**A PROJECT REPORT ON**

**Automated Phrase Mining from Massive Text Corpora**

**Submitted to \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* University for the partial fulfillment of the Requirement for the**

**Award of Degree for**

**\*\*\*\*\*\*\*\*\*\*\*Course Name\*\*\*\*\*\*\*\*\*\*\*\*\***

**Done By**

**Mr. /Miss \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*Institute of Management and Computer Science**

**CERTIFICATE**

**This is to certify that Mr., /Miss \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* bearing Roll No. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* have developed software project Titled \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* For \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Software Solutions as Partial fulfillment for the award of the Degree of \*\*\*\*\*\*\*\*\*\*\*\*\*\***

**Head of the Department Principal**

**\*\*\*\*\*\*College Name\*\*\*\*\*\***

**External**

**ACKNOWLEDGEMENT**

At every outset I express my gratitude to almighty lord for showering his grace and blessings upon me to complete this project.

Although our name appears on the cover of this book, many people had contributed in some form or the other form to this project Development. We could not done this project without the assistance or support of each of the following we thank you all.

I wish to place on my record my deep sense of gratitude to my project guide, **Mr. \*\*\*\*\*\*, \*\*\*\*\* Software Solutions,** for his constant motivation and valuable help through the project work. Express my gratitude to **Mr. \*\*\*\*\*\***, Director of \*\*\*\*\*\*\* **Institute of Management & Computer Sciences** for his valuable suggestions and advices throughout the \*\*\*\*\* course. I also extend my thanks to other Faculties for their Cooperation during my Course.

Finally I would like to thank my friends for their cooperation to complete this project.

\*\*\*\*\*\*\*Your Name\*\*\*\*\*\*\*\*\*

**ABSTRACT**

**Abstract:**

As one of the fundamental tasks in text analysis, phrase mining aims at extracting quality phrases from a text corpus and has various downstream applications including information extraction/retrieval, taxonomy construction, and topic modeling. Most existing methods rely on complex, trained linguistic analyzers, and thus likely have unsatisfactory performance on text corpora of new domains and genres without extra but expensive adaption. None of the state-of-the-art models, even data-driven models, is fully automated because they require human experts for designing rules or labeling phrases. In this paper, we propose a novel framework for automated phrase mining, Auto Phrase, which supports any language as long as a general knowledge base (e.g., Wikipedia) in that language is available, while benefiting from, but not requiring, a POS tagger. Compared to the state-of-the-art methods, Auto Phrase has shown significant improvements in both effectiveness and efficiency on five real-world datasets across different domains and languages. Besides, Auto Phrase can be extend to model single-word quality phrases.

**LIST OF CONTENTS**

Page Nos.

1. Introduction

1.1 Purpose

1.2 Scope and Objective

2. Literature Survey

3. Fundamental Concepts on (Domain)

4. System Analysis

4.1 Existing System

4.1.1 Drawbacks

4.2 Problem statement

4.3 Proposed System

4.3.1 Advantages

4.4 Modules Description

4.5 Feasibility Study

4.5.1 Economic Feasibility

4.5.2 Operational Feasibility

4.5.3 Technical Feasibility

5. System Requirements Specification

5.1 Introduction

5.2 Purpose

5.3 Functional Requirements

5.4 Non Functional Requirements

5.5 Input & Output Design

5.6 Hardware Requirements

5.6 Software Requirements

6. System Design

6.1 System Specifications

6.2 System Components

6.3 DFD’s

6.4 UML Diagrams

6.5 Data Dictionaries and ER Diagram

7. Implementation

7.1 Technology Description

8. Coding

9. System Testing

9.1 Testing Methodologies

9.2 Test cases

9.3 Result Analysis

10. Output Screens

11. Reports

12. Conclusion

13. Future Enhancements

14. References

15. Bibliography

**Introduction**

Phrase mining refers to the process of automatic extraction of high-quality phrases (e.g., scientific terms and general entity names) in a given corpus (e.g., research papers and news). Representing the text with quality phrases instead of n-grams can improve computational models

for applications such as information extraction/retrieval, taxonomy construction, and topic modeling [21], [10],[19].Almost all the state-of-the-art methods, however, require human experts at certain levels. Most existing methods [13], [31], [38] rely on complex, trained linguistic analyzers (e.g., dependency parsers) to locate phrase mentions, and thus may have unsatisfactory performance on text corpora of new domains and genres without extra but expensive adaption. Our latest domain-independent method SegPhrase [23] outperforms many other approaches [1], [13], [31], [38], [9], [30], [10], but still needs domain experts to first carefully select hundreds of varyingquality phrases from millions of candidates, and then annotate them with binary labels. Such reliance on manual efforts by domain and linguistic experts becomes an impediment for timely analysis of massive, emerging text corpora in specific domains. An ideal automated phrase mining method is supposed to be domain-independent, with minimal human effort 1 or reliance on linguistic analyzers. Bearing this in mind, we propose a novel automated phrase mining framework AutoPhrase in this paper, going beyond SegPhrase, to further avoid additional manual labeling effort and enhance the performance, mainly using the following two new techniques.

1) Robust Positive-Only Distant Training. In fact, many high-quality phrases are freely available in general knowledge bases, and they can be easily obtained to a scale that is much larger than that produced by human experts. Domain-specific corpora usually contain some quality phrases also encoded in general knowledge bases, even when there may be no other domain-specific knowledge bases. Therefore, for distant training, we leverage the existing highquality phrases, as available from general knowledge bases, such as Wikipedia and Freebase, to get rid of additional manual labeling effort. We independently build samples of positive labels from general knowledge bases and negative labels from the given domain corpora, and train a number of base classifiers. We then aggregate the predictions from these classifiers, whose independence helps reduce the noise from negative labels.

2) POS-Guided Phrasal Segmentation. There is a trade-off

between the accuracy and domain-independence when incorporating linguistic processors in the phrase mining method.

On the domain independence side, the accuracy might be limited without linguistic knowledge.It is difficult to support multiple languages well,if the method is completely language-blind.

On the accuracy side, relying on complex,trained linguistic analyzers may hurt the domain-independence of the phrase mining method. For example, it is expensive to adaptdependency parsers to special domains like clinical reports.As a compromise, we propose to incorporate a pre-trained part-of-speech (POS) tagger to further enhance the performance, when it is available for the language of the document collection. The POSguided phrasal segmentation leverages the shallow syntactic information in POS tags to guide the phrasal segmentation model locating the boundaries of phrases more accurately.

In principle, AutoPhrase can support any language as long as a general knowledge base in that language is available. In fact, at least 58 languages have more than 100,000 articles in Wikipedia as of Feb, 20172.Moreover, since pre-trained part-of-speech (POS) taggers are widely available in many languages (e.g., more than 20 languages in TreeTagger [35]3), the POS-guided phrasal segmentation can be applied in many scenarios. It is worth mentioning that for domain-specific knowledge bases (e.g., MeSH terms in the biomedical domain) and trained POS taggers, the same paradigm applies. In this study, without loss of generality, we only assume the availability of a general knowledge base together with a pre-trained POS tagger.

As demonstrated in our experiments, AutoPhrase not only works effectively in multiple domains like scientific papers, business reviews, and Wikipedia articles, but also supports multiple languages, such as English, Spanish, and Chinese. In addition, AutoPhrase can be extended to model single-word phrases.

Our main contributions are highlighted as follows:

We study an important problem, automated phrase mining, and analyze its major challenges as above.

We propose a robust positive-only distant training method for phrase quality estimation to minimize the human effort.

We develop a novel phrasal segmentation model to leverage POS tags to achieve further improvement, when a POS tagger is available.

We demonstrate the robustness, accuracy, and efficiency of our method and show improvements

over prior methods, with results of experiments conducted on five real-world datasets in different domains (scientific papers, business reviews, and Wikipedia articles) and different languages (English,Spanish, and Chinese).

We successfully extend AutoPhrase to model singleword phrases, which brings about 10% to 30% recall improvements on different datasets.

**1.1 Purpose:**

1.2 **Scope and Objective:**

**Literature Survey:**

**RANDOMIZING OUTPUTS TO INCREASE PREDICTION ACCURACY**

Bagging and boosting reduce error by changing both the inputs and outputs to form perturbed training sets, grow predictors on these perturbed training sets and combine them. A question that has been frequently asked is whether it is possible to get comparable performance by perturbing the outputs alone. Two methods of randomizing outputs are experimented with. One is called output smearing and the other output flipping. Both are shown to consistently do better than bagging.In recent research in combining predictors, it has been recognized that the critical thing to success in combining low-bias predictors such as trees and neural nets has been through methods that reduce the variability in the predictor due to training set variability. Assume that the training set consists of N independent draws from the same underlying distribution.Conceptually, training sets of size N can be drawn repeatedly and the same algorithm used to construct a predictor on each training set. These predictors will vary, and the extent of the variability is a dominant factor in the generalization prediction error. Given a training set {(*yn* ,**x***n* ),*n*=1,...*N*} where the y's are either class labels or numerical values, the most common way of reducing variability is by perturbing the training set to produce alternative training sets, growing a predictor on each perturbed training set, and either averaging the outcomes (regression) or letting them vote for the most popular class (classification). Examples are bagging (Breiman[1996b]) and arcing-boosting However, in work (Breiman[1996b]) on subset selection in linear regression, I found the somewhat surprising result that adding noise to the y's while leaving the input **x**-vectors unchanged worked just as well as bagging. This paper extends those earlier results to non-linear contexts in both regression and classification and also tries to understand why perturbing outputs only works well. Although we use trees in the experimental results, the ideas have more general validity.

**Comparative Analysis of Viterbi Training and Maximum Likelihood Estimation for HMMs**

We present an asymptotic analysis of Viterbi Training (VT) and contrast it with a more conventionalMaximum Likelihood (ML) approach to parameter estimation in Hidden Markov Models. While ML estimator works by (locally) maximizing the likelihood of the observed data, VT seeks to maximize the probability of the most likely hidden state sequence. We develop an analytical framework based on a generating function formalism and illustrate it on an exactly solvable model of HMM with one unambiguous symbol. For this particular model the ML objective function is continuously degenerate. VT objective, in contrast, is shown to have only finite degeneracy. Furthermore, VT converges faster and results in sparser (simpler) models, thus realizing an automatic Occam’s razor for HMM learning. For more general scenario VT can be worse compared to ML but still capable of correctly recovering most of the parameters.Hidden Markov Models (HMM) provide one of the simplest examples of structured data observed through a noisy channel. The inference problems of HMM naturally divide into two classes [20, 9]:

recovering the hidden sequence of states given the observed sequence, and *ii)* estimating themodel parameters (transition probabilities of the hidden Markov chain and/or conditional probabilities of observations) from the observed sequence. The first class of problems is usually solved via the maximum a posteriori (MAP) method and its computational implementation known as Viterbi algorithm [20, 9]. For the parameter estimation problem, the prevailing method is maximum likelihood (ML) estimation, which finds the parameters by maximizing the likelihood of the observed data. Since global optimization is generally intractable, in practice it is implemented through an expectation–maximization (EM) procedure known as Baum–Welch algorithm [20, 9].An alternative approach to parameter learning is Viterbi Training (VT), also known in the literature as segmental K-means, Baum–Viterbi algorithm, classification EM, hard EM, etc. Instead of maximizing the likelihood of the observed data, VT seeks to maximize the probability of the most likely hidden state sequence. Maximizing VT objective function is hard [8], so in practice it is implemented via an EM-style iterations between calculating the MAP sequence and adjusting the model parameters based on the sequence statistics. It is known that VT lacks some of the desired features of ML estimation such as consistency [17], and in fact, can produce biased estimates [9]. However, it has been shown to perform well in practice, which explains its widespread use in applications such as speech recognition [16], unsupervised dependency parsing [24], grammar induction [6], ion channel modeling [19]. It is generally assumed that VT is more robust and faster but usually less accurate, although for certain tasks it outperforms conventional EM [24]. The current understanding of when and under what circumstances one method should be preferred over the other is not well–established. For HMMs with continuos observations, Ref. [18] established 1 an upper bound on the difference between theML and VT objective functions, and showed that both approaches produce asymptotically similar estimates when the dimensionality of the observation space is very large. Note, however, that this asymptotic limit is not very interesting as it makes the structure imposed by the Markovian process irrelevant. A similar attempt to compare both approaches on discrete models (for stochastic context free grammars) was presented in [23]. However, the established bound was very loose.

**A Nonparametric Method for Extraction of Candidate Phrasal Terms**

This paper introduces a new method for identifying candidate phrasal terms (also known as multiword units) which applies a nonparametric, rank-based heuristic measure. Evaluation of this measure, the *mutual rank ratio* metric, shows that it produces better results than standard statistical measures when applied to this task. The ordinary vocabulary of a language like English contains thousands of *phrasal terms* -- multiword lexical units including compound nouns, technical terms, idioms, and fixed collocations. The exact number of phrasal terms is difficult to determine, as new ones are coined regularly, and it is sometimes difficult to determine whether a phrase is a fixed term or a regular, compositional expression. Accurate identification of phrasal terms is important in a variety of contexts, including natural language parsing, question answering systems, information retrieval systems, among others.Insofar as phrasal terms function as lexical units, their component words tend to cooccur more often, to resist substitution or paraphrase, to follow fixed syntactic patterns, and to display some degree of semantic noncompositionality (Manning, 1999:183-186). However, none of these characteristics are amenable to a simple algorithmic interpretation. It is true that various term extraction systems have been developed, such as Xtract (Smadja 1993), Termight (Dagan & Church 1994), and TERMS (Justeson & Katz 1995) among others (cf. Daille 1996, Jacquemin & Tzoukermann 1994, Jacquemin, Klavans, & Toukermann 1997, Boguraev & Kennedy 1999, Lin 2001). Such systems typically rely on a combination of linguistic knowledge and statistical association measures. Grammatical patterns, such as adjective-noun or noun-noun sequences are selected then ranked statistically, and the resulting ranked list is either used directly or submitted for manual filtering.Generating Typed Dependency Parses from Phrase Structure Parses This paper describes a system for extracting typed dependency parses of English sentences from phrase structure parses. In order to capture inherent relations occurring in corpus texts that can be critical in real-world applications, many NP relations are included in the set of grammatical relations used. We provide a comparison of our system with Minipar and the Link parser. The typed dependency extraction facility described here is integrated in the Stanford Parser, available for download. We describe a system for automatically extracting typed dependency parses of English sentences from phrase structure parses. Typed dependencies and phrase structures are different ways of representing the structure of sentences: while a phrase structure parse represents nesting of multi-word constituents, a dependency parse represents dependencies between individual words. A typed dependency parse additionally labels dependencies with grammatical relations, such as subject or indirect object. There has been much linguistic discussion of the two formalisms. There are formal isomorphisms between certain structures, such as between dependency grammars and one bar-level, headed phrase structure grammars (Miller, 2000). In more complex theories there is significant debate: dominant Chomskyan theories (Chomsky, 1981) have defined grammatical relations as configurations at phrase structure, while other theories such as Lexical-Functional Grammar has rejected the adequacy of such an approach (Bresnan, 2001). Our goals here are more practical, though in essence we are following an approach where structural configurations are used to define grammatical roles.

**Extracting Noun Phrases from Large-Scale Texts: A Hybrid Approach and Its Automatic Evaluation**

To acquire noun phrases from running texts is useful for many applications, such as word grouping, terminology indexing, *etc.* The reported literatures adopt pure probabilistic approach, or pure rule-based noun phrases grammar to tackle this problem. In this paper, we apply a probabilistic chunker to deciding the implicit boundaries of constituents and utilize the linguistic knowledge to extract the noun phrases by a finite state mechanism. The test texts are SUSANNE Corpus and the results are evaluated by comparing the parse field of SUSANNE Corpus automatically. The results of this preliminary experiment are encouraging.From the cognitive point of view, human being must recognize, learn and understand the entities or concepts(concrete or abstract) in the texts for natural language comprehension. These entities or concepts are usually described by noun phrases. The evidences from the language learning of children also show the belief (Snowand Ferguson, 1977). Therefore, if we can grasp the noun phases of the texts, we will understand the texts to some extent. This consideration is also captured by theories of discourse analysis, such as Discourse Representation Theory (Kamp, 1981).

**Noun-Phrase Analysis in Unrestricted Text for Information Retrieval:**

Information retrieval is an important application area of natural-language processing where one encounters the genuine challenge of processing large quantities of unrestricted natural-language text. This paper reports on the application of a few simple, yet robust and efficient nounphrase analysis techniques to create better indexing phrases for information retrieval.

In particular, we describe a hybrid approach to the extraction of meaningful (continuous or discontinuous) subcompounds from complex noun phrases using both corpus statistics and linguistic heuristics. Results of experiments show that indexing based on such extracted subcompounds improves both recall and precision in an information retrieval system. The noun-phrase analysis techniques are also potentially useful for book indexing and automatic thesaurus extraction. Information retrieval (IR) is an important application area of naturaManguage processing (NLP). 1 The IR (or perhaps more accurately "text retrieval") task may be characterized as the problem of selecting a subset of documents (from a document collection) whose content is relevant to the information need of a user as expressed by a query. The document collections involved in IR are often gigabytes of unrestricted natural-language text. A user's query may be expressed in a controlled language (e.g., a boolean expression of keywords) or, more desirably, a natural language, such as English.A typical IR system works as follows. The documentsto be retrieved are processed to extract *indexing terms* or *content carriers,* which are usually single words or (less typically) phrases. The indexing terms provide a description of the document's content. Weights are often assigned to terms to indicate how well they describe the document. A (natural-language) query is processed in a similar way to extract *query terms.* Query terms are then matched against the indexing terms of a document to determine the relevance of each document to the quer3a

Fundamental Concepts on (Domain)

## 1 Data mining

### 1.1 What is data mining?

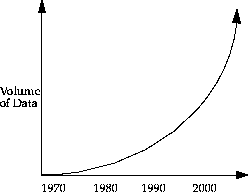
The past two decades has seen a dramatic increase in the amount of information or data being stored in electronic format. This accumulation of data has taken place at an explosive rate. It has been estimated that the amount of information in the world doubles every 20 months and the size and number of databases are increasing even faster. The increase in use of electronic data gathering devices such as point-of-sale or remote sensing devices has contributed to this explosion of available data. Figure 1 from the Red Brick company illustrates the data explosion.  


