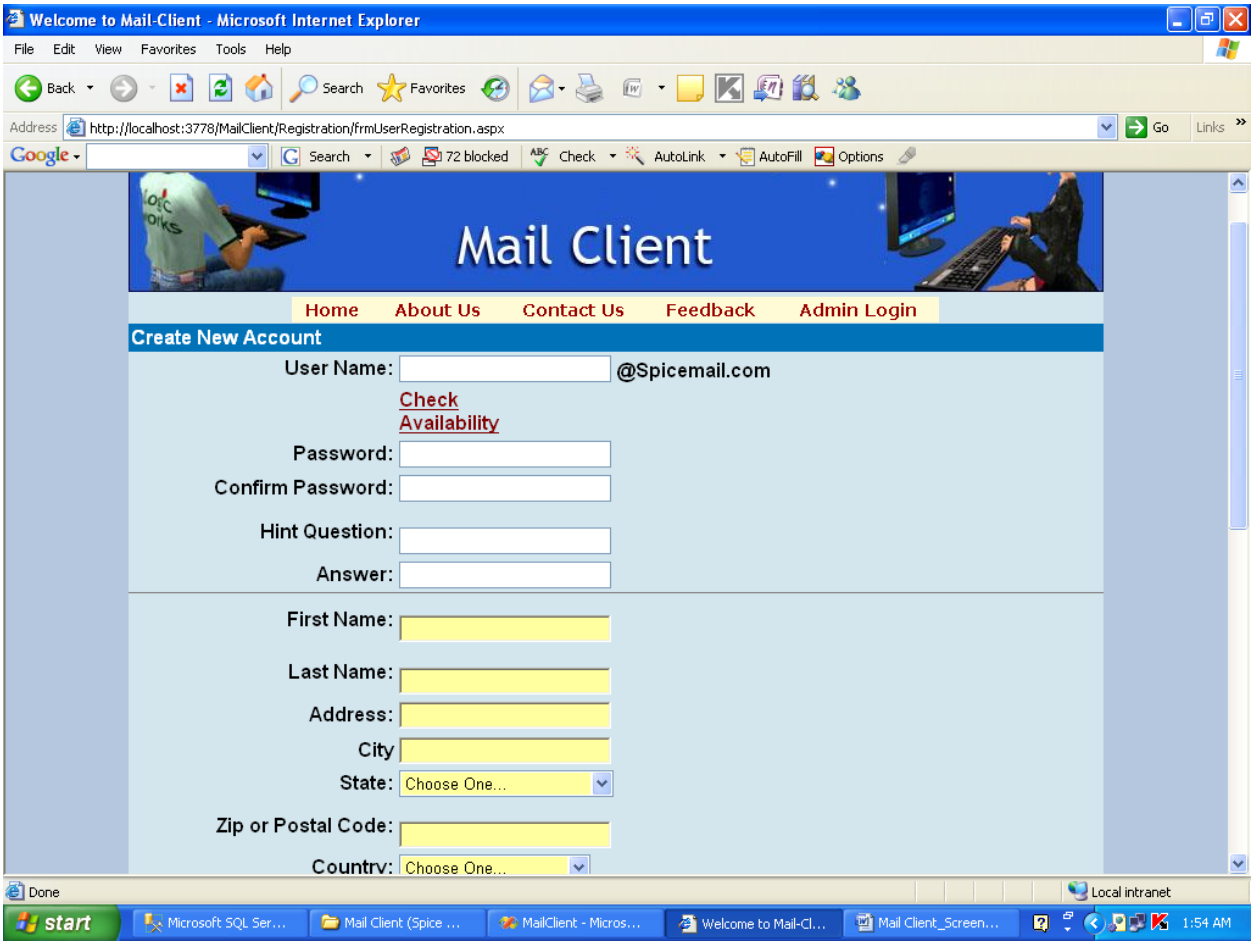
Pagename: user feed back
Fields:delete
Process: feed back of user
Table: feed back
Stored procedure: sp_feed back
Remarks:
Error: no



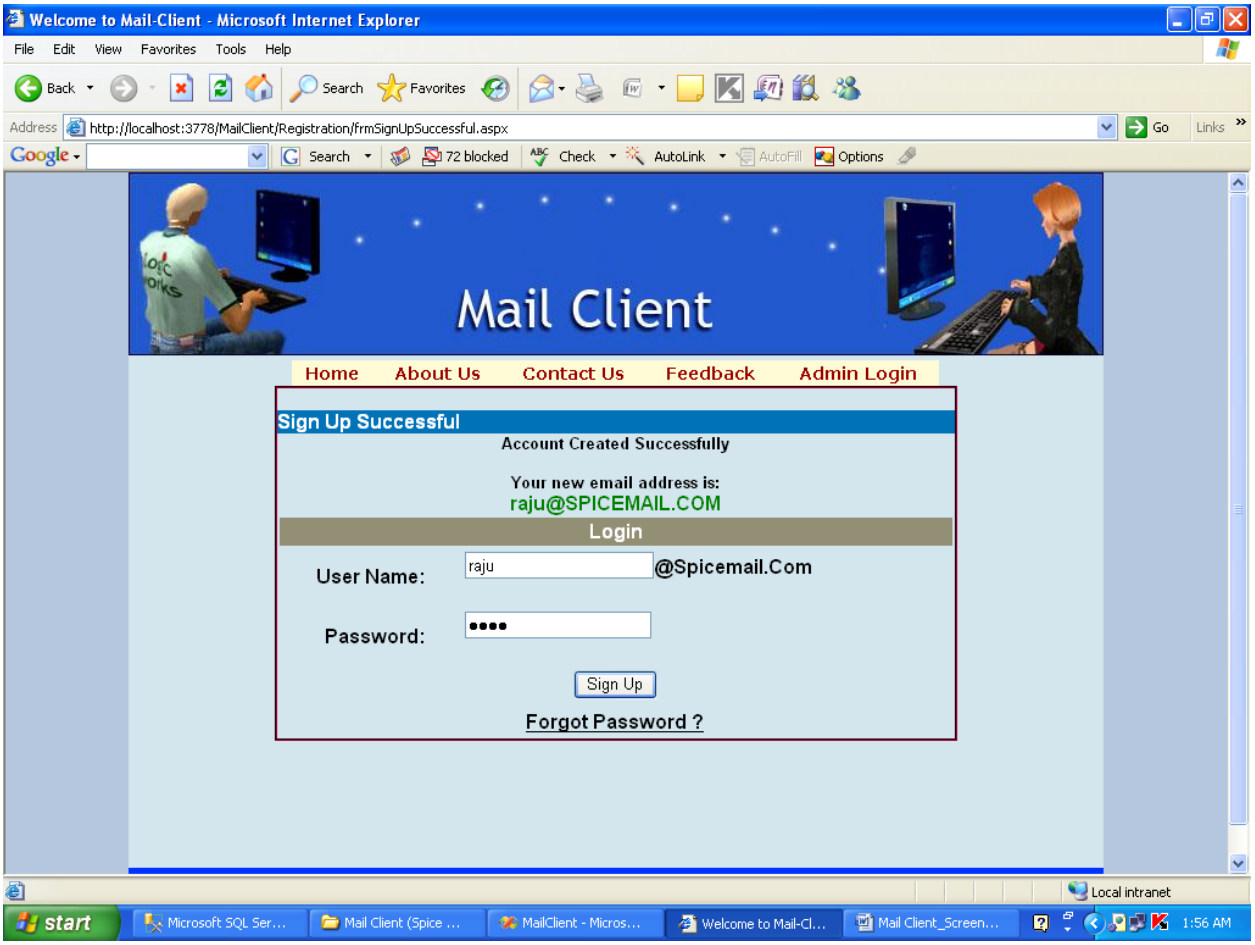
Pagename: view all user by date
Fields:from,to,user name
Process: enter fields



Pagename: change password
Fields:user name,old password,new password,confirm password
Process: changes password
Table: password
Stored procedure: sp_change password
Remarks:new password
Error: no



Pagename: create new account
Fields:pwd.hint qus,ans,name.last name,address,city,state,code,country
Process: register
Table: register
Stored procedure: sp_register
Remarks: creste an account
Error: no



Pagename: after registration
Fields:user name,password
Process: after login
Remarks:after login this will come
Error: no

CHAPTER 10

Coding

10.1 Methodology:

After the design process is performed, the design model is coded into machine under stable language. Since our project have both front end and back end so we used HTML embedded with asp java script and xml to design the visual interface in front end by the help of Microsoft Front page and MS SQL server 7 in back end. The front end has web page based interface .our work was to develop a site with robust and automatically generated interface. A process that handels data and act according to the stimuli generated from the user site this was done by the help of scripting in ASP script for server side interaction between front end and back end and javscript on the clint site to automatically generate error message when ever and wherever necessary .As well as some codes are ther in java script to facilitate the user. We have designed a large number of forms in html and object oriented approach is adopted to design the database. Each entity is described independently and a specific table is designed and also the appropriate relationship among tables is described.