Figure 1: The Growing Base of Data

Data storage became easier as the availability of large amounts of computing power at low cost ie the cost of processing power and storage is falling, made data cheap. There was also the introduction of new machine learning methods for knowledge representation based on logic programming etc. in addition to traditional statistical analysis of data. The new methods tend to be computationally intensive hence a demand for more processing power.

Having concentrated so much attention on the accumulation of data the problem was what to do with this valuable resource? It was recognised that information is at the heart of business operations and that decision-makers could make use of the data stored to gain valuable insight into the business. Database Management systems gave access to the data stored but this was only a small part of what could be gained from the data. Traditional on-line transaction processing systems, OLTPs, are good at putting data into databases quickly, safely and efficiently but are not good at delivering meaningful analysis in return. Analysing data can provide further knowledge about a business by going beyond the data explicitly stored to derive knowledge about the business. This is where Data Mining or Knowledge Discovery in Databases (KDD) has obvious benefits for any enterprise.

The term data mining has been stretched beyond its limits to apply to any form of data analysis. Some of the numerous definitions of Data Mining, or Knowledge Discovery in Databases are:

Data Mining, or Knowledge Discovery in Databases (KDD) as it is also known, is the nontrivial extraction of implicit, previously unknown, and potentially useful information from data. This encompasses a number of different technical approaches, such as clustering, data summarization, learning classification rules, finding dependency net works, analysing changes, and detecting anomalies.

William J Frawley, Gregory Piatetsky-Shapiro and Christopher J Matheus

Data mining is the search for relationships and global patterns that exist in large databases but are `hidden' among the vast amount of data, such as a relationship between patient data and their medical diagnosis. These relationships represent valuable knowledge about the database and the objects in the database and, if the database is a faithful mirror, of the real world registered by the database.

Marcel Holshemier & Arno Siebes (1994)

The analogy with the mining process is described as:

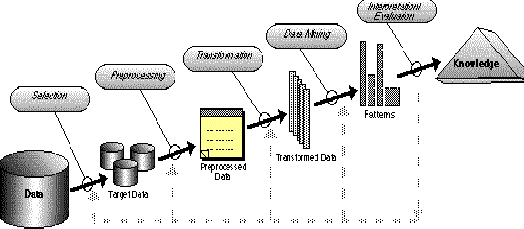
Data mining refers to "using a variety of techniques to identify nuggets of information or decision-making knowledge in bodies of data, and extracting these in such a way that they can be put to use in the areas such as decision support, prediction, forecasting and estimation. The data is often voluminous, but as it stands of low value as no direct use can be made of it; it is the hidden information in the data that is useful"

Clementine User Guide, a data mining toolkit

Basically data mining is concerned with the analysis of data and the use of software techniques for finding patterns and regularities in sets of data. It is the computer which is responsible for finding the patterns by identifying the underlying rules and features in the data. The idea is that it is possible to strike gold in unexpected places as the data mining software extracts patterns not previously discernable or so obvious that no-one has noticed them before.

Data mining analysis tends to work from the data up and the best techniques are those developed with an orientation towards large volumes of data, making use of as much of the collected data as possible to arrive at reliable conclusions and decisions. The analysis process starts with a set of data, uses a methodology to develop an optimal representation of the structure of the data during which time knowledge is acquired. Once knowledge has been acquired this can be extended to larger sets of data working on the assumption that the larger data set has a structure similar to the sample data. Again this is analogous to a mining operation where large amounts of low grade materials are sifted through in order to find something of value.

The following diagram summarises the some of the stages/processes identified in data mining and knowledge discovery by Usama Fayyad & Evangelos Simoudis, two of leading exponents of this area.



The phases depicted start with the raw data and finish with the extracted knowledge which was acquired as a result of the following stages:

* Selection - selecting or segmenting the data according to some criteria e.g. all those people who own a car, in this way subsets of the data can be determined.
* Preprocessing - this is the data cleansing stage where certain information is removed which is deemed unnecessary and may slow down queries for example unnecessary to note the sex of a patient when studying pregnancy. Also the data is reconfigured to ensure a consistent format as there is a possibility of inconsistent formats because the data is drawn from several sources e.g. sex may recorded as f or m and also as 1 or 0.
* Transformation - the data is not merely transferred across but transformed in that overlays may added such as the demographic overlays commonly used in market research. The data is made useable and navigable.
* Data mining - this stage is concerned with the extraction of patterns from the data. A pattern can be defined as given a set of facts(data) F, a language L, and some measure of certainty C a pattern is a statement S in L that describes relationships among a subset Fs of F with a certainty c such that S is simpler in some sense than the enumeration of all the facts in Fs.
* Interpretation and evaluation - the patterns identified by the system are interpreted into knowledge which can then be used to support human decision-making e.g. prediction and classification tasks, summarizing the contents of a database or explaining observed phenomena.

### 1.2 Data mining background

Data mining research has drawn on a number of other fields such as inductive learning, machine learning and statistics etc.

#### 1.2.1 Inductive learning

Induction is the inference of information from data and inductive learning is the model building process where the environment i.e. database is analysed with a view to finding patterns. Similar objects are grouped in classes and rules formulated whereby it is possible to predict the class of unseen objects. This process of classification identifies classes such that each class has a unique pattern of values which forms the class description. The nature of the environment is dynamic hence the model must be adaptive i.e. should be able learn.

Generally it is only possible to use a small number of properties to characterise objects so we make abstractions in that objects which satisfy the same subset of properties are mapped to the same internal representation.

Inductive learning where the system infers knowledge itself from observing its environment has two main strategies:

* supervised learning - this is learning from examples where a teacher helps the system construct a model by defining classes and supplying examples of each class. The system has to find a description of each class i.e. the common properties in the examples. Once the description has been formulated the description and the class form a classification rule which can be used to predict the class of previously unseen objects. This is similar to discriminate analysis as in statistics.
* unsupervised learning - this is learning from observation and discovery. The data mine system is supplied with objects but no classes are defined so it has to observe the examples and recognise patterns (i.e. class description) by itself. This system results in a set of class descriptions, one for each class discovered in the environment. Again this similar to cluster analysis as in statistics.

Induction is therefore the extraction of patterns. The quality of the model produced by inductive learning methods is such that the model could be used to predict the outcome of future situations in other words not only for states encountered but rather for unseen states that could occur. The problem is that most environments have different states, i.e. changes within, and it is not always possible to verify a model by checking it for all possible situations.

Given a set of examples the system can construct multiple models some of which will be simpler than others. The simpler models are more likely to be correct if we adhere to Ockhams razor, which states that if there are multiple explanations for a particular phenomena it makes sense to choose the simplest because it is more likely to capture the nature of the phenomenon.

#### 1.2.2 Statistics

Statistics has a solid theoretical foundation but the results from statistics can be overwhelming and difficult to interpret as they require user guidance as to where and how to analyse the data. Data mining however allows the expert's knowledge of the data and the advanced analysis techniques of the computer to work together.

Statistical analysis systems such as SAS and SPSS have been used by analysts to detect unusual patterns and explain patterns using statistical models such as linear models. Statistics have a role to play and data mining will not replace such analyses but rather they can act upon more directed analyses based on the results of data mining. For example statistical induction is something like the average rate of failure of machines.

#### 1.2.3 Machine Learning

Machine learning is the automation of a learning process and learning is tantamount to the construction of rules based on observations of environmental states and transitions. This is a broad field which includes not only learning from examples, but also reinforcement learning, learning with teacher, etc. A learning algorithm takes the data set and its accompanying information as input and returns a statement e.g. a concept representing the results of learning as output. Machine learning examines previous examples and their outcomes and learns how to reproduce these and make generalisations about new cases.

Generally a machine learning system does not use single observations of its environment but an entire finite set called the training set at once. This set contains examples i.e. observations coded in some machine readable form. The training set is finite hence not all concepts can be learned exactly.

#### 1.2.4 Differences between Data Mining and Machine Learning

Knowledge Discovery in Databases (KDD) or Data Mining, and the part of Machine Learning (ML) dealing with learning from examples overlap in the algorithms used and the problems addressed.

The main differences are:

* KDD is concerned with finding understandable knowledge, while ML is concerned with improving performance of an agent. So training a neural network to balance a pole is part of ML, but not of KDD. However, there are efforts to extract knowledge from neural networks which are very relevant for KDD.
* KDD is concerned with very large, real-world databases, while ML typically (but not always) looks at smaller data sets. So efficiency questions are much more important for KDD.
* ML is a broader field which includes not only learning from examples, but also reinforcement learning, learning with teacher, etc.

KDD is that part of ML which is concerned with finding understandable knowledge in large sets of real-world examples. When integrating machine learning techniques into database systems to implement KDD some of the databases require:

* more efficient learning algorithms because realistic databases are normally very large and noisy. It is usual that the database is often designed for purposes different from data mining and so properties or attributes that would simplify the learning task are not present nor can they be requested from the real world. Databases are usually contaminated by errors so the data mining algorithm has to cope with noise whereas ML has laboratory type examples i.e. as near perfect as possible.
* more expressive representations for both data, e.g. tuples in relational databases, which represent instances of a problem domain, and knowledge, e.g. rules in a rule-based system, which can be used to solve users' problems in the domain, and the semantic information contained in the relational schemata.

Practical KDD systems are expected to include three interconnected phases

* Translation of standard database information into a form suitable for use by learning facilities;
* Using machine learning techniques to produce knowledge bases from databases; and
* Interpreting the knowledge produced to solve users' problems and/or reduce data spaces. Data spaces being the number of examples.

### 1.3 Data Mining Models

IBM have identified two types of model or modes of operation which may be used to unearth information of interest to the user.

#### 1.3.1 Verification Model

The verification model takes an hypothesis from the user and tests the validity of it against the data. The emphasis is with the user who is responsible for formulating the hypothesis and issuing the query on the data to affirm or negate the hypothesis.

In a marketing division for example with a limited budget for a mailing campaign to launch a new product it is important to identify the section of the population most likely to buy the new product. The user formulates an hypothesis to identify potential customers and the characteristics they share. Historical data about customer purchase and demographic information can then be queried to reveal comparable purchases and the characteristics shared by those purchasers which in turn can be used to target a mailing campaign. The whole operation can be refined by `drilling down' so that the hypothesis reduces the `set' returned each time until the required limit is reached.

The problem with this model is the fact that no new information is created in the retrieval process but rather the queries will always return records to verify or negate the hypothesis. The search process here is iterative in that the output is reviewed, a new set of questions or hypothesis formulated to refine the search and the whole process repeated. The user is discovering the facts about the data using a variety of techniques such as queries, multidimensional analysis and visualization to guide the exploration of the data being inspected.

#### 1.3.2 Discovery Model

The discovery model differs in its emphasis in that it is the system automatically discovering important information hidden in the data. The data is sifted in search of frequently occurring patterns, trends and generalisations about the data without intervention or guidance from the user. The discovery or data mining tools aim to reveal a large number of facts about the data in as short a time as possible.

An example of such a model is a bank database which is mined to discover the many groups of customers to target for a mailing campaign. The data is searched with no hypothesis in mind other than for the system to group the customers according to the common characteristics found.

### 1.4 Data Warehousing

Data mining potential can be enhanced if the appropriate data has been collected and stored in a data warehouse. A data warehouse is a relational database management system (RDMS) designed specifically to meet the needs of transaction processing systems. It can be loosely defined as any centralised data repository which can be queried for business benefit but this will be more clearly defined later. Data warehousing is a new powerful technique making it possible to extract archived operational data and overcome inconsistencies between different legacy data formats. As well as integrating data throughout an enterprise, regardless of location, format, or communication requirements it is possible to incorporate additional or expert information. It is,

the logical link between what the managers see in their decision support EIS applications and the company's operational activities

John McIntyre of SAS Institute Inc

In other words the data warehouse provides data that is already transformed and summarized, therefore making it an appropriate environment for more efficient DSS and EIS applications.

#### 1.4.1 Characteristics of a data warehouse

According to Bill Inmon, author of Building the Data Warehouse and the guru who is widely considered to be the originator of the data warehousing concept, there are generally four characteristics that describe a data warehouse:

* subject-oriented: data are organized according to subject instead of application e.g. an insurance company using a data warehouse would organize their data by customer, premium, and claim, instead of by different products (auto, life, etc.). The data organized by subject contain only the information necessary for decision support processing.
* integrated: When data resides in many separate applications in the operational environment, encoding of data is often inconsistent. For instance, in one application, gender might be coded as "m" and "f" in another by 0 and 1. When data are moved from the operational environment into the data warehouse, they assume a consistent coding convention e.g. gender data is transformed to "m" and "f".
* time-variant: The data warehouse contains a place for storing data that are five to 10 years old, or older, to be used for comparisons, trends, and forecasting. These data are not updated.
* non-volatile: Data are not updated or changed in any way once they enter the data warehouse, but are only loaded and accessed.

#### 1.4.2 Processes in data warehousing

The first phase in data warehousing is to "insulate" your current operational information, ie to preserve the security and integrity of mission-critical OLTP applications, while giving you access to the broadest possible base of data. The resulting database or data warehouse may consume hundreds of gigabytes - or even terabytes - of disk space, what is required then are efficient techniques for storing and retrieving massive amounts of information. Increasingly, large organizations have found that only parallel processing systems offer sufficient bandwidth.

The data warehouse thus retrieves data from a variety of heterogeneous operational databases. The data is then transformed and delivered to the data warehouse/store based on a selected model (or mapping definition). The data transformation and movement processes are executed whenever an update to the warehouse data is required so there should some form of automation to manage and execute these functions. The information that describes the model and definition of the source data elements is called "metadata". The metadata is the means by which the end-user finds and understands the data in the warehouse and is an important part of the warehouse. The metadata should at the very least contain;

* the structure of the data;
* the algorithm used for summarization;
* and the mapping from the operational environment to the data warehouse.

Data cleansing is an important aspect of creating an efficient data warehouse in that it is the removal of certain aspects of operational data, such as low-level transaction information, which slow down the query times. The cleansing stage has to be as dynamic as possible to accommodate all types of queries even those which may require low-level information. Data should be extracted from production sources at regular intervals and pooled centrally but the cleansing process has to remove duplication and reconcile differences between various styles of data collection.

Once the data has been cleaned it is then transferred to the data warehouse which typically is a large database on a high performance box either SMP, Symmetric Multi-Processing or MPP, Massively Parallel Processing. Number-crunching power is another important aspect of data warehousing because of the complexity involved in processing ad hoc queries and because of the vast quantities of data that the organisation want to use in the warehouse. A data warehouse can be used in different ways for example it can be used as a central store against which the queries are run or it can be used to like a data mart. Data marts which are small warehouses can be established to provide subsets of the main store and summarised information depending on the requirements of a specific group/department. The central store approach generally uses very simple data structures with very little assumptions about the relationships between data whereas marts often use multidimensional databases which can speed up query processing as they can have data structures which are reflect the most likely questions.

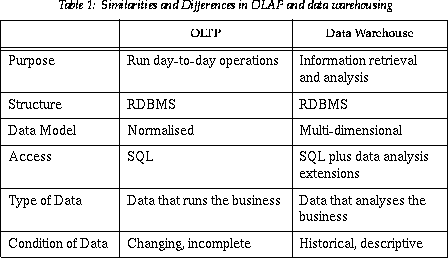
Many vendors have products that provide one or more of the above described data warehouse functions. However, it can take a significant amount of work and specialized programming to provide the interoperability needed between products from multiple vendors to enable them to perform the required data warehouse processes. A typical implementation usually involves a mixture of products from a variety of suppliers.

Another approach to data warehousing is Parsaye's Sandwich Paradigm put forward by Dr. Kamran Parsaye, CEO of Information Discovery, Hermosa Beach, CA. This paradigm or philosophy encourages acceptance of the probability that the first iteration of a data-warehousing effort will require considerable revision. The Sandwich Paradigm advocates the following approach:

* pre-mine the data to determine what formats and data are needed to support a data-mining application;
* build a prototype mini-data warehouse i.e the meat of the sandwich, with most of the features envisaged for the end product;
* revise the strategies as necessary;
* build the final warehouse.

#### 1.4.3 Data warehousing and OLTP systems

A database which is built for on line transaction processing, OLTP, is generally regarded as unsuitable for data warehousing as they have been designed with a different set of needs in mind ie maximising transaction capacity and typically having hundreds of tables in order not to lock out users etc. Data warehouses are interested in query processing as opposed to transaction processing.

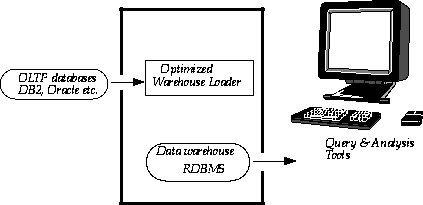
OLTP systems cannot be repositories of facts and historical data for business analysis. They cannot quickly answer ad hoc queries and rapid retrieval is almost impossible. The data is inconsistent and changing, duplicate entries exist, entries can be missing and there is an absence of historical data which is necessary to analyse trends. Basically OLTP offers large amounts of raw data which is not easily understood. The data warehouse offers the potential to retrieve and analyse information quickly and easily. Data warehouses do have similarities with OLTP as shown in the table below.  


The data warehouse serves a different purpose from that of OLTP systems by allowing business analysis queries to be answered as opposed to "simple aggregations" such as `what is the current account balance for this customer?' Typical data warehouse queries include such things as `which product line sells best in middle-America and how does this correlate to demographic data?'

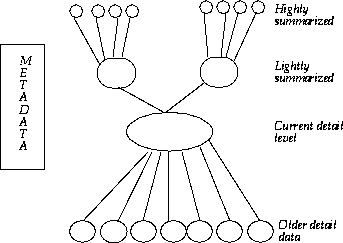
#### 1.4.4 The Data Warehouse model

Data warehousing is the process of extracting and transforming operational data into informational data and loading it into a central data store or warehouse. Once the data is loaded it is accessible via desktop query and analysis tools by the decision makers.

The data warehouse model is illustrated in the following diagram.

Figure 2: A data warehouse model  


The data within the actual warehouse itself has a distinct structure with the emphasis on different levels of summarization as shown in the figure below.

Figure 3: The structure of data inside the data warehouse  


The current detail data is central in importance as it:

* reflects the most recent happenings, which are usually the most interesting;
* it is voluminous as it is stored at the lowest level of granularity;
* it is always (almost) stored on disk storage which is fast to access but expensive and complex to manage.

Older detail data is stored on some form of mass storage, it is infrequently accessed and stored at a level detail consistent with current detailed data.

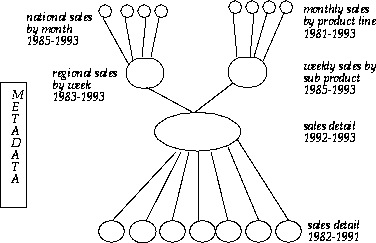
Lightly summarized data is data distilled from the low level of detail found at the current detailed level and generally is stored on disk storage. When building the data warehouse have to consider what unit of time is summarization done over and also the contents or what attributes the summarized data will contain.

Highly summarized data is compact and easily accessible and can even be found outside the warehouse.

Metadata is the final component of the data warehouse and is really of a different dimension in that it is not the same as data drawn from the operational environment but is used as:

* a directory to help the DSS analyst locate the contents of the data warehouse,
* a guide to the mapping of data as the data is transformed from the operational environment to the data warehouse environment,
* a guide to the algorithms used for summarization between the current detailed data and the lightly summarized data and the lightly summarized data and the highly summarized data, etc.

The basic structure has been described but Bill Inmon fills in the details to make the example come alive as shown in the following diagram.

Figure 4: An example of levels of summarization of data inside the data warehouse  


The diagram assumes the year is 1993 hence the current detail data is 1992-93. Generally sales data doesn't reach the current level of detail for 24 hours as it waits until it is no longer available to the operational system i.e. it takes 24 hours for it to get to the data warehouse. Sales details are summarized weekly by subproduct and region to produce the lightly summarized detail. Weekly sales are then summarized again to produce the highly summarized data.

#### 1.4.5 Problems with data warehousing

One of the problems with data mining software has been the rush of companies to jump on the band wagon as

these companies have slapped `data warehouse' labels on traditional transaction-processing products, and co-opted the lexicon of the industry in order to be considered players in this fast-growing category.

Chris Erickson, president and CEO of Red Brick (HPCwire, Oct. 13, 1995)

Red Brick Systems have established a criteria for a relational database management system (RDBMS) suitable for data warehousing, and documented 10 specialized requirements for an RDBMS to qualify as a relational data warehouse server, this criteria is listed in the next section.

According to Red Brick, the requirements for data warehouse RDBMSs begin with the loading and preparation of data for query and analysis. If a product fails to meet the criteria at this stage, the rest of the system will be inaccurate, unreliable and unavailable.

#### 1.4.6 Criteria for a data warehouse

The criteria for data warehouse RDBMSs are as follows:

* Load Performance - Data warehouses require incremental loading of new data on a periodic basis within narrow time windows; performance of the load process should be measured in hundreds of millions of rows and gigabytes per hour and must not artificially constrain the volume of data required by the business.
* Load Processing - Many steps must be taken to load new or updated data into the data warehouse including data conversions, filtering, reformatting, integrity checks, physical storage, indexing, and metadata update. These steps must be executed as a single, seamless unit of work.
* Data Quality Management - The shift to fact-based management demands the highest data quality. The warehouse must ensure local consistency, global consistency, and referential integrity despite "dirty" sources and massive database size. While loading and preparation are necessary steps, they are not sufficient. Query throughput is the measure of success for a data warehouse application. As more questions are answered, analysts are catalysed to ask more creative and insightful questions.
* Query Performance - Fact-based management and ad-hoc analysis must not be slowed or inhibited by the performance of the data warehouse RDBMS; large, complex queries for key business operations must complete in seconds not days.
* Terabyte Scalability - Data warehouse sizes are growing at astonishing rates. Today these range from a few to hundreds of gigabytes, and terabyte-sized data warehouses are a near-term reality. The RDBMS must not have any architectural limitations. It must support modular and parallel management. It must support continued availability in the event of a point failure, and must provide a fundamentally different mechanism for recovery. It must support near-line mass storage devices such as optical disk and Hierarchical Storage Management devices. Lastly, query performance must not be dependent on the size of the database, but rather on the complexity of the query.
* Mass User Scalability - Access to warehouse data must no longer be limited to the elite few. The RDBMS server must support hundreds, even thousands, of concurrent users while maintaining acceptable query performance.
* Networked Data Warehouse - Data warehouses rarely exist in isolation. Multiple data warehouse systems cooperate in a larger network of data warehouses. The server must include tools that coordinate the movement of subsets of data between warehouses. Users must be able to look at and work with multiple warehouses from a single client workstation. Warehouse managers have to manage and administer a network of warehouses from a single physical location.
* Warehouse Administration - The very large scale and time-cyclic nature of the data warehouse demands administrative ease and flexibility. The RDBMS must provide controls for implementing resource limits, chargeback accounting to allocate costs back to users, and query prioritization to address the needs of different user classes and activities. The RDBMS must also provide for workload tracking and tuning so system resources may be optimized for maximum performance and throughput. "The most visible and measurable value of implementing a data warehouse is evidenced in the uninhibited, creative access to data it provides the end user.
* Integrated Dimensional Analysis - The power of multidimensional views is widely accepted, and dimensional support must be inherent in the warehouse RDBMS to provide the highest performance for relational OLAP tools. The RDBMS must support fast, easy creation of precomputed summaries common in large data warehouses. It also should provide the maintenance tools to automate the creation of these precomputed aggregates. Dynamic calculation of aggregates should be consistent with the interactive performance needs.
* Advanced Query Functionality - End users require advanced analytic calculations, sequential and comparative analysis, and consistent access to detailed and summarized data. Using SQL in a client/server point-and-click tool environment may sometimes be impractical or even impossible. The RDBMS must provide a complete set of analytic operations including core sequential and statistical operations.

### 1.5 Data mining problems/issues

Data mining systems rely on databases to supply the raw data for input and this raises problems in that databases tend be dynamic, incomplete, noisy, and large. Other problems arise as a result of the adequacy and relevance of the information stored.

#### 1.5.1 Limited Information

A database is often designed for purposes different from data mining and sometimes the properties or attributes that would simplify the learning task are not present nor can they be requested from the real world. Inconclusive data causes problems because if some attributes essential to knowledge about the application domain are not present in the data it may be impossible to discover significant knowledge about a given domain. For example cannot diagnose malaria from a patient database if that database does not contain the patients red blood cell count.

#### 1.5.2 Noise and missing values

Databases are usually contaminated by errors so it cannot be assumed that the data they contain is entirely correct. Attributes which rely on subjective or measurement judgements can give rise to errors such that some examples may even be mis-classified. Error in either the values of attributes or class information are known as noise. Obviously where possible it is desirable to eliminate noise from the classification information as this affects the overall accuracy of the generated rules.

Missing data can be treated by discovery systems in a number of ways such as;

* simply disregard missing values
* omit the corresponding records
* infer missing values from known values
* treat missing data as a special value to be included additionally in the attribute domain
* or average over the missing values using Bayesian techniques.

Noisy data in the sense of being imprecise is characteristic of all data collection and typically fit a regular statistical distribution such as Gaussian while wrong values are data entry errors. Statistical methods can treat problems of noisy data, and separate different types of noise.

#### 1.5.3 Uncertainty

Uncertainty refers to the severity of the error and the degree of noise in the data. Data precision is an important consideration in a discovery system.

#### 1.5.4 Size, updates, and irrelevant fields

Databases tend to be large and dynamic in that their contents are ever-changing as information is added, modified or removed. The problem with this from the data mining perspective is how to ensure that the rules are up-to-date and consistent with the most current information. Also the learning system has to be time-sensitive as some data values vary over time and the discovery system is affected by the `timeliness' of the data.

Another issue is the relevance or irrelevance of the fields in the database to the current focus of discovery for example post codes are fundamental to any studies trying to establish a geographical connection to an item of interest such as the sales of a product.

### 1.6 Potential Applications

Data mining has many and varied fields of application some of which are listed below.

#### 1.6.1 Retail/Marketing

* Identify buying patterns from customers
* Find associations among customer demographic characteristics
* Predict response to mailing campaigns
* Market basket analysis

#### 1.6.2 Banking

* Detect patterns of fraudulent credit card use
* Identify `loyal' customers
* Predict customers likely to change their credit card affiliation
* Determine credit card spending by customer groups
* Find hidden correlations between different financial indicators
* Identify stock trading rules from historical market data

#### 1.6.3 Insurance and Health Care

* Claims analysis - i.e which medical procedures are claimed together
* Predict which customers will buy new policies
* Identify behaviour patterns of risky customers
* Identify fraudulent behaviour

#### 1.6.4 Transportation

* Determine the distribution schedules among outlets
* Analyse loading patterns

#### 1.6.5 Medicine

* Characterise patient behaviour to predict office visits
* Identify successful medical therapies for different illnesses

## 2 Data Mining Functions

Data mining methods may be classified by the function they perform or according to the class of application they can be used in. Some of the main techniques used in data mining are described in this section.

### 2.1 Classification

Data mine tools have to infer a model from the database, and in the case of supervised learning this requires the user to define one or more classes. The database contains one or more attributes that denote the class of a tuple and these are known as predicted attributes whereas the remaining attributes are called predicting attributes. A combination of values for the predicted attributes defines a class.

When learning classification rules the system has to find the rules that predict the class from the predicting attributes so firstly the user has to define conditions for each class, the data mine system then constructs descriptions for the classes. Basically the system should given a case or tuple with certain known attribute values be able to predict what class this case belongs to.

Once classes are defined the system should infer rules that govern the classification therefore the system should be able to find the description of each class. The descriptions should only refer to the predicting attributes of the training set so that the positive examples should satisfy the description and none of the negative. A rule said to be correct if its description covers all the positive examples and none of the negative examples of a class.

A rule is generally presented as, if the left hand side (LHS) then the right hand side (RHS), so that in all instances where LHS is true then RHS is also true, are very probable. The categories of rules are:

* exact rule - permits no exceptions so each object of LHS must be an element of RHS
* strong rule - allows some exceptions, but the exceptions have a given limit
* probablistic rule - relates the conditional probability P(RHS|LHS) to the probability P(RHS)

Other types of rules are classification rules where LHS is a sufficient condition to classify objects as belonging to the concept referred to in the RHS.

### 2.2 Associations

Given a collection of items and a set of records, each of which contain some number of items from the given collection, an association function is an operation against this set of records which return affinities or patterns that exist among the collection of items. These patterns can be expressed by rules such as "72% of all the records that contain items A, B and C also contain items D and E." The specific percentage of occurrences (in this case 72) is called the confidence factor of the rule. Also, in this rule, A,B and C are said to be on an opposite side of the rule to D and E. Associations can involve any number of items on either side of the rule.

A typical application, identified by IBM, that can be built using an association function is Market Basket Analysis. This is where a retailer run an association operator over the point of sales transaction log, which contains among other information, transaction identifiers and product identifiers. The set of products identifiers listed under the same transaction identifier constitutes a record. The output of the association function is, in this case, a list of product affinities. Thus, by invoking an association function, the market basket analysis application can determine affinities such as "20% of the time that a specific brand toaster is sold, customers also buy a set of kitchen gloves and matching cover sets."

Another example of the use of associations is the analysis of the claim forms submitted by patients to a medical insurance company. Every claim form contains a set of medical procedures that were performed on a given patient during one visit. By defining the set of items to be the collection of all medical procedures that can be performed on a patient and the records to correspond to each claim form, the application can find, using the association function, relationships among medical procedures that are often performed together.

### 2.3 Sequential/Temporal patterns

Sequential/temporal pattern functions analyse a collection of records over a period of time for example to identify trends. Where the identity of a customer who made a purchase is known an analysis can be made of the collection of related records of the same structure (i.e. consisting of a number of items drawn from a given collection of items). The records are related by the identity of the customer who did the repeated purchases. Such a situation is typical of a direct mail application where for example a catalogue merchant has the information, for each customer, of the sets of products that the customer buys in every purchase order. A sequential pattern function will analyse such collections of related records and will detect frequently occurring patterns of products bought over time. A sequential pattern operator could also be used to discover for example the set of purchases that frequently precedes the purchase of a microwave oven.

Sequential pattern mining functions are quite powerful and can be used to detect the set of customers associated with some frequent buying patterns. Use of these functions on for example a set of insurance claims can lead to the identification of frequently occurring sequences of medical procedures applied to patients which can help identify good medical practices as well as to potentially detect some medical insurance fraud.

### 2.4 Clustering/Segmentation

Clustering and segmentation are the processes of creating a partition so that all the members of each set of the partition are similar according to some metric. A cluster is a set of objects grouped together because of their similarity or proximity. Objects are often decomposed into an exhaustive and/or mutually exclusive set of clusters.

Clustering according to similarity is a very powerful technique, the key to it being to translate some intuitive measure of similarity into a quantitative measure. When learning is unsupervised then the system has to discover its own classes i.e. the system clusters the data in the database. The system has to discover subsets of related objects in the training set and then it has to find descriptions that describe each of these subsets.

There are a number of approachs for forming clusters. One approach is to form rules which dictate membership in the same group based on the level of similarity between members. Another approach is to build set functions that measure some property of partitions as functions of some parameter of the partition.

#### 2.4.1 IBM - Market Basket Analysis example

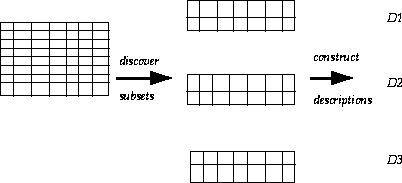
IBM have used segmentation techniques in their Market Basket Analysis on POS transactions where they separate a set of untagged input records into reasonable groups according to product revenue by market basket i.e. the market baskets were segmented based on the number and type of products in the individual baskets.

Each segment reports total revenue and number of baskets and using a neural network 275,000 transaction records were divided into 16 segments. The following types of analysis were also available, revenue by segment, baskets by segment, average revenue by segment etc.

## 3 Data Mining Techniques

### 3.1 Cluster Analysis

In an unsupervised learning environment the system has to discover its own classes and one way in which it does this is to cluster the data in the database as shown in the following diagram. The first step is to discover subsets of related objects and then find descriptions e.eg D1, D2, D3 etc. which describe each of these subsets.

Figure 5: Discovering clusters and descriptions in a database  


Clustering and segmentation basically partition the database so that each partition or group is similar according to some criteria or metric. Clustering according to similarity is a concept which appears in many disciplines. If a measure of similarity is available there are a number of techniques for forming clusters. Membership of groups can be based on the level of similarity between members and from this the rules of membership can be defined. Another approach is to build set functions that measure some property of partitions ie groups or subsets as functions of some parameter of the partition. This latter approach achieves what is known as optimal partitioning.

Many data mining applications make use of clustering according to similarity for example to segment a client/customer base. Clustering according to optimization of set functions is used in data analysis e.g. when setting insurance tariffs the customers can be segmented according to a number of parameters and the optimal tariff segmentation achieved.

Clustering/segmentation in databases are the processes of separating a data set into components that reflect a consistent pattern of behaviour. Once the patterns have been established they can then be used to "deconstruct" data into more understandable subsets and also they provide sub-groups of a population for further analysis or action which is important when dealing with very large databases. For example a database could be used for profile generation for target marketing where previous response to mailing campaigns can be used to generate a profile of people who responded and this can be used to predict response and filter mailing lists to achieve the best response.

### 3.2 Induction

A database is a store of information but more important is the information which can be inferred from it. There are two main inference techniques available ie deduction and induction.

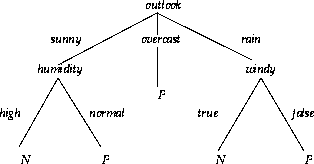
* Deduction is a technique to infer information that is a logical consequence of the information in the database e.g. the join operator applied to two relational tables where the first concerns employees and departments and the second departments and managers infers a relation between employee and managers.
* Induction has been described earlier as the technique to infer information that is generalised from the database as in the example mentioned above to infer that each employee has a manager. This is higher level information or knowledge in that it is a general statement about objects in the database. The database is searched for patterns or regularities.

Induction has been used in the following ways within data mining.

#### 3.2.1 decision trees

Decision trees are simple knowledge representation and they classify examples to a finite number of classes, the nodes are labelled with attribute names, the edges are labelled with possible values for this attribute and the leaves labelled with different classes. Objects are classified by following a path down the tree, by taking the edges, corresponding to the values of the attributes in an object.

The following is an example of objects that describe the weather at a given time. The objects contain information on the outlook, humidity etc. Some objects are positive examples denote by P and others are negative i.e. N. Classification is in this case the construction of a tree structure, illustrated in the following diagram, which can be used to classify all the objects correctly.