To establish a connection with the back end we made a connection string in sql and then a module is designed as *connect.asp,* which connects to the sql database education under server name hardy by system dsn name contact.dsn with userid sa and password “thakur”. SINCE the website is still to be implemented fully so the connection string may change as the constraint arises when uploading the data and transferring the table and

sources to a remote server where we may need to use the connection string using server.map path dsn less connection.

CODING

WEB.CONFIG

```
<?xml version="1.0"?>
<configuration>
      <configSections>
            <sectionGroup name="system.web.extensions"
type="System.Web.Configuration.SystemWebExtensionsSectionGroup,
System.Web.Extensions, Version=3.5.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35">
                  <sectionGroup name="scripting"
type="System.Web.Configuration.ScriptingSectionGroup,
System.Web.Extensions, Version=3.5.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35">
                        <section name="scriptResourceHandler"
type="System.Web.Configuration.ScriptingScriptResourceHandlerSecti
on, System.Web.Extensions, Version=3.5.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35" requirePermission="false"
allowDefinition="MachineToApplication"/>
                        <sectionGroup name="webServices"
type="System.Web.Configuration.ScriptingWebServicesSectionGroup,
System.Web.Extensions, Version=3.5.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35">
                              <section name="jsonSerialization"
type="System.Web.Configuration.ScriptingJsonSerializationSection,
System.Web.Extensions, Version=3.5.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35" requirePermission="false"
allowDefinition="Everywhere"/>
                              <section name="profileService"
type="System.Web.Configuration.ScriptingProfileServiceSection,
System.Web.Extensions, Version=3.5.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35" requirePermission="false"
allowDefinition="MachineToApplication"/>
                              <section name="authenticationService"
type="System.Web.Configuration.ScriptingAuthenticationServiceSecti
on, System.Web.Extensions, Version=3.5.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35" requirePermission="false"
allowDefinition="MachineToApplication"/>
                              <section name="roleService"
type="System.Web.Configuration.ScriptingRoleServiceSection,
System.Web.Extensions, Version=3.5.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35" requirePermission="false"
allowDefinition="MachineToApplication"/></sectionGroup>
                  </sectionGroup>
```

```
            </sectionGroup>
      </configSections>
      <appSettings>
            <add key="ConnStr" value="Data source=AJIT-
PC\MSSQLSERVER1;Initial Catalog=MailClient;Integrated
Security=true"/>
      </appSettings>
      <system.web>
            <pages>
                  <controls>
                        <add tagPrefix="asp"
namespace="System.Web.UI" assembly="System.Web.Extensions,
Version=3.5.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35"/>
                        <add tagPrefix="asp"
namespace="System.Web.UI.WebControls"
assembly="System.Web.Extensions, Version=3.5.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35"/></controls>
            </pages>
            <!--
           Set compilation debug="true" to insert debugging
           symbols into the compiled page. Because this
           affects performance, set this value to true only
           during development.
    -->
            <compilation debug="true">
                  <assemblies>
                        <add assembly="System.Design,
Version=2.0.0.0, Culture=neutral,
PublicKeyToken=B03F5F7F11D50A3A"/>
                        <add assembly="System.Windows.Forms,
Version=2.0.0.0, Culture=neutral,
PublicKeyToken=B77A5C561934E089"/>
                        <add assembly="System.Core, Version=3.5.0.0,
Culture=neutral, PublicKeyToken=B77A5C561934E089"/>
                        <add assembly="System.Web.Extensions,
Version=3.5.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35"/>
                        <add assembly="System.Xml.Linq,
Version=3.5.0.0, Culture=neutral,
PublicKeyToken=B77A5C561934E089"/>
                        <add
assembly="System.Data.DataSetExtensions, Version=3.5.0.0,
Culture=neutral, PublicKeyToken=B77A5C561934E089"/>
                        <add assembly="System.Web.Extensions.Design,
Version=3.5.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35"/></assemblies>
            </compilation>
            <httpHandlers>
                  <remove verb="*" path="*.asmx"/>
```

```
                <add verb="*" path="*.asmx" validate="false"
type="System.Web.Script.Services.ScriptHandlerFactory,
System.Web.Extensions, Version=3.5.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35"/>
                <add verb="*" path="*_AppService.axd"
validate="false"
type="System.Web.Script.Services.ScriptHandlerFactory,
System.Web.Extensions, Version=3.5.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35"/>
                <add verb="GET,HEAD" path="ScriptResource.axd"
type="System.Web.Handlers.ScriptResourceHandler,
System.Web.Extensions, Version=3.5.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35" validate="false"/>
          </httpHandlers>
          <httpModules>
                <add name="ScriptModule"
type="System.Web.Handlers.ScriptModule, System.Web.Extensions,
Version=3.5.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35"/>
          </httpModules>
     </system.web>
     <system.web.extensions>
          <scripting>
                <webServices>
                     <!-- Uncomment this line to customize
maxJsonLength and add a custom converter -->
                     <!--
      <jsonSerialization maxJsonLength="500">
        <converters>
          <add name="ConvertMe"
type="Acme.SubAcme.ConvertMeTypeConverter"/>
        </converters>
      </jsonSerialization>
      -->
                     <!-- Uncomment this line to enable the
authentication service. Include requireSSL="true" if appropriate.
-->
                     <!--
        <authenticationService enabled="true" requireSSL =
"true|false"/>
      -->
                     <!-- Uncomment these lines to enable the
profile service. To allow profile properties to be retrieved
          and modified in ASP.NET AJAX applications, you need to
add each property name to the readAccessProperties and
          writeAccessProperties attributes. -->
                     <!--
      <profileService enabled="true"

readAccessProperties="propertyname1,propertyname2"
```

```
writeAccessProperties="propertyname1,propertyname2" />
      -->
                  </webServices>
                  <!--
      <scriptResourceHandler enableCompression="true"
enableCaching="true" />
      -->
            </scripting>
      </system.web.extensions>
      <system.webServer>
            <validation
validateIntegratedModeConfiguration="false"/>
            <modules>
                  <remove name="ScriptModule"/><add
name="ScriptModule" preCondition="managedHandler"
type="System.Web.Handlers.ScriptModule, System.Web.Extensions,
Version=3.5.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35"/>
            </modules>
            <handlers>
                  <remove name="ScriptHandlerFactory"/>
                  <remove name="ScriptHandlerFactoryAppServices"/>
                  <remove name="ScriptResource"/><remove
name="WebServiceHandlerFactory-Integrated"/>
                  <add name="ScriptHandlerFactory" verb="*"
path="*.asmx" preCondition="integratedMode"
type="System.Web.Script.Services.ScriptHandlerFactory,
System.Web.Extensions, Version=3.5.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35"/>
                  <add name="ScriptHandlerFactoryAppServices"
verb="*" path="*_AppService.axd" preCondition="integratedMode"
type="System.Web.Script.Services.ScriptHandlerFactory,
System.Web.Extensions, Version=3.5.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35"/>
                  <add name="ScriptResource"
preCondition="integratedMode" verb="GET,HEAD"
path="ScriptResource.axd"
type="System.Web.Handlers.ScriptResourceHandler,
System.Web.Extensions, Version=3.5.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35"/>
            </handlers>
      </system.webServer>
      <system.codedom>
            <compilers>
                  <compiler language="c#;cs;csharp" extension=".cs"
type="Microsoft.CSharp.CSharpCodeProvider,System, Version=2.0.0.0,
Culture=neutral, PublicKeyToken=b77a5c561934e089"
warningLevel="4">
                        <providerOption name="CompilerVersion"
value="v3.5"/>
```

```
                    <providerOption name="WarnAsError"
value="false"/></compiler>
                <compiler language="vb;vbs;visualbasic;vbscript"
extension=".vb" type="Microsoft.VisualBasic.VBCodeProvider,
System, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089" warningLevel="4">
                    <providerOption name="CompilerVersion"
value="v3.5"/>
                    <providerOption name="OptionInfer"
value="true"/>
                    <providerOption name="WarnAsError"
value="false"/></compiler></compilers></system.codedom>
     <runtime>
          <assemblyBinding xmlns="urn:schemas-microsoft-
com:asm.v1" appliesTo="v2.0.50727"><dependentAssembly>
                    <assemblyIdentity
name="System.Web.Extensions" publicKeyToken="31bf3856ad364e35"/>
                    <bindingRedirect oldVersion="1.0.0.0-
1.1.0.0" newVersion="3.5.0.0"/></dependentAssembly>
                <dependentAssembly>
                    <assemblyIdentity
name="System.Web.Extensions.Design"
publicKeyToken="31bf3856ad364e35"/>
                    <bindingRedirect oldVersion="1.0.0.0-
1.1.0.0"
newVersion="3.5.0.0"/></dependentAssembly></assemblyBinding></runt
ime></configuration>
```

DEFAULT.ASPX

```
using System;
using System.Data;
using System.Configuration;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

public partial class _Default : System.Web.UI.Page
{
    UserRegistrationBL registeration = new UserRegistrationBL();
    protected void Page_Load(object sender, EventArgs e)
    {

    }
    protected void btnSignUp_Click(object sender, EventArgs e)
```

```
    {
        try
        {
            registeration.LoginName = txtUserName.Text.Trim();
            registeration.Password = txtPassword.Text.Trim();
            if (registeration.CheckUserValidity() == true)
            {
                Session["UserName"] = registeration.LoginName;
                registeration.LoginName = txtUserName.Text.Trim();
                registeration.LoginDate =
System.DateTime.Now.Date;
                registeration.LoginTime =
System.DateTime.Now.ToShortTimeString();
                registeration.InsertUserLoginHistory();

Response.Redirect("~/Registration/frmUserHomePage.aspx");


            }
            else
            {

Response.Redirect("~/Registration/frmInvalidUserNamePassword.aspx"
);
            }
        }
        catch (Exception)
        {

            throw;
        }


    }

    protected void lnkForgotPassword_Click(object sender,
EventArgs e)
    {

Response.Redirect("~/Registration/frmPasswordForgot.aspx");
    }
    protected void imgRegister_Click(object sender,
ImageClickEventArgs e)
    {

Response.Redirect("~/Registration/frmUserRegistration.aspx");
    }
}
```

FEEDBACK.ASPX

```
using System;
using System.Data;
using System.Configuration;
using System.Collections;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;
using System.IO;
public partial class frmFeedback : System.Web.UI.Page
{
    UserFeedbackBL feedback = new UserFeedbackBL();
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            FileUpload1.Visible = false;
            lnkRemove.Visible = false;

        }
    }
    protected void imgSend1_Click(object sender,
ImageClickEventArgs e)
    {
        try
        {
            feedback.From = txtFrom.Text.Trim();
            feedback.To = txtTo.Text.Trim();
            feedback.Subject = txtSubject.Text.Trim();
            feedback.Feedback = txtMailMessage.Text.Trim();
            feedback.Date = System.DateTime.Now;
            string Attachments;
            if (FileUpload1.HasFile)
            {
                Attachments = Server.MapPath("~/Attachments/");
                if (!Directory.Exists(Attachments))
                    Directory.CreateDirectory(Attachments);
                FileUpload1.PostedFile.SaveAs(Attachments +
FileUpload1.FileName);
                feedback.Attachement = FileUpload1.FileName;
                int size = 0;
                size = FileUpload1.PostedFile.ContentLength /
1024;

                feedback.Size = size.ToString() + "KB";
            }
            else
            {
                feedback.Attachement = "";
```

```
                feedback.Size = "";
            }


            feedback.InsertFeedback();
            Session["To"] = txtTo.Text.Trim();

Response.Redirect("~/frmSendFeedbackSuccessfully.aspx");

        }
        catch (Exception)
        {

            throw;
        }
    }
    protected void lnkAttachment_Click(object sender, EventArgs e)
    {

        FileUpload1.Visible = true;
        lnkRemove.Visible = true;
    }
    protected void lnkRemove_Click(object sender, EventArgs e)
    {
        FileUpload1.Visible = false;
        lnkRemove.Visible = false;
    }
    protected void ImgCancel1_Click(object sender,
ImageClickEventArgs e)
    {
        Response.Redirect("~/Default.aspx");
    }
    protected void ImgSend2_Click(object sender,
ImageClickEventArgs e)
    {
        try
        {
            feedback.From = txtFrom.Text.Trim();
            feedback.To = txtTo.Text.Trim();
            feedback.Subject = txtSubject.Text.Trim();
            feedback.Feedback = txtMailMessage.Text.Trim();
            feedback.Date = System.DateTime.Now;
            string Attachments;
            if (FileUpload1.HasFile)
            {
                Attachments = Server.MapPath("~/Attachments/");
                if (!Directory.Exists(Attachments))
                    Directory.CreateDirectory(Attachments);
                FileUpload1.PostedFile.SaveAs(Attachments +
FileUpload1.FileName);
                feedback.Attachement = FileUpload1.FileName;
```

```
                int size = 0;
                size = FileUpload1.PostedFile.ContentLength /
1024;

                feedback.Size = size.ToString() + "KB";
            }
            else
            {
                feedback.Attachement = "";
                feedback.Size = "";
            }


            feedback.InsertFeedback();
            Session["To"] = txtTo.Text.Trim();

Response.Redirect("~/frmSendFeedbackSuccessfully.aspx");
        }
        catch (Exception)
        {

            throw;
        }

    }
    protected void ImgCancel2_Click(object sender,
ImageClickEventArgs e)
    {
        Response.Redirect("~/Default.aspx");

    }
}
```

SENDFEEDBACK.ASPX

```
using System;
using System.Data;
using System.Configuration;
using System.Collections;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

public partial class frmSendFeedbackSuccessfully :
System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
```

```
    {
        if (!IsPostBack)
        {
            if (Session["To"] != null)
                lblName.Text = Session["To"].ToString();
            else
                Response.Redirect("Default.aspx");
        }

    }
}
```

REGISTRATION.ASPX

```
using System;
using System.Data;
using System.Configuration;
using System.Collections;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

public partial class Registration_frmInvalidUserNamePassword :
System.Web.UI.Page
{
    UserRegistrationBL registeration = new UserRegistrationBL();
    protected void Page_Load(object sender, EventArgs e)
    {

    }
    protected void btnSignUp_Click(object sender, EventArgs e)
    {
        try
        {
            registeration.LoginName = txtLoginName.Text.Trim();
            registeration.Password = txtPassword.Text.Trim();
            if (registeration.CheckUserValidity() == true)
            {
                Session["UserName"] = registeration.LoginName;
                registeration.LoginName =
txtLoginName.Text.Trim();
                registeration.LoginDate =
System.DateTime.Now.Date;
                registeration.LoginTime =
System.DateTime.Now.ToShortTimeString();
                registeration.InsertUserLoginHistory();
```

```
Response.Redirect("~/Registration/frmUserHomePage.aspx");


            }
            else
            {
                lblMsg.Text = "Try Again...!";
                txtLoginName.Focus();

            }
        }
        catch (Exception)
        {

            throw;
        }

    }


    protected void lnkNewUser_Click(object sender, EventArgs e)
    {

Response.Redirect("~/Registration/frmUserRegistration.aspx");
    }
    protected void lnkHome_Click(object sender, EventArgs e)
    {
        Response.Redirect("~/Default.aspx");
    }
    protected void lnkPasswordForgot_Click(object sender,
EventArgs e)
    {

Response.Redirect("~/Registration/frmPasswordForgot.aspx");
    }
}
```