Figure 6:   
  
Decision tree structure

#### 3.2.2 rule induction

A data mine system has to infer a model from the database that is it may define classes such that the database contains one or more attributes that denote the class of a tuple ie the predicted attributes while the remaining attributes are the predicting attributes. Class can then be defined by condition on the attributes. When the classes are defined the system should be able to infer the rules that govern classification, in other words the system should find the description of each class.

Production rules have been widely used to represent knowledge in expert systems and they have the advantage of being easily interpreted by human experts because of their modularity i.e. a single rule can be understood in isolation and doesn't need reference to other rules. The propositional like structure of such rules has been described earlier but can summed up as if-then rules.

### 3.3 Neural networks

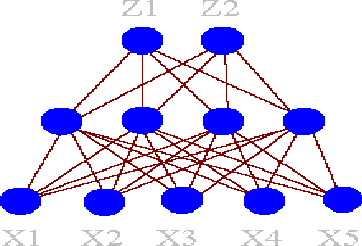
Neural networks are an approach to computing that involves developing mathematical structures with the ability to learn. The methods are the result of academic investigations to model nervous system learning. Neural networks have the remarkable ability to derive meaning from complicated or imprecise data and can be used to extract patterns and detect trends that are too complex to be noticed by either humans or other computer techniques. A trained neural network can be thought of as an "expert" in the category of information it has been given to analyse. This expert can then be used to provide projections given new situations of interest and answer "what if" questions.

Neural networks have broad applicability to real world business problems and have already been successfully applied in many industries. Since neural networks are best at identifying patterns or trends in data, they are well suited for prediction or forecasting needs including:

* sales forecasting
* industrial process control
* customer research
* data validation
* risk management
* target marketing etc.

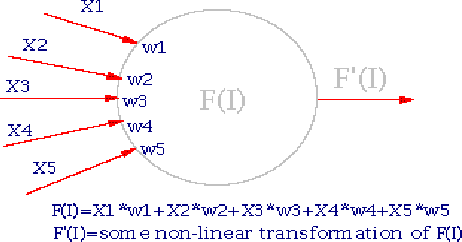
Neural networks use a set of processing elements (or nodes) analogous to neurons in the brain. These processing elements are interconnected in a network that can then identify patterns in data once it is exposed to the data, i.e the network learns from experience just as people do. This distinguishes neural networks from traditional computing programs, that simply follow instructions in a fixed sequential order.

The structure of a neural network looks something like the following:

Figure 7: Structure of a neural network  


The bottom layer represents the input layer, in this case with 5 inputs labels X1 through X5. In the middle is something called the hidden layer, with a variable number of nodes. It is the hidden layer that performs much of the work of the network. The output layer in this case has two nodes, Z1 and Z2 representing output values we are trying to determine from the inputs. For example, predict sales (output) based on past sales, price and season (input).

Each node in the hidden layer is fully connected to the inputs which means that what is learned in a hidden node is based on all the inputs taken together. Statisticians maintain that the network can pick up the interdependencies in the model. The following diagram provides some detail into what goes on inside a hidden node.

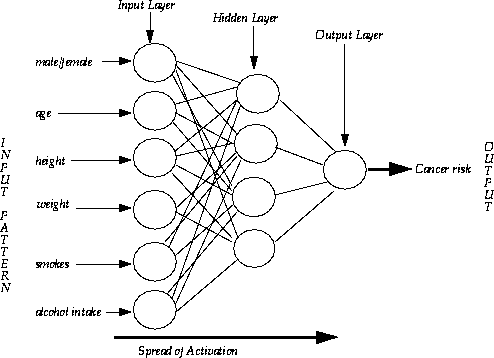
Figure 8: Inside a Node   


Simply speaking a weighted sum is performed: X1 times W1 plus X2 times W2 on through X5 and W5. This weighted sum is performed for each hidden node and each output node and is how interactions are represented in the network.

The issue of where the network get the weights from is important but suffice to say that the network learns to reduce error in it's prediction of events already known (ie, past history).

The problems of using neural networks have been summed by Arun Swami of Silicon Graphics Computer Systems. Neural networks have been used successfully for classification but suffer somewhat in that the resulting network is viewed as a black box and no explanation of the results is given. This lack of explanation inhibits confidence, acceptance and application of results. He also notes as a problem the fact that neural networks suffered from long learning times which become worse as the volume of data grows.

The Clementine User Guide has the following simple diagram to summarise a neural net trained to identify the risk of cancer from a number of factors.

Figure 9:   
  
Example Neural network from Clementine User Guide

### 3.4 On-line Analytical processing

A major issue in information processing is how to process larger and larger databases, containing increasingly complex data, without sacrificing response time. The client/server architecture gives organizations the opportunity to deploy specialized servers which are optimized for handling specific data management problems. Until recently, organizations have tried to target relational database management systems (RDBMSs) for the complete spectrum of database applications. It is however apparent that there are major categories of database applications which are not suitably serviced by relational database systems. Oracle, for example, has built a totally new Media Server for handling multimedia applications. Sybase uses an object-oriented DBMS (OODBMS) in its Gain Momentum product which is designed to handle complex data such as images and audio. Another category of applications is that of on-line analytical processing (OLAP). OLAP was a term coined by E F Codd (1993) and was defined by him as;

the dynamic synthesis, analysis and consolidation of large volumes of multidimensional data

Codd has developed rules or requirements for an OLAP system;

* multidimensional conceptual view
* transparency
* accessibility
* consistent reporting performance
* client/server architecture
* generic dimensionality
* dynamic sparse matrix handling
* multi-user support
* unrestricted cross dimensional operations
* intuitative data manipulation
* flexible reporting
* unlimited dimensions and aggregation levels

An alternative definition of OLAP has been supplied by Nigel Pendse who unlike Codd does not mix technology prescriptions with application requirements. Pendse defines OLAP as, Fast Analysis of Shared Multidimensional Information which means;

Fast in that users should get a response in seconds and so doesn't lose their chain of thought;

Analysis in that the system can provide analysis functions in an intuitative manner and that the functions should supply business logic and statistical analysis relevant to the users application;

Shared from the point of view of supporting multiple users concurrently;

Multidimensional as a main requirement so that the system supplies a multidimensional conceptual view of the data including support for multiple hierarchies;

Information is the data and the derived information required by the user application.

One question is what is multidimensional data and when does it become OLAP? It is essentially a way to build associations between dissimilar pieces of information using predefined business rules about the information you are using. Kirk Cruikshank of Arbor Software has identified three components to OLAP, in an issue of UNIX News on data warehousing;

* A multidimensional database must be able to express complex business calculations very easily. The data must be referenced and mathematics defined. In a relational system there is no relation between line items which makes it very difficult to express business mathematics.
* Intuitative navigation in order to `roam around' data which requires mining hierarchies.
* Instant response i.e. the need to give the user the information as quick as possible.

Dimensional databases are not without problem as they are not suited to storing all types of data such as lists for example customer addresses and purchase orders etc. Relational systems are also superior in security, backup and replication services as these tend not to be available at the same level in dimensional systems. The advantages of a dimensional system are the freedom they offer in that the user is free to explore the data and receive the type of report they want without being restricted to a set format.

#### 3.4.1 OLAP Example

An example OLAP database may be comprised of sales data which has been aggregated by region, product type, and sales channel. A typical OLAP query might access a multi-gigabyte/multi-year sales database in order to find all product sales in each region for each product type. After reviewing the results, an analyst might further refine the query to find sales volume for each sales channel within region/product classifications. As a last step the analyst might want to perform year-to-year or quarter-to-quarter comparisons for each sales channel. This whole process must be carried out on-line with rapid response time so that the analysis process is undisturbed. OLAP queries can be characterized as on-line transactions which:

* Access very large amounts of data, e.g. several years of sales data.
* Analyse the relationships between many types of business elements e.g. sales, products, regions, channels.
* Involve aggregated data e.g. sales volumes, budgeted dollars and dollars spent.
* Compare aggregated data over hierarchical time periods e.g. monthly, quarterly, yearly.
* Present data in different perspectives e.g. sales by region vs. sales by channels by product within each region.
* Involve complex calculations between data elements e.g. expected profit as calculated as a function of sales revenue for each type of sales channel in a particular region.
* Are able to respond quickly to user requests so that users can pursue an analytical thought process without being stymied by the system.

#### 3.4.2 Comparison of OLAP and OLTP

OLAP applications are quite different from On-line Transaction Processing (OLTP) applications which consist of a large number of relatively simple transactions. The transactions usually retrieve and update a small number of records that are contained in several distinct tables. The relationships between the tables are generally simple.

A typical customer order entry OLTP transaction might retrieve all of the data relating to a specific customer and then insert a new order for the customer. Information is selected from the customer, customer order, and detail line tables. Each row in each table contains a customer identification number which is used to relate the rows from the different tables. The relationships between the records are simple and only a few records are actually retrieved or updated by a single transaction.

The difference between OLAP and OLTP has been summarised as, OLTP servers handle mission-critical production data accessed through simple queries; while OLAP servers handle management-critical data accessed through an iterative analytical investigation. Both OLAP and OLTP, have specialized requirements and therefore require special optimized servers for the two types of processing.

OLAP database servers use multidimensional structures to store data and relationships between data. Multidimensional structures can be best visualized as cubes of data, and cubes within cubes of data. Each side of the cube is considered a dimension.

Each dimension represents a different category such as product type, region, sales channel, and time. Each cell within the multidimensional structure contains aggregated data relating elements along each of the dimensions. For example, a single cell may contain the total sales for a given product in a region for a specific sales channel in a single month. Multidimensional databases are a compact and easy to understand vehicle for visualizing and manipulating data elements that have many inter relationships.

OLAP database servers support common analytical operations including: consolidation, drill-down, and "slicing and dicing".

* Consolidation - involves the aggregation of data such as simple roll-ups or complex expressions involving inter-related data. For example, sales offices can be rolled-up to districts and districts rolled-up to regions.
* Drill-Down - OLAP data servers can also go in the reverse direction and automatically display detail data which comprises consolidated data. This is called drill-downs. Consolidation and drill-down are an inherent property of OLAP servers.
* "Slicing and Dicing" - Slicing and dicing refers to the ability to look at the database from different viewpoints. One slice of the sales database might show all sales of product type within regions. Another slice might show all sales by sales channel within each product type. Slicing and dicing is often performed along a time axis in order to analyse trends and find patterns.

OLAP servers have the means for storing multidimensional data in a compressed form. This is accomplished by dynamically selecting physical storage arrangements and compression techniques that maximize space utilization. Dense data (i.e., data exists for a high percentage of dimension cells) are stored separately from sparse data (i.e., a significant percentage of cells are empty). For example, a given sales channel may only sell a few products, so the cells that relate sales channels to products will be mostly empty and therefore sparse. By optimizing space utilization, OLAP servers can minimize physical storage requirements, thus making it possible to analyse exceptionally large amounts of data. It also makes it possible to load more data into computer memory which helps to significantly improve performance by minimizing physical disk I/O.

In conclusion OLAP servers logically organize data in multiple dimensions which allows users to quickly and easily analyse complex data relationships. The database itself is physically organized in such a way that related data can be rapidly retrieved across multiple dimensions. OLAP servers are very efficient when storing and processing multidimensional data. RDBMSs have been developed and optimized to handle OLTP applications. Relational database designs concentrate on reliability and transaction processing speed, instead of decision support need. The different types of server can therefore benefit a broad range of data management applications.

### 3.5 Data Visualisation

Data visualisation makes it possible for the analyst to gain a deeper, more intuitive understanding of the data and as such can work well along side data mining. Data mining allows the analyst to focus on certain patterns and trends and explore in-depth using visualisation. On its own data visualisation can be overwhelmed by the volume of data in a database but in conjunction with data mining can help with exploration.

## 4 Software - past and present developments

This section outlines the historic background or the evolution of database systems in terms of parallel processing and data mining with reference to the part played by some of the main vendors and their successes.

### 4.1 New architectures

The best of the best commercial database packages are now available for massively parallel processors including IBM DB2, INFORMIX-OnLine XPS, ORACLE7 RDBMS and SYBASE System 10. This evolution, however, has not been an easy road for the pioneers.

HPCwire by Michael Erbschloe, contributing editor Oct. 6, 1995

The evolution described by Michael Erbschloe is detailed and expanded on in the following sections.

#### 4.1.1 Obstacles

What were the problems at the start?

* the typical scientific user knew nothing of commercial business applications and gave little attention or credence to the adaptation of high performance computers to business environments.
* the business database programmers, who, although well versed in database management and applications, knew nothing of massively parallel principles.

The solution was for database software producers to create easy-to-use tools and form strategic relationships with hardware manufacturers and consulting firms.

#### 4.1.2 The key

The key is the retooling database software to maspar environments. Parallel processors can easily assign small, independent transactions to different processors. With more processors, more transactions can be executed without reducing throughput. This same concept applies to executing multiple independent SQL statements. A set of SQL statements can be broken up and allocated to different processors to increase speed.

Multiple data streams allow several operations to proceed simultaneously. A customer table, for example, can be spread across multiple disks, and independent threads can search each subset of the customer data. As data is partitioned into multiple subsets performance is increased. I/O subsystems then just feed data from the disks to the appropriate threads or streams.

An essential part of designing a database for parallel processing is the partitioning scheme. Because large databases are indexed, independent indexes must also be partitioned to maximize performance. There are five partitioning methods used to accomplish this:

1. Hashing, where data is assigned to disks based on a hash key

2. Round-robin partitioning, which assigns a row to partitions in sequence.

3. Allocating rows to nodes based on ranges of values.

4. Schema partitioning (Sybase Navigation Server), which lets you tie tables to specific partitions.

5. User-defined roles (Informix).

#### 4.1.3 Oracle was first

Oracle was the first to market parallel database packages with their flagship product, ORACLE7 RDBMS having been installed at over 100 user sites. Oracle began beta support for the IBM SP platform in July 1994.

Ease of use is an important factor in the success of any commercial application and by design the Oracle Parallel Server hides the complexities of data layout from the users. Users who wish to add disks or processor nodes can do so without complex data reorganization and application re-partitioning. In addition, Oracle Parallel Server software uses the same SQL interface as the Oracle7 database. Since no new commands or extensions to existing commands are needed, previously developed tools and applications will run unchanged.

The Oracle Parallel Server technology performs both the parallelization and optimization automatically, eliminating the need to re-educate application developers and end users. It is also easy for user organizations to deploy because it eliminates many traditional implementation burdens.

Reference - http://www.oracle.com.

#### 4.1.4 Red Brick has a strong showing

Red Brick Systems, based in Los Gatos, Calif., specializes in software products used for fast and accurate business decisions where large client/server databases, usually tens to hundreds of gigabytes in size with hundreds of millions of records, are the norm. These applications require historical context, but timely analysis of complex data relationships for both consolidated and detailed business information.

Red Brick Warehouse VPT, (Very large data warehouse support, Parallel query processing, Time based data management), is a DBMS tuned for data warehouse applications. It employs specialized indexing techniques which are designed to facilitate data warehousing. The join accelerator STARjoin uses a special index to multiple tables that participate in a join. With its parallel capability it can run applications that can handle up to 500 GB or more of data It is a parallel database product that significantly improves the organization, availability, administration, and performance of data warehouse applications.

Unlike RDBMS products optimized for on-line transaction processing, Red Brick Warehouse VPT allows business management applications to be developed and deployed quickly;

* to query very large databases of information gathered from disparate sources;
* to provide the best access to both consolidated and detailed business information;
* and to simply run fast.

Red Brick's server-based relational engine is accessible by several popular front-end client application environments which support Microsoft ODBC, Sybase Open Client, and Information Builders, Inc. EDA/SQL interfaces.

Reference - http://www.redbrick.com.

#### 4.1.5 IBM is still the largest

IBM is the world's largest producer of database management software. Eighty percent of the FORTUNE 500, including the top 100 companies, rely on DB2 database solutions to manage data on mainframes, minicomputers, RISC workstations and personal computers. The availability of the new DB2 Parallel Edition, extends the functionality and reliability of the DB2 to IBM's high-performance parallel systems SP2. With DB2 Parallel Edition running on the SP2, users can access very large databases, process huge amount of data, and perform complex queries in minutes.

DB2 Parallel Edition is packaged with the SP2 running AIX and a set of services to help users speed their transactions and quickly and easily derive the benefits of parallel computing. The turnkey solution, called POWERquery, provides a relatively cost-effective, large-scale decision support.

DB2 Parallel Edition is a member of the IBM DB2 family of databases, therefore users do not have to rewrite any applications or retrain their staffs. To a user, the database appears to be a single database server, only faster. It is faster because all functions are performed in parallel, including data and index scans, index creation, backup and restore, joins, inserts, updates and deletes.

Reference - http://www.ibm.com.

#### 4.1.6 INFORMIX is online with 8.0

Informix has been supporting SMP with Informix Parallel Data Query (PDQ) as part of its Dynamic Scalable Architecture (DSA) and through DSA/XMP by extending PDQ functions to work in loosely coupled parallel environments, including clusters. Online 8.0 is the latest high-performance, scalable database server based on Informix's industry-leading DSA. OnLine XPS extends DSA to loosely coupled, shared-nothing computing architectures including clusters of symmetric multiprocessing (SMP) systems and (MPP) systems.

One key to Informix's success on SMP is a joint development agreement with Sequent Computer Systems (Beaverton, Ore. that resulted in a rebuild of the core of Informix OnLine to a multithreaded system with small-grained, lightweight threads. Virtual processors are pooled and the DBMS allocates them dynamically to CPUs, based on processing requirements. OnLine XPS' high availability, systems management based on the Tivoli Management Environment (TME), data partitioning, enhanced parallel SQL operations, and other features are designed to simplify and economize VLDB applications. OnLine XPS also offers a significant improvement in performance for mission-critical, data-intensive tasks associated with data warehousing, decision support, imaging, document management and workflow, and other VLDB operational environments.

Although Informix databases, such as OnLine XPS and INFORMIX-OnLine Dynamic Server, are at the heart of data warehousing solutions, other products and services must integrate with the databases to ensure a successful data warehouse implementation, a critical component of a data warehouse architecture is online analytical processing (OLAP).

Informix delivers relational multidimensional capabilities through strategic partnerships with Information Advantage, MicroStrategy, and Stanford Technology Group. Informix also has proven partnerships with technology providers, such as Business Objects, Coopers & Lybrand, Evolutionary Technologies, KPMG, Price Waterhouse, Prism, and SHL Systemhouse, to provide capabilities such as data modelling, data extraction, data access, multidimensional analysis, and systems integration.

Reference - http://www.informix.com.

#### 4.1.7 Sybase and System 10

Sybase has improved multithreading with System 10 which has been designed to handle interquery and transaction parallelizing on SMP computers with very large, heavyweight threads. Up to 64 processors can be utilized as SQL servers configured into a single system image. This was accomplished in part by the use of the Sybase Navigation server which takes advantage of parallel computers. Parallelism is achieved by an SQL Server on a processor and control servers, which manages parallel operations. Sybase's Navigation Server partitions data by hashing, ranges, or schema partitioning. Reports indicate that the partitioning scheme and keys chosen impact parallel performance.

Sybase IQ was delivered to 24 beta customers in July, providing predictable interactive access to large amounts of data directly in the warehouse. While offering up to 100-fold query performance improvement over standard relational databases, Sybase IQ slashes warehouse query costs by orders of magnitude, requiring up to 80 percent less disk, up to 98 percent less I/O, and utilizing existing hardware, according to Sybase.

An optional extension for the SYBASE SQL Server, SYBASE IQ includes patent-pending Bit-Wise indexing that allows significantly more data to be processed in each instruction, resulting in up to thousands of times faster performance without adding hardware. Beyond simple bit maps, Bit-Wise indexing makes it possible to index every field in the database --including character and numeric fields not supported by other bit-map indexing schemes -- in less than the size of the raw data, substantially reducing disk costs. SYBASE IQ indexes provide a complete map of the data, eliminating table scans and directly accessing just the information required, reducing I/O by up to 98 percent and resulting in fast, predictable answers to any query.

Reference - http://www.sybase.com.

#### 4.1.8 Information Harvester

Information Harvester software on the Convex Exemplar offers market researchers in retail, insurance, financial and telecommunications firms the ability to analyse large data sets in a short time.

The flexibility of the Information Harvesting induction algorithm enables it to adapt to any system. The data can be in the form of numbers, dates, codes, categories, text or any combination thereof. Information Harvester is designed to handle faulty, missing and noisy data. Large variations in the values of an individual field do not hamper the analysis. Information Harvester claims unique abilities to recognize and ignore irrelevant data fields when searching for patterns. In full-scale parallel-processing versions, Information Harvester can handle millions of rows and thousands of variables.

The Exemplar series, based on HP's high performance PA-RISC processors, is the first supercomputer-class family of systems to track the price/performance development cycle of the desktop. They are being used for a range of applications including automotive, tire and aircraft design, petroleum research and exploration, seismic processing, and university, scientific and biomedical research.

Reference - http://www.convex.com

### 4.2 Vendors and Applications

This section examines some of the major vendors of siftware with supporting case studies.

#### 4.2.1 Information Harvesting Inc

The problem of deriving meaningful information from enormous amounts of complex data is being handled by the data mining software produced by Information Harvesting Inc. (IH), founded in 1994 and based in Cambridge, Mass. It makes use of conventional statistical analysis techniques by building upon a proprietary tree-based learning algorithm similar to CART, ID3 and Chaid that generates expert-system-like rules from datasets, initially presented in forms such as numbers, dates, categories, codes, or any combination.

The proprietary Information Harvesting algorithm operates by creating a set of bins for each field in the data, with groups of values within a field ultimately determining the rules. According to the distribution of values the algorithm delineates bin boundaries via fuzzy logic to determine where a given value falls within a bin and thus how the values may be grouped.

A binary tree then generates rules from the data. At the uppermost node the algorithm analyses all data rows, and at each lower level subsets created by the node above are analysed. Each node arrives at a set of rules categorizing the data reviewed at that level. Each rule may include multiple variables (combined with ANDs) or multiple clauses (combined with ORs) and derives from the way variables fall into various bins. A prediction can be based on one or more rules.

Rule quality, the amount of error for each rule, and importance, how often each rule is used for making predictions, are also assessed by the software. This avoids the effect of simply memorizing historical data or misunderstanding the relevance of a given rule. Design rows are used to extract the rules per se, but test rows are utilized to determine the rules level of accuracy.

In addition, the program is set to optimize results by running over the same datasets again and again while adjusting the internal parameters for the best result. Optimization can be achieved with either a rapid hill-climbing algorithm or completely with a modified genetic algorithm.

The data mining modules are written in ANSI C and thus can be ported to a wide range of platforms: on client/server architecture (where the application uses TCP/IP), parallel processing machines, or mainframe supercomputers.

Two examples of companies using the software are:

Healthcare - Michael Reese Medical Associates (MRMA) employed data mining software from Information Harvesting and Vantage Point as a tool for gaining advantage in contract negotiations. The 28-doctor group had to predict trends in type, price, location, and use of service, since they must negotiate with insurance companies to provide certain services at a set monthly fee, doctors must accurately predict their per member/per month cost to break even or make a profit. Normally physicians could only make an intuitive estimate roughly based on after-the-fact evaluations of prior estimates when determining this critical figure whereas data mining offered a new approach.

Finance - The Philadelphia Police and Fire Federal Credit Union (PFFCU) used data mining to maximize their membership base by cultivating multiple relationships (e.g. consumer loans, annuities, credit cards, etc.) with members. Because the membership base is extremely homogeneous (police and fire dept. employees and their families), data had to be deeply drilled to identify segmented groups. Used in conjunction with software such as InterGlobal Financial Systems' Credit Analyzer, Information Harvester identified members most and least profitable to the organization as well as those who would make attractive loan candidates. Data mining often led PFFCU to accurate but counter-intuitive results. For example, members who had filed for bankruptcy were more inclined to clear debts with the Credit Union than outside lenders. Thus, PFFCU identified members with imperfect credit histories but a strong tendency to pay, whereas these individuals would be ignored by large conventional lenders.

#### 4.2.2 Red Brick

Red Brick have a number of cases to present in support of the use of their data mining technology, two of which are H.E.B. of San Antonio, and Hewlett-Packard.

H.E.B.- Category management in retailing

H.E.B. of San Antonio, Texas (sales of approx. $4.5 billion, 225 stores, 50,000 employees) was able to bring a category management application from design to roll out in under nine months because it kept the requirements simple and had database support from Red Brick and server support from Hewlett-Packard Company.

Previously, the marketing information department would take ad hoc requests for information from users, write a program to extract the information, and return the information to the user a week or so later - not timely enough for most business decisions and in some cases not what the user really wanted in the first place.

The organizational change to category management was implemented in 1990. The category manager is characterized as the "CEO" of the category with profit and loss responsibilities, final decision over which products to buy and which to delete, and where the products are to be located on the shelves. The category manager also decides which stores get which products. Although H.E.B. stores are only within the state of Texas, it is a diverse market where some stores near Mexico are 98% Hispanic while suburban Dallas stores may be only 2% Hispanic. The change to category management centralized all merchandising and marketing decisions, removing these decisions from the stores.

As category managers built up their negotiating skills, technical skills, and partnering skills over three years, the need for more timely decision-support information grew. An enterprise-wide survey of users to determine requirements took until September 1993. The company then benchmarked three database management systems - Red Brick, Teradata and Time Machine - and picked Red Brick. The group leased the hardware, a Hewlett-Packard 9000 model T500 (2-processor, with 768M of RAM, and 100GB of disk space--the system now has 200 GB). For a user interface, the company contracted for a custom graphical front-end based on Windows. Also, a COBOL programmer was used to write data extraction programs to take P.O.S. data from the mainframe, format the data properly, and transfer the data to the Red Brick database.

The model was delivered in March 1994 and the application has been up and running without problems since then. The company maintains two years of data by week, by item (257,000 UPCs), by store. This is about 400 million detail records. Summary files are only maintained by time and total company, which can be an advantage.

The goal was to have all queries answered in 4 seconds, but some trends reports with large groups of items over long time periods take 30 - 40 seconds. The users are not always technically oriented, so the design intentionally aimed for simplicity. The system is ad hoc to the extent that the user can specify time, place, and product.

H.E.B. feels that category managers are now making fact-based decisions to determine which products to put in which stores, how much product to send to a store, and the proper product mix. Historically, buyers usually were promoted from the stores and had considerable product knowledge whereas now category managers are coming from other operational areas such as finance and human resources. This is possible because the system give people with limited product knowledge the equivalent of years experience.

Hewlett-Packard: "Discovering" Data To Manage Worldwide Support

Hewlett-Packard, a premier, global provider of hardware systems is known for manufacturing high quality products but to maintain its reputation they depended on delivering service and support through and after product delivery.

The Worldwide Customer Support Organization (WCSO) within Hewlett-Packard is responsible for providing support services to its hardware and software customers. For several years, WCSO has used a data warehouse of financial, account, product, and service contract information to support decision making. WCSO Information Management is responsible for developing and supporting this data warehouse.

Until 1994, WCSO Information Management supported business information queries with a data warehouse architecture based on two HP3000/Allbase systems and an IBM DB2 system. This was a first attempt at collecting, integrating, and storing data related to customer support for decision-making purposes. As they increasingly relied upon the data warehouse, they began to demand better performance, additional data coverage, and more timely data availability.

The warehouse architecture did not keep pace with the increased requirements from WCSO users. Users wanted to get information quickly. Both load and query performance were directly impacted as more data was added. It was to decided to investigate other warehouse alternatives with the aim of finding a new data warehouse that would significantly improve load/query performance, be more cost effective, and support large amounts of data without sacrificing performance. To help select the best combination of hardware and software for the new warehouse, benchmarks were conducted using Red Brick and two other RDBMS products. They did not look at Oracle or Sybase because they were promoting OLTP data functionality and weren't focused upon data warehousing.

Benchmarks included tests simulating some of HP's most demanding user queries, testing the load times for tables in the five to eight million row range. Tests also were conducted to verify that performance did not degrade as data was added into the warehouse. "The Red Brick product performed head and shoulders above the rest," recalls Ryan Uda, Program Manager for WCSO's Information Management Program. Benchmark results showed Red Brick loading data in one hour against ten hours for other systems. Red Brick's query performance was consistently five to ten times faster. Red Brick returned consistently superior performance results even when large amounts of data were added to the warehouse.

HP chose to use Red Brick software on an HP9000 and the project began with the consolidation of the existing three databases into a single data warehouse named "Discovery." This downsizing provided significant cost savings and increased resource efficiencies in managing and supporting the warehouse environment. Today, Discovery supports approximately 250 marketing, finance, and administration users in the Americas, Europe, and Asia-Pacific regions. They pull query results into their desktop report writers, load information into worksheets, or use the data to feed Executive Information Systems. User satisfaction has risen dramatically due to Discovery's vastly improved performance and remodelled business views.

#### 4.2.3 Oracle

For large scale data mining, Oracle on the SP2 offers customers robust functionality and excellent performance. Data spread across multiple SP2 processor nodes is treated as a single image affording exceptionally fast access to very large databases. Oracle Parallel Query allows multiple users to submit complex queries at the same time. Individual complex queries can be broken down and processed across several processing nodes simultaneously. Execution time can be reduced from overnight to hours or minutes, enabling organizations to make better business decisions faster.

Oracle offers products that help customers create, administer and use their data warehouse. Oracle has a large suite of connectivity products that provide transparent access to many popular mainframe databases. Through the use of these products, customers can move data from legacy mainframe applications into the data warehouse on the SP2.

Some of the examples of their technology at work are as follows:

John Alden Insurance based in Miami, Fla., is using Oracle Parallel Query on the SP2 to mine healthcare information and they have seen orders-of-magnitude improvements in response time for typical business queries.

ShopKo Stores, a $2 billion, Wisconsin-based mass merchandise chain which operates 128 stores throughout the Midwest and Northeast, chose the SP2 to meet their current and projected needs for both data mining and mission-critical merchandising applications.

Pacific Bell and U.S. West, both telecommunications providers, have are using the Oracle Warehouse to improve their ability to track customers and identify new service needs. The solutions are based on the Oracle Warehouse, introduced in June, 1995.

* Pacific Bell's data warehouse provides a common set of summarized and compressed information to base decision support systems. The first system is designed to analyse product profitability, and similar decision support systems are in development for marketing, capital investment and procurement, and two additional financial systems.
* U.S. West has implemented a warehousing system to analyse intra-area code calling data from its three operating companies. Running Oracle7 Release 7.2 on a 9-CPU symmetric multiprocessing system from Pyramid, US West's initial centralized architecture supports use by 20 executives and marketing specialists. The next phase will deliver warehouse access to more than 400 service representatives, which will ultimately be expanded up to 4,500 service representatives.

#### 4.2.4 Informix - Data Warehousing

As a major player in the field of data mining Informix have a number of success stories to quote some of which are:

Informix and Associated Grocers (retail example)

Associated Grocers, one of the leading cooperative grocery wholesalers in the northwest United States, with revenues of $1.2 billion, is replacing its traditional mainframe environment with a three-tiered client/server architecture based on Informix database technology. The new system's advanced applications have cut order-fulfilment times in half, reduced inventory carrying costs, and enabled the company to offer its 350 independent grocers greater selection at a lower cost. The details are -

* Hardware: Hewlett-Packard, IBM, AT&T GIS
* Partners: Micro Focus and Lawson Associates
* Applications: Inventory management, post billing, radio frequency, POS scanning, and data warehousing
* Key Informix Products: INFORMIX-OnLine Dynamic Server

In 1991, Associated Grocers embarked on a phased transition from its mainframe-based information system to open systems. The company initially used IBM RS/6000 hardware, and has since included Hewlett-Packard and NCR. In evaluating relational database management systems, Associated Grocers developed a checklist of requirements including education/training, scalability, technical support, solid customer references, and future product direction.

After selecting Informix as its company wide database standard, Associated Grocers then assembled the rest of its system architecture using a three tier model. On tier one, the "client" presentation layer, graphical user interfaces are developed using Microsoft(R) Windows(TM) and Visual Basic(TM). Tier two, based on Hewlett-Packard hardware, runs Micro Focus COBOL applications on top of the OEC Developer Package from Open Environment Corporation. This helps Associated Grocers develop DCE-compliant applications. The third layer, the data layer, is the INFORMIX-OnLine database.

Associated Grocers' pilot Informix-based application provides real-time inventory information for its deli warehouse. In the past, merchandise was received manually, and pertinent product information was later keyed into Associated Grocers' financial system. In contrast, the new system recognizes merchandise right at the receiving dock. Hand-held radio frequency devices allow merchandise to be immediately scanned into the Informix database. Product is assigned to a warehouse location and its expiration date is noted. When orders are filled, products with the earliest expiration dates are shipped first.

An extension to the deli warehouse system is a new post billing system, which is the ability to separate physical and financial inventory. Previously, merchandise could not be released for sale until the financial systems had been updated, which typically occurred over night. The new Informix-based system allows for immediate sale and distribution of recently received merchandise.

A third Informix-based application enables Associated Grocers to economically sell unique items-slow moving merchandise which is ordered monthly versus daily. Rather than incurring the high cost to warehouse these items, Associated Grocers created a direct link to outside speciality warehouses to supply the needed items on demand. Independent stores simply order the merchandise from Associated Grocers. The order goes into Associated Grocers' billing system then gets transmitted to the speciality warehouse, which immediately ships the merchandise to Associated Grocers. The speciality items are loaded onto Associated Grocers' delivery trucks and delivered along with the rest of an independent store's order.

Host Marriott (retail example)

Host Marriott has revenues of $1.1 billion and is a leading provider of food, beverage, and merchandise concession outlets located at airports, travel plazas, and toll roads throughout the United States. The company is streamlining its information systems to develop better cost controls and more effectively manage operations. To accomplish this, Host Marriott selected Informix database technology as its strategic IS foundation, which includes the development of a data warehouse using INFORMIX-OnLine Dynamic Server(TM) and INFORMIX-NewEra(TM). The new system will deliver valuable information throughout the organization, from field operators to corporate analysts. Details of the solution are:

* Hardware: IBM, Hewlett-Packard
* Applications: Sales and marketing, inventory management, labor productivity, and data warehousing
* Informix Products: INFORMIX-OnLine Dynamic Server, INFORMIX-NewEra, INFORMIX-ESQL/C

The company split into two separate companies; Host Marriott and Marriott International, and as the company grew more diverse, so did its computer systems. Unique and more advanced information systems were coupled with inadequate ones. As a result, financial consolidation was primarily done manually, with sales information from each outlet keyed into individual computer systems every night. The information was then sent to Host Marriotts corporate office, where it was posted to the mainframe accounting system, which had no analysis capabilities. Any analysis had to be completed via a second system, proving to be a labor-intensive and slow process.

In an effort to streamline operations and improve system flexibility, Host Marriott is replacing its manually-intensive system with a series of new client/server-based applications using Informix development tools and relational database products running on an IBM RS/6000 and Hewlett-Packard Vectra PCs.

The first of Host Marriotts new Informix-based applications automates its sales and marketing functions. It was developed using INFORMIX-HyperScript(R) Tools--a visual programming environment used to create client/server applications for Windows(TM), UNIX(R), and Macintosh(R) systems, and INFORMIX-ESQL/C--a database application development tool which is used to embed SQL statements directly into C code. Instead of waiting for individual end-of-day reports, the system automatically polls sales data from the point-of-sale terminals at each outlet and consolidates it in the INFORMIX-SE relational database.