NEWUSERPASS.ASPX

```
using System;
using System.Data;
using System.Configuration;
using System.Collections;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;
```

```
public partial class Registration_frmNewUserPassword :
System.Web.UI.Page
{
    UserRegistrationBL registration = new UserRegistrationBL();
    protected void Page_Load(object sender, EventArgs e)
    {

        if (!IsPostBack)
        {
            BindPassword();

        }

    }
    private void BindPassword()
    {
        lblPassword.Text = Session["Password"].ToString();
    }
    protected void btnLogin_Click(object sender, EventArgs e)
    {
        Response.Redirect("~/Default.aspx");
    }
}
```

FORGETPASS.ASPX

```
using System;
using System.Data;
using System.Configuration;
using System.Collections;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

public partial class Registration_frmPasswordForgot :
System.Web.UI.Page
{
    UserRegistrationBL registration = new UserRegistrationBL();
    protected void Page_Load(object sender, EventArgs e)
    {


    }
    protected void lnkNewUser_Click(object sender, EventArgs e)
    {
```

```
Response.Redirect("~/Registration/frmUserRegistration.aspx");
    }

    protected void btnContinue_Click(object sender, EventArgs e)
    {
        try
        {
            registration.LoginName = txtLoginName.Text.Trim();
            registration.Question = txtQuestion.Text.Trim();
            registration.Answer = txtAnswer.Text.Trim();
            if (registration.CheckPasswordInfo() == true)
            {
                DataSet ds = new DataSet();
                ds = registration.SelectPassword();
                int num,num1;
                Random ran = new Random();
                Random ran1 = new Random();
                num = ran.Next(100);
                num1 = ran1.Next(1000);

registration.Password=num+ds.Tables[0].Rows[0][0].ToString()+num1;
                registration.UpdateNewPassword();
                Session["Password"] = registration.Password;

Response.Redirect("~/Registration/frmNewUserPassword.aspx");
            }
            else
            {
                lblMsg.Text = "Enter Correct Value...!";

            }
        }
        catch (Exception)
        {

            throw;
        }
    }
    protected void btnCancel_Click(object sender, EventArgs e)
    {
        Response.Redirect("~/Default.aspx");
        txtAnswer.Text = "";
        txtLoginName.Text = "";
        txtQuestion.Text = "";


    }
}
```

SIGNUP.ASPX

```
using System;
using System.Data;
using System.Configuration;
using System.Collections;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

public partial class Registration_frmSignUpSuccessful :
System.Web.UI.Page
{
    UserRegistrationBL registeration = new UserRegistrationBL();
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
           lblUserName.Text =
Session["UserName"].ToString()+"@SPICEMAIL.COM";
        }

    }

    protected void btnSignUp_Click(object sender, EventArgs e)
    {

        try
        {
            txtUserName.Text = Session["UserName"].ToString();
            registeration.LoginName = txtUserName.Text.Trim();
            registeration.Password = txtPassword.Text.Trim();
            if (registeration.CheckUserValidity() == true)
            {
                Session["UserName"] = registeration.LoginName;

Response.Redirect("~/Registration/frmUserHomePage.aspx");


            }
        }
        catch (Exception ex)
        {

            lblMsg.Text = ex.Message;
            lblMsg.Visible = true;
        }
```

```
    }
    protected void lnkForgotPassword_Click(object sender,
EventArgs e)
    {
        Response.Redirect("frmPasswordForgot.aspx");
    }
}
```

ADDRESSBOOK.ASPX

```
using System;
using System.Data;
using System.Configuration;
using System.Collections;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

public partial class Registration_RegisterUser_frmSeeContactBook :
System.Web.UI.Page
{
    UserAddressBookBL address = new UserAddressBookBL();
    protected void Page_Load(object sender, EventArgs e)
    {
        if (Session["UserName"] == null)
        {
            Response.Redirect("~/Default.aspx");
        }
        if (!IsPostBack)
        {
            BindSearchLetters();
        }

    }
    private void BindSearchLetters()
    {
        try
        {
            string strLetters =
"ALL,A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S,T,U,V,W,X,Y,Z";
            string[] strArray = strLetters.Split(',');
            DataTable dt = new DataTable();
            DataColumn dc = new DataColumn("Letter");
            dt.Columns.Add(dc);
            for (int i = 0; i < strArray.Length; i++)
            {
```

```
                DataRow dr = dt.NewRow();
                dr[0] = strArray[i];
                dt.Rows.Add(dr);
            }
            DataList1.DataSource = dt;
            DataList1.DataBind();
        }
        catch (Exception ex)
        {
            Label1.Text = ex.Message;
        }
    }
    private void BindGridview()
    {
        if (ddlContactType.SelectedIndex == 1 &&
ddlNameType.SelectedIndex == 1)
        {
            address.LoginName = Session["UserName"].ToString();
            address.FirstName = txtName.Text.Trim();
            GridView1.DataSource =
address.SelectDetailOnFirstName();
            GridView1.DataBind();
        }
        else if (ddlContactType.SelectedIndex == 2 &&
ddlNameType.SelectedIndex == 1)
        {
            address.LoginName = Session["UserName"].ToString();
            address.FirstName = txtName.Text.Trim();
            GridView1.DataSource =
address.SelectOfficialDetailOnFirstName();
            GridView1.DataBind();
        }
        else if (ddlContactType.SelectedIndex == 1 &&
ddlNameType.SelectedIndex == 2)
        {
            address.LoginName = Session["UserName"].ToString();
            address.LastName = txtName.Text.Trim();
            GridView1.DataSource =
address.SelectDetailOnLastName();
            GridView1.DataBind();
        }
        else if (ddlContactType.SelectedIndex == 2 &&
ddlNameType.SelectedIndex == 2)
        {
            address.LoginName = Session["UserName"].ToString();
            address.LastName = txtName.Text.Trim();
            GridView1.DataSource =
address.SelectOfficialDetailOnLastName();
            GridView1.DataBind();
        }
```

```
        else if (ddlContactType.SelectedIndex == 2 &&
ddlNameType.SelectedIndex == 3)
        {
            address.LoginName = Session["UserName"].ToString();
            address.CompanyName = txtName.Text.Trim();
            GridView1.DataSource =
address.SelectOfficialDetailOnCompanyName();
            GridView1.DataBind();
        }
    }
    protected void Button1_Click(object sender, EventArgs e)
    {
        BindGridview();
    }


    protected void GridView1_PageIndexChanging(object sender,
GridViewPageEventArgs e)
    {
        GridView1.PageIndex = e.NewPageIndex;
        BindGridview();
    }

    protected void btnMail_Click(object sender, EventArgs e)
    {

Response.Redirect("~/Registration/RegisterUser/frmUserInbox.aspx")
;
    }
    protected void btnNewContact_Click(object sender, EventArgs e)
    {

Response.Redirect("~/Registration/RegisterUser/frmAddContactType.a
spx");
    }
    protected void btnShowConatct_Click(object sender, EventArgs
e)
    {

Response.Redirect("~/Registration/RegisterUser/frmShowContactByTyp
e.aspx");
    }

    private void BindGridview1()
    {

        if (ddlContactType1.SelectedIndex == 1 &&
ddlBrowseContact.SelectedIndex == 1)
        {   GridView1.DataSource =
address.ShowPersonalDetailBySelectedLetter();
            GridView1.DataBind();
```

```
        }
        else if (ddlContactType1.SelectedIndex == 2 &&
ddlBrowseContact.SelectedIndex == 1)
        {
            GridView1.DataSource =
address.ShowOfficialDetailBySelectedFirstNameLetter();
            GridView1.DataBind();
        }
        else if (ddlContactType1.SelectedIndex == 1 &&
ddlBrowseContact.SelectedIndex == 2)
        {
            GridView1.DataSource =
address.ShowPersonalDetailBySelectedLastNameLetter();
            GridView1.DataBind();
        }
        else if (ddlContactType1.SelectedIndex == 2 &&
ddlBrowseContact.SelectedIndex == 2)
        {
            GridView1.DataSource =
address.ShowOfficialDetailBySelectedLastNameLetter();
            GridView1.DataBind();
        }
         else if (ddlContactType1.SelectedIndex == 2 &&
ddlBrowseContact.SelectedIndex == 3)
        {
            GridView1.DataSource =
address.ShowOfficialDetailBySelectedCompanyNameLetter();
            GridView1.DataBind();
        }
    }
    protected void DataList1_ItemCommand(object source,
DataListCommandEventArgs e)
    {
        if (e.CommandName == "Letter")
        {
           address.LoginName = Session["UserName"].ToString();
           address.FirstName = e.CommandArgument.ToString();
           address.LastName = e.CommandArgument.ToString();
           address.CompanyName = e.CommandArgument.ToString();
           BindGridview1();
        }
    }
    protected void GridView1_RowCommand(object sender,
GridViewCommandEventArgs e)
    {
       HyperLink hyp;
        try
        {
            foreach (GridViewRow gr in GridView1.Rows)
            {
                if (gr.RowIndex == GridView1.SelectedIndex)
```

```
                {
                    hyp = (HyperLink)gr.FindControl("hyp1");
                }
            }
        }
        catch (Exception)
        {

            throw;
        }
    }

}
```

USERHOME.ASPX

```
using System;
using System.Data;
using System.Configuration;
using System.Collections;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

public partial class Registration_frmUserHomePage :
System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {

    }
    private void BindData()
    {

    }
    protected void lnkProfile_Click(object sender, EventArgs e)
    {

Response.Redirect("~/Registration/frmUpdateUserProfile.aspx");
    }
}
```

USERREGISTRATION.ASPX

```
using System;
using System.Data;
```

```
using System.Configuration;
using System.Collections;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

public partial class Registration_frmUserRegistration :
System.Web.UI.Page
{
    Country country = new Country();
    StateBL state = new StateBL();
    IntrestBL intrest = new IntrestBL();
    OccupationBL occupation = new OccupationBL();
    IncomeBL income = new IncomeBL();
    IndustryBL industry = new IndustryBL();
    UserRegistrationBL registration = new UserRegistrationBL();
    protected void Page_Load(object sender, EventArgs e)
    {
        GMDatePicker1.MaxDate = System.DateTime.Now;
        GMDatePicker1.MinDate = System.DateTime.Now.AddYears(-
100);
        BindRandomNumber();
        if (!IsPostBack)
        {

            BindData();
        }
        GMDatePicker1.Attributes.Add("readonly", "readonly()");
    }

    //-------------------------------------------------------------
------------------
    private void BindRandomNumber()
    {
        Random num = new Random();
        txtRandomNumber.Text = num.Next(500000).ToString();
    }
    //-------------------------------------------------------------
---------------
    private void BindData()
    {
        ddlCountry.DataSource = country.ShowCountry();
        ddlCountry.DataTextField = "CountryName";
        ddlCountry.DataBind();
        ddlCountry.Items.Insert(0, "Choose One...");

        ddlState.DataSource = state.ShowAllState();
        ddlState.DataTextField = "StateName";
```

```
        ddlState.DataBind();
        ddlState.Items.Insert(0, "Choose One...");

        chklistInrest.DataSource = intrest.ShowAllIntrest();
        chklistInrest.DataTextField = "Interest";
        chklistInrest.DataBind();


        ddlIncome.DataSource = income.ShowAllIncome();
        ddlIncome.DataTextField = "Income";
        ddlIncome.DataBind();
        ddlIncome.Items.Insert(0, "Choose One...");

        ddlIndustry.DataSource = industry.ShowAllIndustry();
        ddlIndustry.DataTextField = "IndustryType";
        ddlIndustry.DataBind();
        ddlIndustry.Items.Insert(0, "Choose One...");

        ddlOccupation.DataSource = occupation.ShowAllOccupation();
        ddlOccupation.DataTextField = "Occupation";
        ddlOccupation.DataBind();
        ddlOccupation.Items.Insert(0, "Choose One...");

    }


    protected void lnkAvailability_Click(object sender, EventArgs
e)
    {
        if (txtName.Text.Trim().Length < 1)
        {
            lblAvailability.Text = "Plz Enter User Name...!";
            txtName.Focus();
            return;

        }
        registration.LoginName = txtName.Text.Trim();
        if (registration.CheckUserAvailability() == true)
        {
            lblAvailability.Text = "User Name Already Exists...!";

        }

        else
        {
            lblAvailability.Text = "User Name Not Exists...!";
        }
    }

    protected void btnContinue_Click(object sender, EventArgs e)
    {
```

```
        try
        {
            registration.LoginName = txtName.Text.Trim();
            if (txtPassword.Text.Trim().Length < 6)
            {
                lblMsg.Focus();
                lblMsg.Text = "Password Should Have Atleast 6
Characters";
                return;
            }
            else
            {
                registration.Password = txtPassword.Text.Trim();
                registration.Question = txtQuestion.Text.Trim();
                registration.Answer = txtAnswer.Text.Trim();
                registration.FirstName = txtFName.Text.Trim();
                registration.LastName = txtLName.Text.Trim();
                registration.Address = txtAddress.Text.Trim();
                registration.City = txtCity.Text.Trim();
                registration.State =
ddlState.SelectedItem.Text.Trim();

                registration.PinCode = txtPinCode.Text.Trim();
                registration.Country =
ddlCountry.SelectedItem.Text.Trim();
                registration.Email = txtMail.Text.Trim();
                registration.Phone = txtPhone.Text.Trim();
                registration.DOB = GMDatePicker1.Date;
                registration.Gender =
ddlGender.SelectedItem.Text.Trim();
                if (ddlLanguage.SelectedIndex == 0)
                    registration.Language = "";
                else
                registration.Language =
ddlLanguage.SelectedItem.Text.Trim();
            if (ddlIncome.SelectedIndex == 0)
                registration.Income = "";
            else
                registration.Income =
ddlIncome.SelectedItem.Text.Trim();
            if (ddlOccupation.SelectedIndex == 0)
                registration.Occupation = "";
            else
                registration.Occupation =
ddlOccupation.SelectedItem.Text.Trim();
            if (ddlIndustry.SelectedIndex == 0)
                registration.IndustryType = "";
            else
                registration.IndustryType =
ddlIndustry.SelectedItem.Text.Trim();
                string Intrest = "";
```

```
                for (int i = 0; i < chklistInrest.Items.Count;
i++)
                {
                    if (chklistInrest.Items[i].Selected == true)
                        Intrest = Intrest +
chklistInrest.Items[i].Text + ",";


                }
                Intrest = Intrest.Remove(Intrest.Length - 1, 1);
                registration.Interest = Intrest;
                registration.Date = System.DateTime.Now.Date;
                if (txtNumber.Text.Trim().Length < 1)
                {
                    lblMsg.Text = "Enter No. to Match...!";
                    return;
                }
                registration.InsertUserLoginInfo();
                registration.InsertRegistrationInfo();
                Session["UserName"] = registration.LoginName;

Response.Redirect("~/Registration/frmSignUpSuccessful.aspx");

            }

        }
        catch (Exception ex)
        {
            lblMsg.Text = "User Name Already Exists...!";

        }


    }
}
```

CHAPTER 11

Testing

11.0 Testing

Testing is the filter to catch defects before they are "discovered" by the customer or end user. Every time a customer runs the program, he/she generates a "test-case". We tried the test cases to find the defects first since software development is a human activity and there may be potential errors. So testing before the product is delivered helped us to assure the quality and saved resources.

11.1 Testing Objective

Objective of testing is to find errors, demonstrate that software functions that satisfy the specification, reduce the number of errors detected by customer, have "confidence" in the software system. A good test always tries to discover undetected errors and is successful when errors are found since zero defects is not possible. There is always a condition or usage that can lead to an incorrect behavior.

11.2 Testing Steps

We started testing each individual new component and worked out as unit test, integration test, high order test, customer acceptance testing and different testing techniques are used at different times in the process.