This information is used to consolidate and speed up end-of-day reporting, analyse sales, and monitor regulatory compliance. It has reduced a 10 hour process to less than one hour, and enables corporate and concession management to perform the kind of in-depth analysis that allows them to fine tune their product mix, reduce administrative overhead, and ultimately increase profit margins.

Focus is now on a data warehouse to leverage its existing businesses and generate new growth opportunities in the future. The data warehouse is a separate database that Host Marriott is designing explicitly for its data-intensive, decision-support applications. Building a data warehouse will allow them to optimize query times and eliminate impact on the company's production systems. The warehouse is being developed with INFORMIX-NewEra, an open, graphical, object-oriented development environment especially suited for creating enterprise wide client/server database applications.

The foundation of Host Marriotts data warehouse will be INFORMIX-OnLine Dynamic Server, which takes advantage of multiprocessing hardware to perform multiple database functions in parallel. The data warehouse will help the company determine which brands will succeed in which market. It will also help Host Marriott develop more proprietary brands, and deliver better products and services at lower cost.

By pooling sales data, market research, customer satisfaction ratings, etc., Host Marriott will be able to perform detailed analysis in order to eliminate unnecessary costs from operations, and fully leverage new business opportunities. Relying on Informix products and services is enabling Host Marriott to make the important shift from simple data processing to strategic business analysis.

#### 4.2.5 Sybase

There is a lot of interest and activity in data warehousing, recent surveys show that more than 70 percent of Fortune 1000 companies have Data Warehousing projects budgeted or underway at an average cost of $3 million and a typical development time of 6 to 18 months (Meta Group Inc.).

Conventional warehousing applications today extract basic business data from operational systems, edit or transform it in some fashion to ensure its accuracy and clarity, and move it by means of transformation products, custom programming, or "sneaker net" to the newly deployed analytical database system. This extract, edit, load, query, extract, edit, load, query system might be acceptable if business life were very simple and relatively static but that is not the case, new data and data structures are added, changes are made to existing data, and even whole new databases are added.

Sybase Warehouse WORKS

Sybase Warehouse WORKS was designed around four key functions in data warehousing:

* Assembling data from multiple sources
* Transforming data for a consistent and understandable view of the business
* Distributing data to where it is needed by business users
* Providing high-speed access to the data for those business users

The Sybase Warehouse WORKS Alliance Program provides a complete, open, and integrated solution for organizations building and deploying data warehouse solutions. The program addresses the entire range of technology requirements for data warehouse development, including data transformation, data distribution, and interactive data access. The alliance partners have made commitments to adopt the Warehouse WORKS architecture and APIs, as well as to work closely with Sybase in marketing and sales programs.

#### 4.2.6 SG Overview

The advances in data analysis realized through breakthroughs in data warehousing are now being extended by new solutions for data mining. Sophisticated tools for 3D visualization, coupled with data mining software developed by Silicon Graphics, make it possible to bring out patterns and trends in the data that may not have been realized using traditional SQL techniques. These "nuggets" of information can then be brought to the attention of the end user, yielding bottom-line results.

Using fly-through techniques, you can navigate your models on consumer purchasing and channel velocity to follow trends and observe patterns. In response to what you see, you can interact directly with the data, using visual computing to factor critical "what-if" scenarios into your models. By making it possible to go through many such iterations without resorting to over-burdened IS staff for analytical assistance, you can eliminate days - even months - from the review process.

#### 4.2.7 IBM Overview

IBM provides a number of decision support tools to give users a powerful but easy-to-use interface to the data warehouse. IBM Information Warehouse Solutions offer the choice of decision support tools that best meet the needs of the end users in keeping with their commitment to provide open systems implementations.

IBM has announced, a Customer Partnership Program, to work with selected customers to gain experience and validate the applicability of the data mining technology. This offers customers the advantage of IBM's powerful new data mining technology to analyse their data looking for key patterns and associations. Visa and IBM announced an agreement on 30 May 1995 signalling their intention to work together. This will change the way in which Visa and its member banks exchange information worldwide. The proposed structure will facilitate the timely delivery of information and critical decision support tools directly to member financial institutions' desktops worldwide.

IBM Visualizer provides a powerful and comprehensive set of ready to use building blocks and development tools that can support a wide range of end-user requirements for query, report writing, data analysis, chart/graph making, business planning and multimedia database. As a workstation based product, Visualizer is object-oriented and that makes it easy to plug-in additional functions such as those mentioned. And, Visualizer can access databases such as Oracle and Sybase as well as the DB2 family.

There are a number of other decision support products available from IBM based on the platform, operating environment and database with which you need to work. For example, the IBM Application System (AS) provides a client/server architecture and the widest range of decision support functions available for the MVS and VM environments. AS has become the decision support server of choice in these environments because of its capability to access many different data sources. IBM Query Management Facility (QMF) provides query, reporting and graphics functions in the MVS, VM, and CICS environments. The Data Interpretation System (DIS) is an object-oriented set of tools that enable end users to access, analyse and present information with little technical assistance. It is a LAN-based client/server architecture that enables access to IBM and non-IBM relational databases as well as host applications in the MVS and VM environment. These and other products are available from IBM to provide the functions and capabilities needed for a variety of implementation alternatives.

**4. System Analysis**

**4.1 Existing System:**

Almost all the state-of-the-art methods, however, require human experts at certain levels. Most existing methods rely on complex, trained linguistic analyzers (e.g., dependency parsers) to locate phrase mentions, and thus may have unsatisfactory performance on text corpora of new domains and genres without extra but expensive adaption. Our latest domain-independent method SegPhrase outperforms many other approaches but still needs domain experts to first carefully select hundreds of varying quality phrases from millions of candidates, and then annotate them with binary labels. Text indexing algorithms typically filter out stop words and restrict candidate terms to noun phrases. With predefined part-of-speech (POS) rules, one can identify noun phrases as term candidates in POS-tagged documents. Supervised noun phrase chunking techniques, exploit such tagged documents to automatically learn rules for identifying noun phrase boundaries. Other methods may utilize more sophisticated NLP technologies such as dependency parsing to further enhance the precision With candidate terms collected, the next step is to leverage certain statistical measures derived from the corpus to estimate phrase quality. Some methods rely on other reference corpora for the calibration of “termhood”.

**4.1.1 Drawbacks:**

The dependency on these various kinds of linguistic analyzers, domain-dependent language rules, and expensive human labeling, makes it challenging to extend these approaches to emerging, big, and unrestricted corpora, which may include many different domains, topics, and languages.

SegPhrase, additional manual labeling effort needed

**4.2 Problem statement:**

The dependency on these various kinds of linguistic analyzers, domain-dependent language rules, and expensive human labeling, makes it challenging to extend these approaches to emerging, big, and unrestricted corpora, which may include many different domains, topics, and languages.

SegPhrase, additional manual labeling effort needed

**4.3 Proposed System:**

A novel automated phrase mining framework, Auto Phrase has two new techniques.1. Robust Positive-Only Distant Training,2. POS-Guided Phrasal Segmentation. AutoPhrase can support any language as long as a general knowledge base in that language is available, the POS-guided phrasal segmentation can be applied in many scenarios. It is worth mentioning that for domain-specific knowledge bases (e.g., MeSH terms in the biomedical domain) and trained POS taggers, the same paradigm applies. In this study, without loss of generality, we only assume the availability of a general knowledge base together with a pre-trained POS tagger.

**4.3.1 Advantages**:

Our main contributions are highlighted as follows:

1. Automated phrase mining

2. A robust positive-only distant training method for phrase quality estimation to minimize the human effort.

3Develop a novel phrasal segmentation model to leverage POS tags to achieve further improvement, when a POS tagger is available.

**4.4 Modules Description:**

**Robust Positive-Only Distant Training**:

In fact, many high-quality phrases are freely available in general knowledge bases, and they can be easily obtained to a scale that is much larger than that produced by human experts. Domain-specific corpora usually contain some quality phrases also encoded in general knowledge bases, even when there may be no other domain-specific knowledge bases. Therefore, for distant training, we leverage the existing high quality phrases, as available from general knowledge bases, to get rid of additional manual labeling effort. We independently build samples of positive labels from general knowledge bases and negative labels from the given domain corpora, and train a number of base classifiers. We then aggregate the predictions from these classifiers, whose independence helps reduce the noise from negative labels.

**POS-Guided Phrasal Segmentation**:

There is a trade-off between the accuracy and domain-independence when incorporating linguistic processors in the phrase mining method. On the domain independence side, the accuracy might be limited without linguistic knowledge. It is difficult to support multiple languages well, if the method is completely language-blind. On the accuracy side, relying on complex, trained linguistic analyzers may hurt the domain-independence of the phrase mining method.

**Label Pools:**

Public knowledge bases usually encode a considerable number of high-quality phrases in the titles, keywords, and internal links of pages. For example, by analyzing the internal links and synonyms5 in English Wikipedia, more than a hundred thousand high-quality phrases were discovered. As a result, we place these phrases in a **positive pool**. Knowledge bases, however, rarely, if ever, identify phrases that fail to meet our criteria, what we call inferior phrases. An important observation is that the number of phrase candidates, based on n-grams (recall leftmost box ), is huge and the majority of them are actually of inferior quality (e.g., “Francisco opera and”). In practice, based on our experiments, among millions of phrase candidates, usually, only about 10% are in good quality. Therefore, phrase candidates that are derived from the given corpus but that fail to match any high quality phrase derived from the given knowledge base, are used to populate a large but noisy **negative pool**.

**4.5 Feasibility Study**

Preliminary investigation examine project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:

* Technical Feasibility
* Operational Feasibility
* Economical Feasibility

**4.5.1 Economic Feasibility**

A system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economical feasibility, the development cost in creating the system is evaluated against the ultimate benefit derived from the new systems. Financial benefits must equal or exceed the costs.

The system is economically feasible. It does not require any addition hardware or software. Since the interface for this system is developed using the existing resources and technologies available at NIC, There is nominal expenditure and economical feasibility for certain.

**4.5.2 Operational Feasibility**

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization’s operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. Some of the important issues raised are to test the operational feasibility of a project includes the following: -

* Is there sufficient support for the management from the users?
* Will the system be used and work properly if it is being developed and implemented?
* Will there be any resistance from the user that will undermine the possible application benefits?

This system is targeted to be in accordance with the above-mentioned issues. Beforehand, the management issues and user requirements have been taken into consideration. So there is no question of resistance from the users that can undermine the possible application benefits.

The well-planned design would ensure the optimal utilization of the computer resources and would help in the improvement of performance status.

**4.5.3 Technical Feasibility**

The technical issue usually raised during the feasibility stage of the investigation includes the following:

* Does the necessary technology exist to do what is suggested?
* Do the proposed equipments have the technical capacity to hold the data required to use the new system?
* Will the proposed system provide adequate response to inquiries, regardless of the number or location of users?
* Can the system be upgraded if developed?
* Are there technical guarantees of accuracy, reliability, ease of access and data security?

Earlier no system existed to cater to the needs of ‘Secure Infrastructure Implementation System’. The current system developed is technically feasible. It is a web based user interface for audit workflow at NIC-CSD. Thus it provides an easy access to the users. The database’s purpose is to create, establish and maintain a workflow among various entities in order to facilitate all concerned users in their various capacities or roles. Permission to the users would be granted based on the roles specified. Therefore, it provides the technical guarantee of accuracy, reliability and security. The software and hard requirements for the development of this project are not many and are already available in-house at NIC or are available as free as open source.

**5. System Requirements Specification**

**5.1 Introduction**

A **Software Requirements Specification** (**SRS**) – a [requirements specification](http://en.wikipedia.org/wiki/Requirements_specification) for a [software system](http://en.wikipedia.org/wiki/Software_system) – is a complete description of the behavior of a system to be developed. It includes a set of [use cases](http://en.wikipedia.org/wiki/Use_case) that describe all the interactions the users will have with the software. In addition to use cases, the SRS also contains non-functional requirements. [Non-functional requirements](http://en.wikipedia.org/wiki/Non-functional_requirements) are requirements which impose constraints on the design or implementation (such as [performance engineering](http://en.wikipedia.org/wiki/Performance_engineering) requirements, [quality](http://en.wikipedia.org/wiki/Quality_%28business%29) standards, or design constraints).

**System requirements specification:** A structured collection of information that embodies the requirements of a system. A [business analyst](http://en.wikipedia.org/wiki/Business_analyst), sometimes titled [system analyst](http://en.wikipedia.org/wiki/System_analyst), is responsible for analyzing the business needs of their clients and stakeholders to help identify business problems and propose solutions. Within the [systems development life cycle](http://en.wikipedia.org/wiki/Systems_development_life_cycle) domain, typically performs a liaison function between the business side of an enterprise and the information technology department or external service providers. Projects are subject to three sorts of requirements:

* [**Business requirements**](http://en.wikipedia.org/wiki/Business_requirements) describe in business terms what must be delivered or accomplished to provide value.
* **Product requirements** describe properties of a system or product (which could be one of

several ways to accomplish a set of business requirements.)

* **Process requirements** describe activities performed by the developing organization. For instance, process requirements could specify specific methodologies that must be followed, and constraints that the organization must obey.

Product and process requirements are closely linked. Process requirements often specify the activities that will be performed to satisfy a product requirement. For example, a maximum development cost requirement (a process requirement) may be imposed to help achieve a maximum sales price requirement (a product requirement); a requirement that the product be maintainable (a Product requirement) often is addressed by imposing requirements to follow particular development styles

**5.2 Purpose**

An systems engineering, a **requirement** can be a description of what a system must do, referred to as a [Functional Requirement](http://en.wikipedia.org/wiki/Functional_requirements). This type of requirement specifies something that the delivered system must be able to do. Another type of requirement specifies something about the system itself, and how well it performs its functions. Such requirements are often called [Non-functional requirements](http://en.wikipedia.org/wiki/Non-functional_requirements), or 'performance requirements' or 'quality of service requirements.' Examples of such requirements include usability, availability, reliability, supportability, testability and maintainability.

A collection of requirements define the characteristics or features of the desired system. A 'good' list of requirements as far as possible avoids saying how the system should implement the requirements, leaving such decisions to the system designer. Specifying how the system should be implemented is called "implementation bias" or "solution engineering". However, implementation constraints on the solution may validly be expressed by the future owner, for example for required interfaces to external systems; for interoperability with other systems; and for commonality (e.g. of user interfaces) with other owned products.

In software engineering, the same meanings of requirements apply, except that the focus of interest is the software itself.

**5.3 FUNCTIONAL Requirements**

**Admin:**

In this module, server has to login with valid username and password. After login successful he can do some operations such as View Users Request, Add Candidate, View all Candidate Details, View candidate Access Points, View search Transactions, View Auto Phrase Mining, Positive Pool search, Negative Pool search, View Profile, Security.

**User:**

In this module, Users registers before doing some operation. After registration successful he can login by using valid username and password. After login successful he can do some operations such as Search for Candidate, View MySearch History, View MyPhrase Mining, View Profile, and Security.

**5.4 NON Functional Requirements**

The major non-functional Requirements of the system are as follows

**Usability**

The system is designed with completely automated process hence there is no or less user intervention.

**Reliability**

The system is more reliable because of the qualities that are inherited from the chosen platform java. The code built by using java is more reliable.

**Performance**

This system is developing in the high level languages and using the advanced front-end and back-end technologies it will give response to the end user on client system with in very less time.

**Supportability**

The system is designed to be the cross platform supportable. The system is supported on a wide range of hardware and any software platform, which is having JVM, built into the system.

**Implementation**

The system is implemented in web environment using struts framework. The apache tomcat is used as the web server and windows xp professional is used as the platform.

Interface the user interface is based on Struts provides HTML Tag.

**5.5 Input & Output Design**

**Input Design:**

Inaccurate input data are the most common causes of errors in data processing. Errors entered by data entry operators can be controlled by the Input design. "Input design is the process of converting user originated inputs to computer based formats". It consists of developing specification and procedure for data preparation.

**Objectives of input design:**

The main objectives of input design are:

1. Controlling amount of input: Due to so many reasons, design should control the quantity of data    for input. Reducing the data requirement can lower cost by reducing labour expenses. By reducing input requirement, the analyst can speed the entire process from data capture to providing results to the users.

2. Avoiding delay: A processing delay resulting from data preparation or data entry operator is called bottleneck. Avoiding bottleneck should always be one objective of the analyst while designing output.

3. Avoiding errors in data: The rate at which errors occurs depends on the quantity of data, i.e. smaller the amount of data to input the fewer the opportunities for errors.

4. Keeping the process simple: Simplicity works and is accepted by the users. Complexity should be avoided when there are simple alternatives.

**Output Design:**

The term output necessarily implies to information on printed or displayed by an information system. Following are the activities that are carried out in output design stage.

Ø  Identification of specific output required to meet the information requirements.

Ø  Selecting of methods for processing outputs.

Ø  Designing of reports, formats or other documents that acts as a carrier of information.

**Output Design Activities**

The output design of an information system must meet the following objectives:

1. The output design should provide information about the past, present or future events. The operational control level outputs provide operations of the past and present events. On the other hand, strategic planning level provides information of the future events.

2. The output design should indicate the important events, opportunities and problems.

3. The output design should be designed keeping in mind that an action must be triggered in response to some event. A set of rule is pre- designed for such trigger.

4. The output design should produce some action to the transaction. For e.g. when the telephone bill is generated, a receipt is printed.