11.3 Testing Techniques

Following testing techniques are used.

11.3.1 White Box Testing

In white box testing we exercises the internal logic of a program, examine internal program structure and derived tests from an examination of the program's logic, tested internal paths and working of the software. To perform this we used to develop test cases for unit and integration testing.

11.3.2 Black Box Testing

In black box testing we exercises the input and output requirements of a program, tested by conducting specifications of requirement for which the software should do. This test is derived from the I/O specification and used in most functional tests.

11.4 White Box Strategies

White box testing strategies uses the control structure of the program and design to derive test cases.

11.4.1 Basis Path Testing

Test cases are derived to exercise the basis set that guarantee that every statement in the program is executed at least once during testing. It is applied in series of steps.

Test case for login button of the login form:

Test case 1: All the following combinations are tested

Password is valid and Username is valid

Password is valid and Username is invalid

Password is invalid and Username is invalid

Password is invalid and Username is valid

Expected results: No error crossing this stage when Password & Username are both permissible combination of characters. In all other case error message is displayed.

Test case 2: Following input combination is tried where permissible set of characters for username & password are entered which are non-existent in table.

Password is valid, existent and Username is valid, existent

Password is valid, existent and Username is valid, nonexistent

Password is valid, nonexistent and Username is valid, existent

Password is valid, nonexistent and Username is valid, nonexistent

Expected Results: if no entry into the system then error message displayed. And user is linked to the registration page.

Test Case 3: To test this both username & password to be entered must be valid & existent in the registration table.

Password = Valid, Existent and Username = Valid, Existent

Similarly all the other internal paths & logic can be tested.

11.4.2 Control Structure Testing

Condition testing

Condition testing is a test case design method that exercises the logical conditions contained in a program module. If a condition is incorrect, then at least one of the conditions is incorrect. So, types of errors in a condition include Boolean operator error (incorrect/missing/extra Boolean operators), Boolean variable error, Boolean parenthesis error, relational operator error and arithmetic expression error. It mainly focuses on testing each condition in the program. The condition testing strategies to be employed are:

Branch testing: For a compound condition C, the true and false branches of C and every simple condition in C need to be executed at least once.

Domain testing: It requires three or four tests to be derived for a relational expression. For a relational expression of the form

E1 <relational operator> E2

Three tests are required to make the value of E1 greater than, equal to, or less than that of E2.

Boolean Expression testing: A condition without relational expression with n variables, all of 2^n possible tests are required provided n > 0. This strategy can detect boolean operator, variable, and parenthesis errors, but it is practical only if n is small.

Loop testing

Loops are cornerstone for the vast majority of all algorithms implemented in software. Loop testing focuses exclusively on the validity of loop constructs. Testing approach for different types of loops is listed below.

Simple loops: The following set of tests will be applied to simple loops, where n is the maximum number of allowable passes through the loop.

- Skip the loop entirely
- Only one pass through the loop
- Two passes through the loop
- m passes through the loop where m < n
- n – 1, n, n + 1 passes through the loop.

Nested loops: Simple loop testing strategy if applied to the nested loops will increase the no. of possible tests geometrically. Approach for nested loops are:

- Start at the innermost loop. Set all other loops to minimum values
- Conduct simple loop tests for the innermost loop while holding the outer loops at their minimum iteration parameter values. Add other tests for out-of-range or excluded values.
- Work outward, conducting tests for the next loop, but keep all other outer loops at minimum values and other nested loops to typical values.
- Continue till all loops have been tested.

Concatenated loops: When concatenated loops are independent then approach for simple loop was applied & if they happen to be dependent the nested loop approach was applied.

Unstructured loops: This class of loops was redesigned to reflect the use of the structured programming constructs.

11.5 Black Box Strategies

Black Box testing strategy focuses on the functional requirements of the software without any regard to the internal structure. It is used in most system level testing where functional, performance, recovery and security & stress are main issues. This applied strategy finds errors in incorrect or missing functions (compared to white box), interface errors, errors in external data structures and in behavior performance problems (Combinations of input make it behave poorly). It also finds errors in initialization and termination (Sensitive to certain inputs (e.g. performance). It is performed much later in process than white box…

A test strategy that uses every possible input condition as a test case would be ideal but is not possible. We apply following Black-Box Testing Strategies.

11.5.1 Graph Based Testing

The first step in black box testing is to understand the objects that are modeled in software and the relationships that connect the objects. Once this has been accomplished, the next step is to define a series of tests that verify all objects that have the expected relationship with another.

11.5.2 Equivalence Partitioning

It is a black-box testing method that divides the input domain of a program into classes of data from which test cases can be derived. It reduces, by more than one, the number of test cases that must be developed since one test case might uncover a class of errors. It covers a large set of other possible test cases hence helps reduce the number of inputs. Incorrect processing of all integer number is detected.

Identifying Equivalence Classes: Each input condition (a sentence or phrase in the specification) is taken & divided into 2 or more classes as valid equivalence class and invalid equivalence class.

Rules for Equivalence Classes:

Range - If an input condition specifies a range (i.e. n is an integer from 1 to 10) then

i) 1 valid (if 1<= n and <= 1000)

ii) 2 invalid (if n < 1 and > 1000)

Specified Value - An input condition specifies a specific value e.g. 10 character string then

i) 1 valid (10 character string)

ii) 2 invalid (any character string less than 10)

Value Set - If the input specifies a set of valid values, then

i) 1 valid condition within the set.

ii) 1 invalid condition outside the set.

Boolean - If an input condition specifies a "must be" situation (e.g. either Yes or No) then

i) 1 valid (if Yes).

ii) 1 invalid (else No).

11.5.3 Boundary Value Analysis

It is a test case design technique that complements equivalence partitioning, by selecting test cases at the "edges" of the class. Greater no. Of errors tends to occur at the boundaries of the input domain rather than in the "Center". Rather than *any* element in class, BVA selects tests at the *edge* of the class. In addition to input condition, test cases can be derived for output conditions. Similar to Equivalence partitioning, BVA, equivalence classes are identified first then boundaries.

Some guidelines for BVA:

1. If an input condition specifies a range bounded by values a and b then test cases will be designed with values a and b and just above and just below a and b.

2. If an input condition specifies a number of values, test cases will be developed that exercise the minimum and maximum numbers. Values just above and just below minimum and maximum are also tested.

These 2 guidelines are also applied to the output conditions. Designing test case to exercise the data structure at its boundary will test boundaries prescribed on internal program data structures.

11.5.4TESTING STRATEGIES FOR WEB APPLICATION

The approach of web app testing adopts the basic principle for all Software testing and applies a strategy and tactics that have been Recommended for object oriented system .the following steps Summarizes the approach:

1. **The content model for the web app is reviewed to uncover errors.**

This is like copyediting

2. **The design model for the web application as reviewed to uncover navigation error**.

Use cases derived as part of the analyst activity, allow a web engineer to exercise each usage scenario against the architectural and navigational design. These non executable test help uncover error in navigation.

3. **Selected processing component and web page is unit tested**.

When web apps are considered, the concepts of the unit changes, Each web page encapsulated in itself content navigation link as Well As script, form and applet (processing element). It is not always Possible or practical to test each of these characteristics individually.

4.**The architecture is constructed and integration tests are Conducted**.

The strategy for integration testing depend upon the architecture that has been chosen for the web application

5.The assembled web application is tested for overall functionality and content delivery.

Like conventional validation, the validation of web based systems and application focuses on user visible action and user recognizable output from the system. To assist in the derivation of validation tests the tester should draw upon use cases the use cases provides a scenario that has high likelihood of uncovering errors in user interaction requirement.

6**. The** web application is implemented in a variety of different Environmental configuration and is tested for compatibility with each configuration.

7. **The web application is tested by controlled and monitored Population of end user.**

Finally with the modular concept inside the application it is being also tested for its Reliability:

The system reliability will be insured through data integrity rules built into the database at the backend and the system rules built into the front-end application. The system will take assurance from the user before making any changes permanent.

MAINTAINABILITY:

The system has been designed taking care of modularity. Faults in the system can be traced to modules and thus

VALIDATION CHECKS

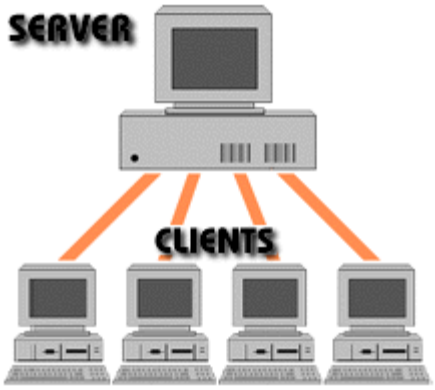
This will be as such to maintain consistent and persistent Information on the web when most of the time the project has to Deal With uploads so a minor error will down the impression of the Company. Therefore, validation checks by software itself using the Flavors of JavaScript and manual checks are also necessary as: -

1.Correct entry of data in the form.

2.Correct updating of question and with the most suitable answer etc.

CHAPTER 12

Implementation

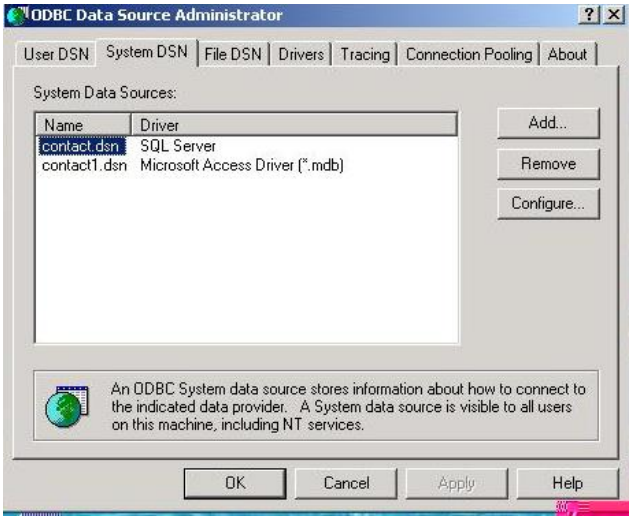


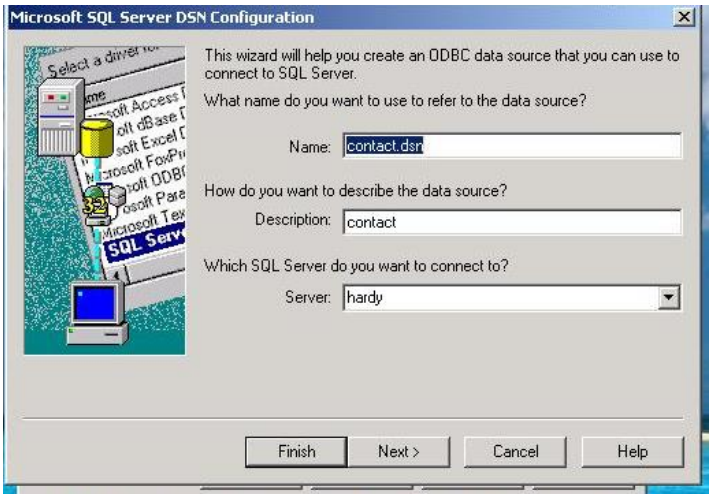
12.1 Implementation at the development site

All the clients are connected to the server in client server mode as intranet is fully functional as the server loaded with IIS is their at the development center with SQL server database so only the client has to enter the name of the local host and the page in the browser he will be getting the page he needs. All the pages, which are developed, are being kept in the root directory Of the server from were a virtual path is assigned to run The codes.

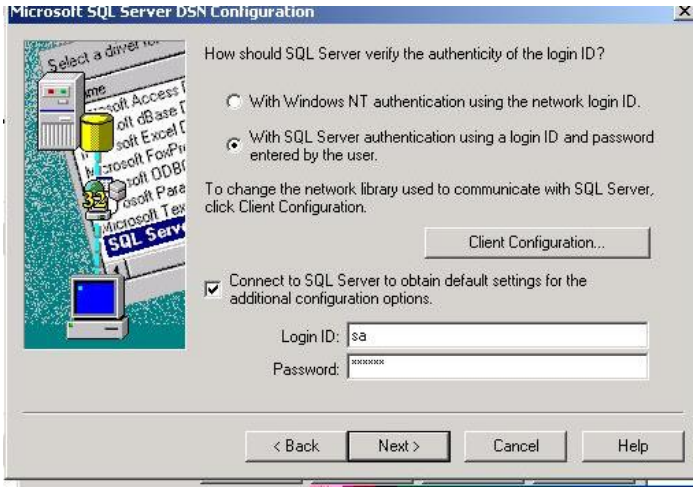
Since we use the D.S.N connection so a connection was Established between the server and the SQL server 7 .In the Following way-