**5.6 Hardware Requirements:**

Processor : Pentium IV

Hard Disk : 500GB

RAM : 2GB or more

**5.6 Software Requirements:**

Operating System : Windows XP/2003 or Linux

User Interface : HTML, CSS

Client-side Scripting : JavaScript

Programming Language : Java

Web Applications : JDBC, Servlet, JSP

IDE/Workbench : My Eclipse

Database : Oracle 10g

Server Deployment : Tomcat 7.X

**6. System Design**

**6.1 Introduction**

The purpose of the design phase is to plan a solution of the problem specified by the requirement document. This phase is the first step in moving from the problem domain to the solution domain. In other words, starting with what is needed, design takes us toward how to satisfy the needs. The design of a system is perhaps the most critical factor affection the quality of the software; it has a major impact on the later phase, particularly testing, maintenance. The output of this phase is the design document. This document is similar to a blueprint for the solution and is used later during implementation, testing and maintenance. The design activity is often divided into two separate phases System Design and Detailed Design.

System Design also called top-level design aims to identify the modules that should be in the system, the specifications of these modules, and how they interact with each other to produce the desired results. At the end of the system design all the major data structures, file formats, output formats, and the major modules in the system and their specifications are decided.

During, Detailed Design, the internal logic of each of the modules specified in system design is decided. During this phase, the details of the data of a module is usually specified in a high-level design description language, which is independent of the target language in which the software will eventually be implemented.

In system design the focus is on identifying the modules, where as during detailed design the focus is on designing the logic for each of the modules. In other works, in system design the attention is on what components are needed, while in detailed design how the components can be implemented in software is the issue.

Design is concerned with identifying software components specifying relationships among components. Specifying software structure and providing blue print for the document phase. Modularity is one of the desirable properties of large systems. It implies that the system is divided into several parts. In such a manner , the interaction between parts is minimal clearly specified.

During the system design activities , Developers bridge the gap between the requirements specification , produced during requirements elicitation and analysis , and the system that is delivered to the user.

Design is the place where the quality is fostered in development . Software design is a process through which requirements are translated into a representation of software.

**6.2 System Model**

**Introduction to UML**

The unified Modeling Language (UML) is a standard language for writing software blueprints. The UML may be used to visualize, specify , construct and document the artifacts of software-intensive system.

The goal of UML is to provide a standard notation that can be used by all object - oriented methods and to select and integrate the best elements .UML is itself does not prescribe or advice on how to use that notation in a software development process or as part of an object - design methodology. The UML is more than just bunch of graphical symbols. Rather , behind each symbol in the UML notation is well-defined semantics.

The system development focuses on three different models of the system.

* Functional model
* Object model
* Dynamic model

**Functional model** in UML is represented with use case diagrams , describing the functionality of the system from user point of view.

**Object model** in UML is represented with class diagrams , describing the structure of the system in terms of objects , attributes , associations and operations.

**Dynamic model**  in UML is represented with sequence diagrams , start chart diagrams and activity diagrams describing the internal behaviour of the system.

**Scenarios**

A Use Case is an abstraction that all describes all possible scenarios involving the described functionality . A scenario is an instance of a use case describing a concrete set of actions.

* The **name** of the scenario enables us to refer it ambiguously. The name of scenario is underlined to indicate it is an instance.
* The **Participating actor instance** field indicates which actor instance are involved in this scenario. Actor instance also have underlined names.

The **Flow of Events** of scenario describe the sequence of events step by step.

**6.3 Data Flow Diagram:**

A graphical tool used to describe and analyze the moment of data through a system manual or automated including the process, stores of data, and delays in the system. Data Flow Diagrams are the central tool and the basis from which other components are developed. The transformation of data from input to output, through processes, may be described logically and independently of the physical components associated with the system. The DFD is also know as a data flow graph or a bubble chart.

DFDs are the model of the proposed system. They clearly should show the requirements on which the new system should be built. Later during design activity this is taken as the basis for drawing the system’s structure charts. The Basic Notation used to create a DFD’s are as follows:

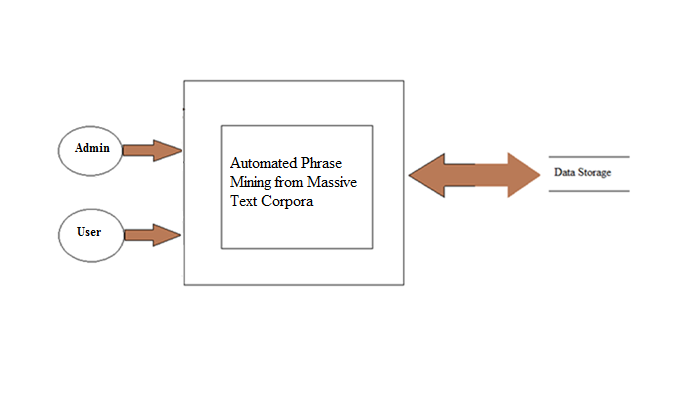
**1. Dataflow:** Data move in a specific direction from an origin to a destination.

**2. Process:** People, procedures, or devices that use or produce (Transform) Data. The physical component is not identified.

**3. Source:** External sources or destination of data, which may be People, programs, organizations or other entities.

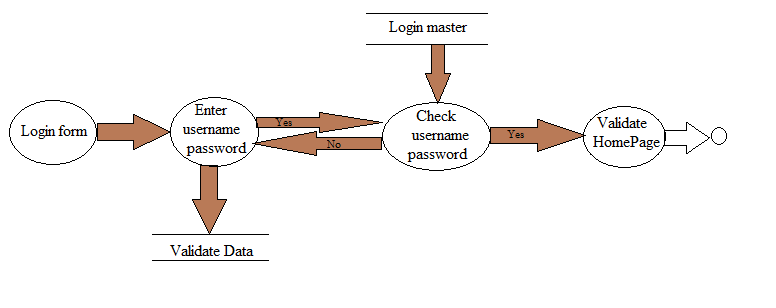
**4. Data Store:** Here data are stored or referenced by a process in the System.

**CONTEXT LEVEL 0 DIAGRAMS:**

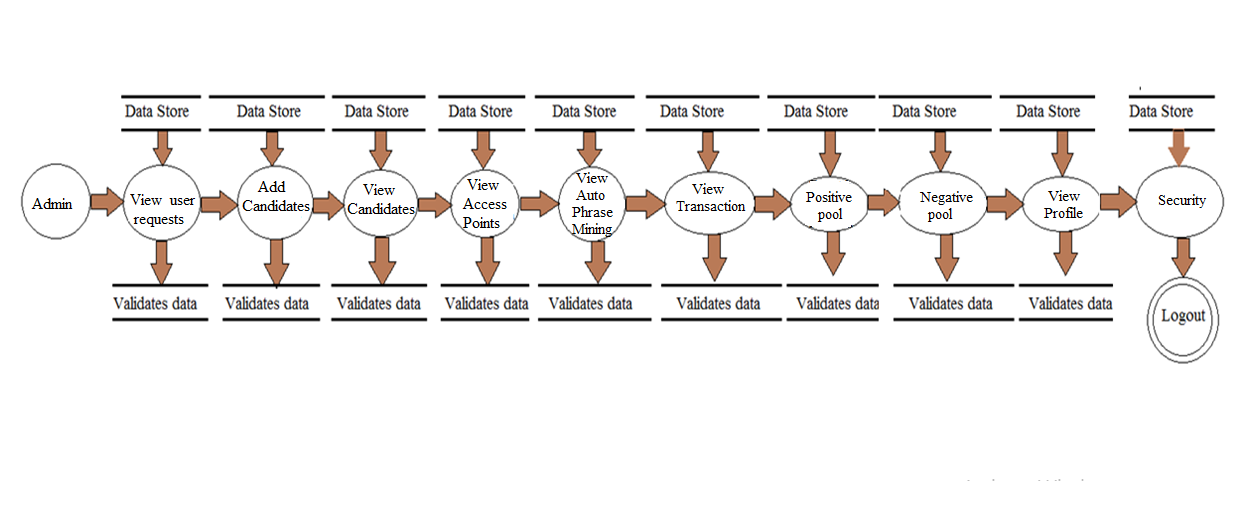
****

**Context level1 Diagram:**

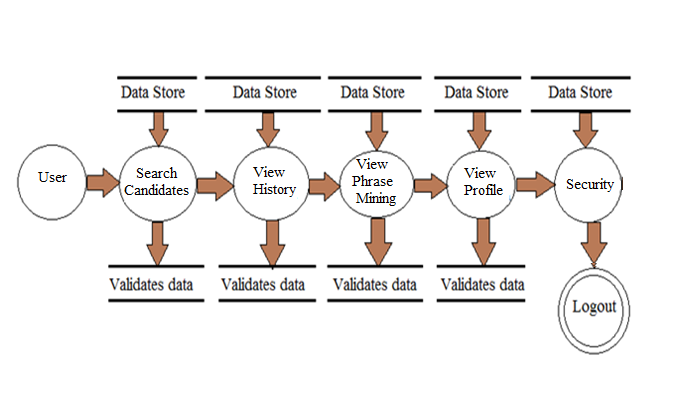
**Login DFD**



**Context level 2:**

****

**Context level 3:**

****

**6.4 UML Diagrams:**

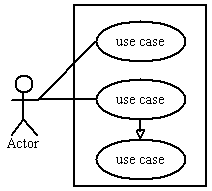
**What is a UML Use Case Diagram?**

Use case diagrams model the functionality of a system using actors and use cases. Use cases are services or functions provided by the system to its users.

**Basic Use Case Diagram Symbols and Notations**

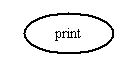
**System**

Draw your system's boundaries using a rectangle that contains use cases. Place actors outside the system's boundaries.



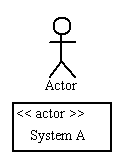
**Use Case**

Draw use cases using ovals. Label with ovals with verbs that represent the system's functions.



**Actors**

Actors are the users of a system. When one system is the actor of another system, label the actor system with the actor stereotype.



**Relationships**

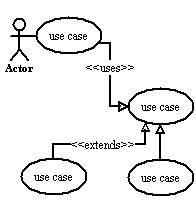
Illustrate relationships between an actor and a use case with a simple line. For relationships among use cases, use arrows labeled either "uses" or "extends." A "uses" relationship indicates that one use case is needed by another in order to perform a task. An "extends" relationship indicates alternative options under a certain use case.

Fig: Admin Use Case diagram



Fig: User Use Case diagram

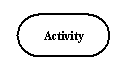
**Activity Diagram**

An activity diagram illustrates the dynamic nature of a system by modeling the flow of control from activity to activity. An activity represents an operation on some class in the system that results in a change in the state of the system. Typically, activity diagrams are used to model workflow or business processes and internal operation. Because an activity diagram is a special kind of state chart diagram, it uses some of the same modeling conventions.

**Basic Activity Diagram Symbols and Notations**

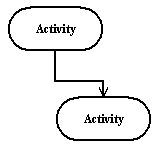
**Action states**

Action states represent the non interruptible actions of objects. You can draw an action state in Smart Draw using a rectangle with rounded corners.



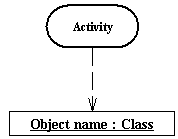
**Action Flow**

Action flow arrows illustrate the relationships among action states.



**Object Flow**

Object flow refers to the creation and modification of objects by activities. An object flow arrow from an action to an object means that the action creates or influences the object. An object flow arrow from an object to an action indicates that the action state uses the object.



**Initial State**

A filled circle followed by an arrow represents the initial action state.

Initial State

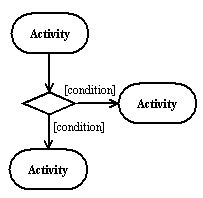
**Final State**

An arrow pointing to a filled circle nested inside another circle represents the final action state.

Final State

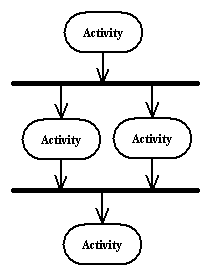
**Branching**

A diamond represents a decision with alternate paths. The outgoing alternates should be labeled with a condition or guard expression. You can also label one of the paths "else."



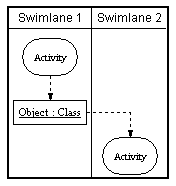
**Synchronization**

A synchronization bar helps illustrate parallel transitions. Synchronization is also called forking and joining.



**Swim lanes**

Swim lanes group related activities into one column.



Activity Diagram

Fig: Admin Activity Diagram



Fig: User Activity diagram

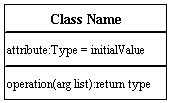
**Class Diagram**

Class diagrams are the backbone of almost every object-oriented method including UML. They describe the static structure of a system.

**Basic Class Diagram Symbols and Notations**

Classes represent an abstraction of entities with common characteristics. Associations represent the relationships between classes.

Illustrate classes with rectangles divided into compartments. Place the name of the class in the first partition (centered, bolded, and capitalized), list the attributes in the second partition, and  
Write operations into the third.



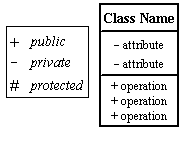
**Active Class**

Active classes initiate and control the flow of activity, while passive classes store data and serve other classes. Illustrate active classes with a thicker border.



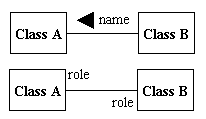
**Visibility**

Use visibility markers to signify who can access the information contained within a class. Private visibility hides information from anything outside the class partition. Public visibility allows all other classes to view the marked information. Protected visibility allows child classes to access information they inherited from a parent class. [.](http://www.smartdraw.com/resources/tutorials/Text-and-Tables)



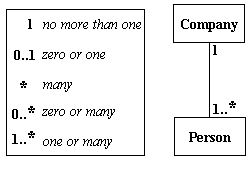
**Associations**

Associations represent static relationships between classes. Place association names above, on, or below the association line. Use a filled arrow to indicate the direction of the relationship. Place roles near the end of an association. Roles represent the way the two classes see each other.  
***Note:*** It's uncommon to name both the association and the class roles.



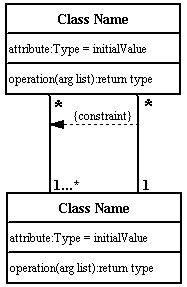
**Multiplicity (Cardinality)**

Place multiplicity notations near the ends of an association. These symbols indicate the number of instances of one class linked to one instance of the other class. For example, one company will have one or more employees, but each employee works for one company only.



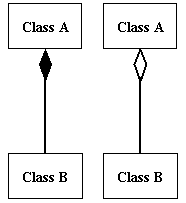
**Constraint**

Place constraints inside curly braces {}.

http://wc1.smartdraw.com/resources/tutorials/images/uml_constraint.gif*Simple Constraint* 

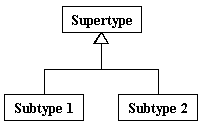
**Composition and Aggregation**

Composition is a special type of aggregation that denotes a strong ownership between Class A, the whole, and Class B, its part. Illustrate **composition** with a filled diamond. Use a hollow diamond to represent a simple **aggregation** relationship, in which the "whole" class plays a more important role than the "part" class, but the two classes are not dependent on each other. The diamond end in both a composition and aggregation relationship points toward the "whole" class or the aggregate



**Generalization**

Generalization is another name for inheritance or an "is a" relationship. It refers to a relationship between two classes where one class is a specialized version of another. For example, Honda is a type of car. So the class Honda would have a generalization relationship with the class car.



In real life coding examples, the difference between inheritance and aggregation can be confusing. If you have an aggregation relationship, the aggregate (the whole) can access only the PUBLIC functions of the part class. On the other hand, inheritance allows the inheriting class to access both the PUBLIC and PROTECTED functions of the super class.



Fig: Server and user class Diagram.

**Sequence Diagram**

Sequence diagrams describe interactions among classes in terms of an exchange of messages over time.

**Basic Sequence Diagram Symbols and Notations**

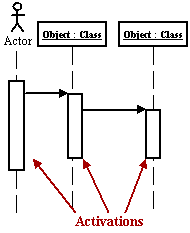
**Class roles**

Class roles describe the way an object will behave in context. Use the UML object symbol to illustrate class roles, but don't list object attributes.

Class roles

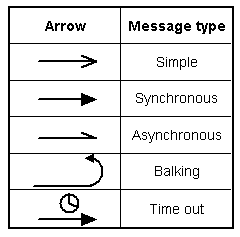
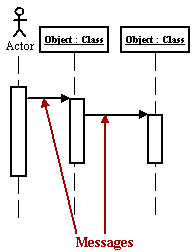
**Activation**

Activation boxes represent the time an object needs to complete a task.



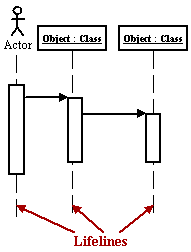
**Messages**

Messages are arrows that represent communication between objects. Use half-arrowed lines to represent asynchronous messages. Asynchronous messages are sent from an object that will not wait for a response from the receiver before continuing its tasks.

  
*Various message types for Sequence and Collaboration diagrams*

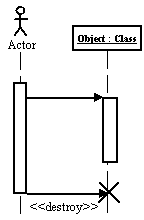
**Lifelines**

Lifelines are vertical dashed lines that indicate the object's presence over time.



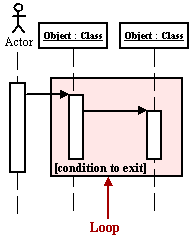
**Destroying Objects**

Objects can be terminated early using an arrow labeled "<< destroy >>" that points to an X.



**Loops**

A repetition or loop within a sequence diagram is depicted as a rectangle. Place the condition for exiting the loop at the bottom left corner in square brackets.



Admin Sequence diagram



Fig: Admin Sequence diagram

User Sequence Diagram

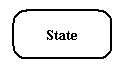
**State chart Diagram**

A state chart diagram shows the behavior of classes in response to external stimuli. This diagram models the dynamic flow of control from state to state within a system.