12.2. CREATION OF SYSTEM DSN NAME

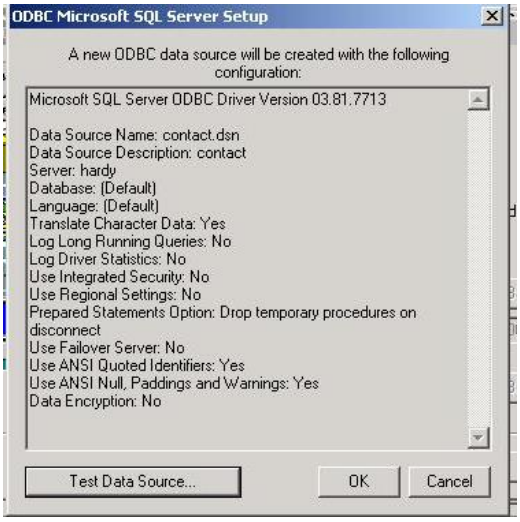




12.1 CONNECTING TO SQLSERVER



12.2 CONFIGURING THE CONNECTION



12.3 TESTING THE CONNECTION

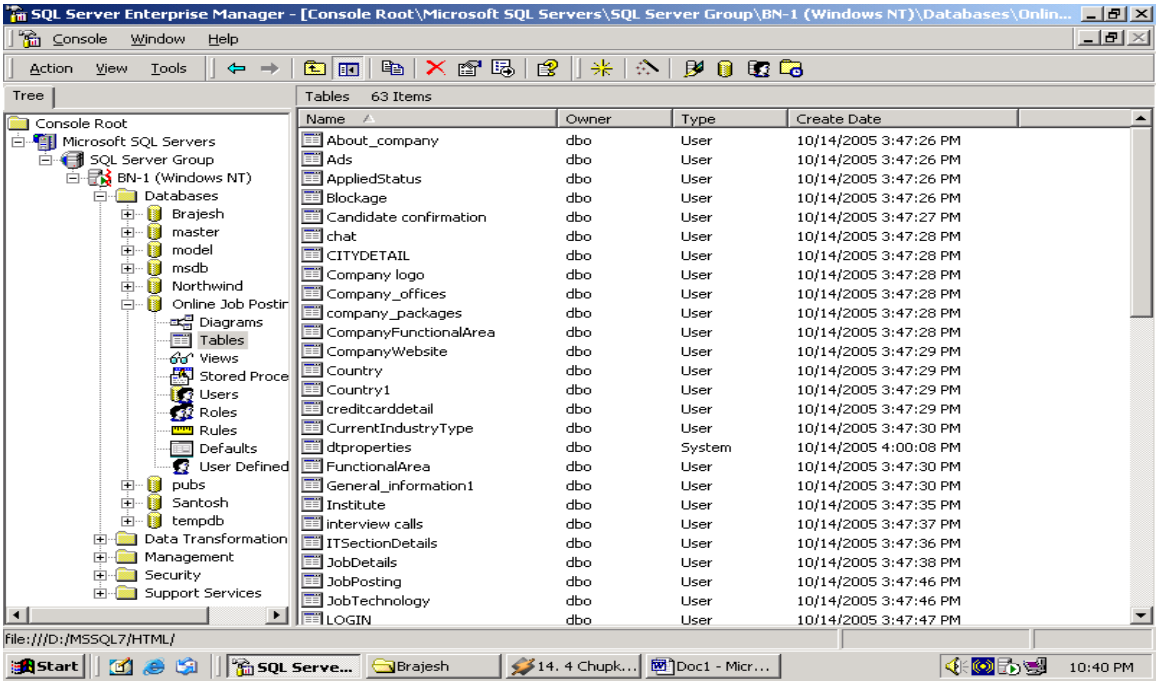
Implementation on the remote server

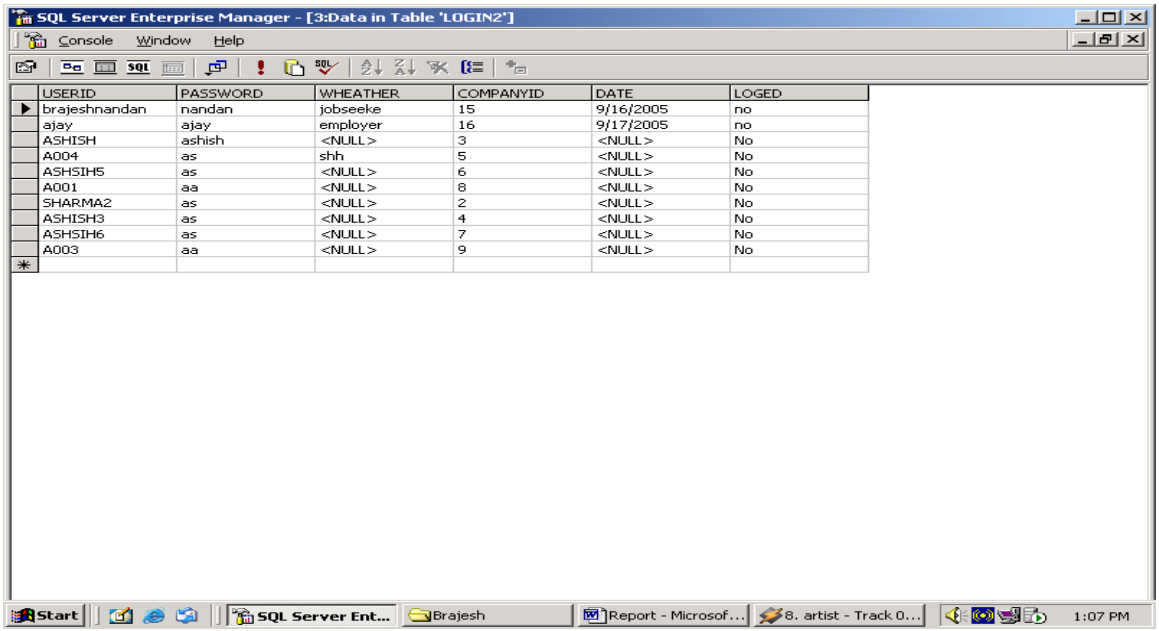
Since the site is under development so as and when we will be getting a domain and SQL server space at reliable cost the data will be transferred at that site the situation may come that we need to define the connection string again may be by server .map path as we are fully efficient to change the connection file as the necessities arises

CHAPTER 13

Database Design

Database, which is used in EMAIL SERVICE AUTOMATION SYSTEM: -





CHAPTER 14

Maintenance

14.1 Introduction

Software maintenance is a set of software engineering activities that occur after software has been delivered to the customer and put into operation. A primary goal of software engineering is to improve the ease with which changes can be made. The software maintenance can be defined by four activities:

- Corrective Maintenance
- Adaptive Maintenance
- Perceptive Maintenance or Enhancement
- Preventive Maintenance or Reengineering

The software documentation is very important job regarding maintenance, because it is a big problem for the user to make any change in working version of software without consent of the developer.

In out context as we developed and designed the core software for a website having the maximum possibility of change we have kept the copy of documentation on the site .As well as are always ready to edit it when any major change is being done at the design and coding part.

CHAPTER 15

Future Scope

- This System being web-based and an undertaking of Cyber Security Division, needs to be thoroughly tested to find out any security gaps.

- A console for the data center may be made available to allow the personnel to monitor on the sites which were cleared for hosting during a particular period.

- Moreover, it is just a beginning; further the system may be utilized in various other types of auditing operation viz. Network auditing or similar process/workflow based applications...

15.1 Software scope

Software scope describes the data and control to be processed, function performance, constraints, interfaces and reliability. Function describes in the statement of scope are evaluated and in some case refined to provide more detail prior to the beginning of the estimation. Because both cost and schedule estimates are functionally oriented, some degree of decomposition is often useful.

We can implement easily this application. Reusability is possible as and when we require in this application. We can update it next version. We can add new features as and when we require. There is flexibility in all the modules. Scope of this document is to put down the requirements, clearly identifying the information needed by the user, the source of the information and outputs expected from the system.

15.2 Future scope

It is directly dependent on the lay stone of the project that is we will have to design a system which when the time passes having a better system initially should not become a joke later.

It is highly likely that the scope will change as the web application project moves forward; the webE process model should be incremental. This allows the development team to “freeze” the scope for one increment so that an operational web application release can be created. The next increment may scope changes suggested by a review of the preceding increment, but once the second increment commences, scope is again frozen temporarily. This approach enables the WebApp team to work without having to accommodate a continual stream of changes but still recognizes the continuous evolution characteristics of most web application. Besides that, the following basic quality in the software always safeguards the future scope of the software.

Reusability : Reusability is possible as and when we require in this application. We can update it next version. Reusable software reduces design, coding and testing cost by amortizing effort over several designs. Reducing the amount of code also simplifies understanding, which increases the likelihood that the code is correct. We follow up

both types of reusability: Sharing of newly written code within a project and reuse of previously written code on new projects.

Extensibility: This software is extended in ways that its original developers may not expect. The following principles enhance extensibility like Hide data structure, Avoid traversing multiple links or methods, Avoid case statements on object type and distinguish public and private operations.

Robustness: Its method is robust if it does not fail even if it receives improper parameters. There is some facilities like Protect against errors, Optimize after the program runs, validate arguments and Avoid predefined limits.

Understandability: A method is understandable if someone other than the creator of the method can understand the code (as well as the creator after a time lapse). We use the method with small and coherent helps to accomplish this.

Cost-effectiveness: Its cost is under the budget and make within given time period. It is desirable to aim for a system with a minimum cost subject to the condition that it must satisfy all the requirements.

Can be rectified easily. The entire source code is well structured and commented to ensure clarity and readability.

PORTABILITY: since it is an Internet based application so its portability and usability depends upon the Clint connected with the Internet. The interface designed that is the web page designing which is one of the major part of web application because it is the first impression regardless of the value of its contents interface should must grab a potential user immediately

CHAPTER 16

Bibliography

References:

- ASP Programming using Oracle
- ASP Black Book**, Techmedia**
- Teach yourself A.S.P **Scott Mitcheel ,James Atkinson**
- ASP 6 Database programming **wrox**
- Microsoft sql server 7 by **NIIT**
- Data Communication & Computer Networks by Andrew S. Tannenbaum
- www.ASP101.com
- www.4guysfromrolla.com
- sqlonlineforum.com
- www.sqlguru.com

APPENDIX A

Acronyms

- TCP – Transmission Control Protocol
- BCNF – Boyce Codd Normal Form
- TE A – The Education For All

- DBA – Database Administrator
- DFD – Data Flow Diagram
- DOB – Date of Birth
- DOM – Date of Marriage
- FD – Functional Dependencies
- IP – Internet Protocol
- LAN -- Local Area Network
- NF-- Normal Form
- PV – Present Value
- RDBMS—Relational Database Management System
- SCM – Software Configuration Management
- SDLC—Software Development Life Cycle
- SRS -- Software Requirement Specification

APPENDIX B

1.5 References

- IEEE Std-830, IEEE New York, 1984 and 1993.
- Software Engineering, A Practitioner's Approach by ***Roger S Pressman***
- Teach yourself A.S.P **Scott Mitcheel ,James Atkinson**
- ASP 6 Database programming **wrox**
- An Introduction To Database Systems by ***Bipin C Desai***
- Object-Oriented Modeling and Design by ***James Rambagh, Michael Blaha, Willium Premerlani, Freedrick Eddy, William Lorensen***