**Basic State chart Diagram Symbols and Notations**

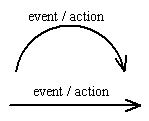
**States**

States represent situations during the life of an object. You can easily illustrate a state in Smart Draw by using a rectangle with rounded corners.



**Transition**

A solid arrow represents the path between different states of an object. Label the transition with the event that triggered it and the action that results from it.



**Initial State**

A filled circle followed by an arrow represents the object's initial state.

Initial State

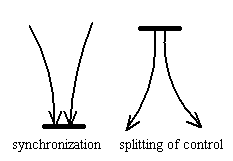
**Final State**

An arrow pointing to a filled circle nested inside another circle represents the object's final state.

Final State

**Synchronization and Splitting of Control**

A short heavy bar with two transitions entering it represents a synchronization of control. A short heavy bar with two transitions leaving it represents a splitting of control that creates multiple states.



Server State Chart Diagram





User State Chart diagram

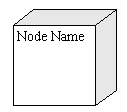
**Deployment Diagram**

Deployment diagrams depict the physical resources in a system including nodes, components, and connections.

**Basic Deployment Diagram Symbols and Notations**

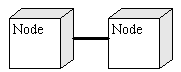
**Component**

A node is a physical resource that executes code components.  
[Learn how to resize grouped objects like nodes.](http://www.smartdraw.com/resources/tutorials/Objects)



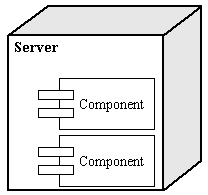
**Association**

Association refers to a physical connection between nodes, such as Ethernet.  
[Learn how to connect two nodes.](http://www.smartdraw.com/resources/tutorials/Lines)



**Components and Nodes**

Place components inside the node that deploys them.



**DEPLOYMENT DIAGRAM:**

****

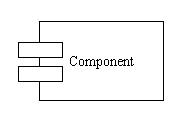
**Component Diagram**

A component diagram describes the organization of the physical components in a system.

**Basic Component Diagram Symbols and Notations**

**Component**

A component is a physical building block of the system. It is represented as a rectangle with tabs.  
[Learn how to resize grouped objects like components.](http://www.smartdraw.com/resources/tutorials/Objects)



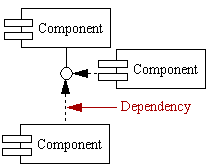
**Interface**

An interface describes a group of operations used or created by components.

Interface

**Dependencies**

Draw dependencies among components using dashed arrows.  
[Learn about line styles in SmartDraw.](http://www.smartdraw.com/resources/tutorials/Lines)

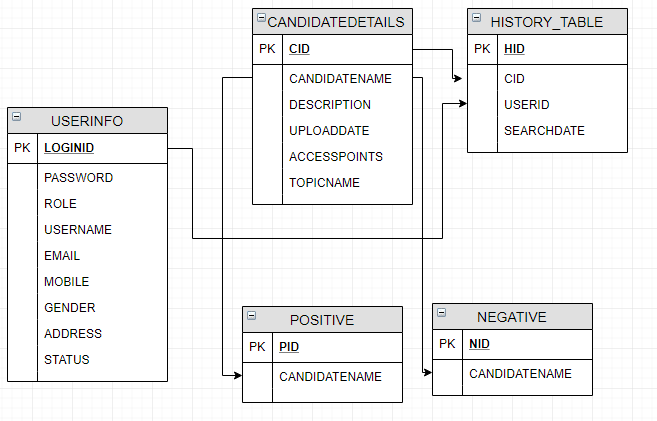


**COMPONENT DIAGRAM:**

****

**6.5 Data Dictionaries and ER Diagram**

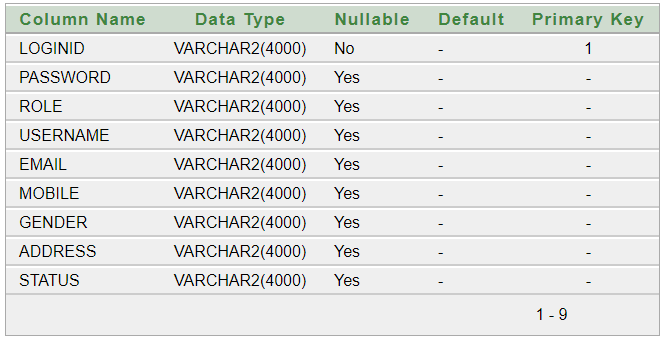
E-R DIAGRAM:



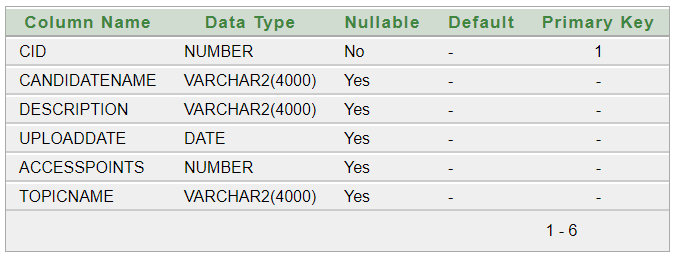
ER diagrams are related to data structure diagrams (DSDs), which focus on the relationships of elements within entities instead of relationships between entities themselves. ER diagrams also are often used in conjunction with data flow diagrams (DFDs), which map out the flow of information for processes or systems.

**Data dictionary**

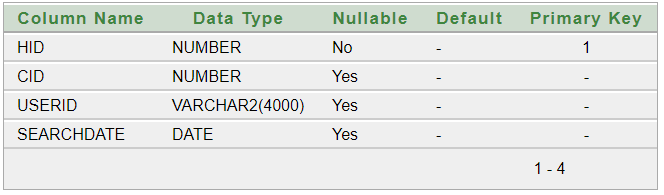
**USERINFO:**

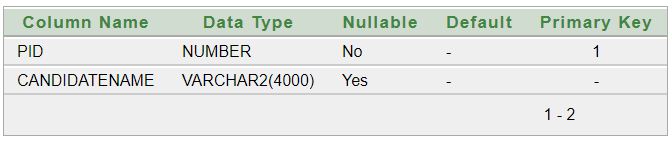
****

**CANDIDATEDETAILS:**

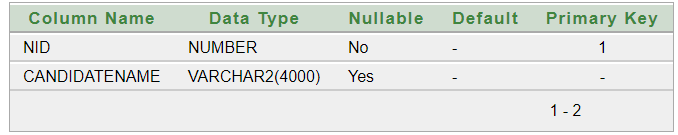
****

**HISTORY\_TABLE:**

****

**POSITIVE: **

**NEGATIVE:**

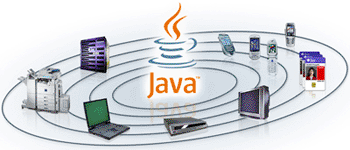
****

**7. Implementation**

**7.1 Technology Description:**

**About the Java Technology**

The Java platform consists of the Java application programming interfaces (APIs) and the Java virtual machine (JVM).



The following Java technology lets developers, designers, and business partners develop and deliver a consistent user experience, with one environment for applications on mobile and embedded devices. Java meshes the power of a rich stack with the ability to deliver customized experiences across such devices.

Java APIs are libraries of compiled code that you can use in your programs. They let you add ready-made and customizable functionality to save you programming time.  
Java programs are run (or interpreted) by another program called the Java Virtual Machine. Rather than running directly on the native operating system, the program is interpreted by the Java VM for the native operating system. This means that any computer system with the Java VM installed can run Java programs regardless of the computer system on which the applications were originally developed.

In the Java programming language, all source code is first written in plain text files ending with the .java extension. Those source files are then compiled into .class files by the javac compiler. A .class file does not contain code that is native to your processor; it instead contains bytecodes — the machine language of the Java Virtual Machine (Java VM). The java launcher tool then runs your application with an instance of the Java Virtual Machine.

Because the Java VM is available on many different operating systems, the same .class files are capable of running on Microsoft Windows, the Solaris TM Operating System (Solaris OS), Linux, or Mac OS.

Java technology is both a programming language and a platform.

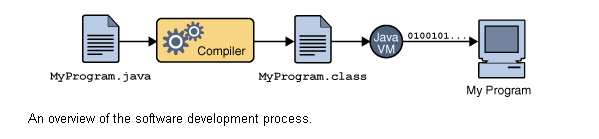
**The Java Programming Language**

The Java programming language is a high-level language that can be characterized by all of the following buzzwords:

|  |  |
| --- | --- |
| * Simple * Object oriented * Distributed * Multithreaded * Dynamic | * Architecture neutral * Portable * High performance * Robust * Secure |

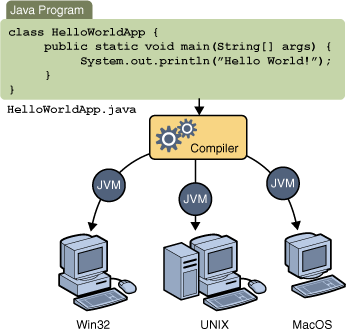
Each of the preceding buzzwords is explained in [The Java Language Environment](http://java.sun.com/docs/white/langenv/) , a white paper written by James Gosling and Henry McGilton.

In the Java programming language, all source code is first written in plain text files ending with the .java extension. Those source files are then compiled into .class files by the javac compiler. A .class file does not contain code that is native to your processor; it instead contains bytecodes — the machine language of the Java Virtual Machine[1](http://download.oracle.com/javase/tutorial/getStarted/intro/definition.html#FOOT) (Java VM). The java launcher tool then runs your application with an instance of the Java Virtual Machine.



An overview of the software development process.

Because the Java VM is available on many different operating systems, the same .class files are capable of running on Microsoft Windows, the Solaris™ Operating System (Solaris OS), Linux, or Mac OS. Some virtual machines, such as the [Java HotSpot virtual machine](http://java.sun.com/products/hotspot/), perform additional steps at runtime to give your application a performance boost. This include various tasks such as finding performance bottlenecks and recompiling (to native code) frequently used sections of code



Through the Java VM, the same application is capable of running on multiple platforms.

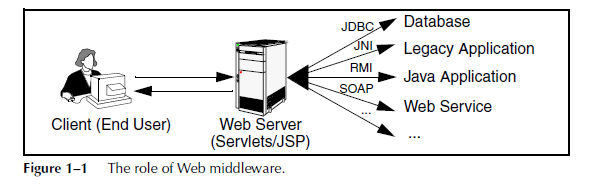
**Servlet and JSP technology**

Servlet and JSP technology has become the technology of choice for developing online stores, interactive

**A Servlet’s Job**

Servlets are Java programs that run on Web or application servers, acting as a middle layer between requests coming from Web browsers or other HTTP clients and databases or applications on the HTTP server. Their job is to perform the following tasks,

as illustrated in Figure 1–1.



1. **Read the explicit data sent by the client.**

The end user normally enters this data in an HTML form on a Web page. However, the data could also come from an applet or a custom HTTP client program. Chapter 4 discusses how servlets read this data.

2. **Read the implicit HTTP request data sent by the browser.**

Figure 1–1 shows a single arrow going from the client to the Web server (the layer where servlets and JSP execute), but there are really two varieties of data: the explicit data that the end user enters in a form and the behind-the-scenes HTTP information. Both varieties are critical. The HTTP information includes cookies, information about media types and compression schemes the browser understands,

3. **Generate the results.**

This process may require talking to a database, executing an RMI or EJB call, invoking a Web service, or computing the response directly. Your real data may be in a relational database. Fine. But your database probably doesn’t speak HTTP or return results in HTML, so the Web browser can’t talk directly to the database. Even if it could, for security reasons, you probably would not want it to. The same argument applies to most other applications. You need the Web middle layer to extract the incoming data from the HTTP stream, talk to the application, and embed the results inside a document.

4. **Send the explicit data (i.e., the document) to the client.**

This document can be sent in a variety of formats, including text (HTML or XML), binary (GIF images), or even a compressed format like gzip that is layered on top of some other underlying format. But, HTML is by far the most common format, so an important servlet/JSP task is to wrap the results inside of HTML.

5. **Send the implicit HTTP response data.**

Figure 1–1 shows a single arrow going from the Web middle layer (the servlet or JSP page) to the client. But, there are really two varieties of data sent: the document itself and the behind-the-scenes HTTP information. Again, both varieties are critical to effective development. Sending HTTP response data involves telling the browser or other client what type of document is being returned (e.g., HTML), setting cookies and caching parameters

**The Advantages of Servlets Over “Traditional” CGI**

Java servlets are more efficient, easier to use, more powerful, more portable, safer, and cheaper than traditional CGI and many alternative CGI-like technologies. With traditional CGI, a new process is started for each HTTP request. If the CGI program itself is relatively short, the overhead of starting the process can dominate the execution time. With servlets, the Java virtual machine stays running and handles each request with a lightweight Java thread, not a heavyweight operating system process. Similarly, in traditional CGI, if there are N requests to the same CGI program, the code for the CGI program is loaded into memory N times. With servlets, however, there would be N threads, but only a single copy of the servlet class would be

loaded. This approach reduces server memory requirements and saves time by instantiating fewer objects. Finally, when a CGI program finishes handling a request, the program terminates. This approach makes it difficult to cache computations, keep database connections open, and perform other optimizations that rely on persistent data. Servlets, however, remain in memory even after they complete a response, so it is straightforward to store arbitrarily complex data between client requests.

**Convenient**

Servlets have an extensive infrastructure for automatically parsing and decoding HTML form data, reading and setting HTTP headers, handling cookies, tracking sessions, and many other such high-level utilities. In CGI, you have to do much of this yourself. Besides, if you already know the Java programming language, why learn Perl too? You’re already convinced that Java technology makes for more reliable and reusable code than does Visual Basic, VBScript, or C++. Why go back to those languages for server-side programming?

**Powerful**

Servlets support several capabilities that are difficult or impossible to accomplish with regular CGI. Servlets can talk directly to the Web server, whereas regular CGI programs cannot, at least not without using a server-specific API. Communicating with the Web server makes it easier to translate relative URLs into concrete path names, for instance. Multiple servlets can also share data, making it easy to implement database connection pooling and similar resource-sharing optimizations. Servlets can also maintain information from request to request, simplifying techniques like session tracking and caching of previous computations.

**Portable**

Servlets are written in the Java programming language and follow a standard API. Servlets are supported directly or by a plug-in on virtually every major Web server. Consequently, servlets written for, say, Macromedia Run can run virtually unchanged on Apache Tomcat, Microsoft Internet Information Server (with a separate plug-in), IBM Web Sphere, planet Enterprise Server, Oracle9i AS, or Star Nine Webster. They are part of the Java 2 Platform, Enterprise Edition (J2EE; see <http://java.sun.com/j2ee/>), so industry support for servlets is becoming even more pervasive.

**Inexpensive**

A number of free or very inexpensive Web servers are good for development use or deployment of low- or medium-volume Web sites. Thus, with servlets and JSP you can start with a free or inexpensive server and migrate to more expensive servers with high-performance capabilities or advanced administration utilities only after your project meets initial success. This is in contrast to many of the other CGI alternatives, which require a significant initial investment for the purchase of a proprietary package. Price and portability are somewhat connected. For example, Marty tries to keep track of the countries of readers that send him questions by email. India was near the top of the list, probably #2 behind the U.S. Marty also taught one of his JSP and servlet

training courses (see http://courses.coreservlets.com/) in Manila, and there was great interest in servlet and JSP technology there. Now, why are India and the Philippines both so interested? We surmise that the answer is twofold. First, both countries have large pools of well-educated software developers.

Second both countries have (or had, at that time) highly unfavorable currency exchange rates against the U.S. dollar. So, buying a special-purpose Web server from a U.S. company consumed a large part of early project funds. But, with servlets and JSP, they could start with a free server: Apache Tomcat (either standalone, embedded in the regular Apache Web server, or embedded in Microsoft IIS). Once the project starts to become successful, they could move to a

server like Caucho Resin that had higher performance and easier administration but that is not free. But none of their servlets or JSP pages have to be rewritten. If their project becomes even larger, they might want to move to a distributed (clustered) environment. No problem: they could move to Macromedia Run Professional, which supports distributed applications (Web farms). Again, none of their servlets or JSP pages have to be rewritten. If the project becomes quite large and complex, they might want to use Enterprise JavaBeans (EJB) to encapsulate their business logic. So, they might switch to BEA Web Logic or Oracle9i AS. Again, none of their servlets

or JSP pages have to be rewritten. Finally, if their project becomes even bigger, they might move it off of their Linux box and onto an IBM mainframe running IBM Web- Sphere. But once again, none of their servlets or JSP pages have to be rewritten

**Secure**

One of the main sources of vulnerabilities in traditional CGI stems from the fact that the programs are often executed by general-purpose operating system shells. So, the CGI programmer must be careful to filter out characters such as backquotes and semicolons that are treated specially by the shell. Implementing this precaution is harder than one might think, and weaknesses stemming from this problem are constantly being uncovered in widely used CGI libraries. A second source of problems is the fact that some CGI programs are processed by languages that do not automatically check array or string bounds. For example, in C and C++ it is perfectly legal to allocate a 100-element array and then write into the 999th “element,” which is really some random part of program memory. So, programmers who forget to perform this check open up their system to deliberate or accidental buffer overflow attacks. Servlets suffer from neither of these problems. Even if a servlet executes a system call (e.g., with Runtime. Exec or JNI) to invoke a program on the local operating system, it does not use a shell to do so. And, of course, array bounds checking and other memory protection features are a central part of the Java programming language.

**Mainstream**

There are a lot of good technologies out there. But if vendors don’t support them and developers don’t know how to use them, what good are they? Servlet and JSP technology is supported by servers from Apache, Oracle, IBM, Sybase, BEA, Macromedia, Caucho, Sun/planet, New Atlanta, ATG, Fujitsu, Ultras, Silver stream, the World Wide Web Consortium (W3C), and many others. Several low-cost plugins add support to Microsoft IIS and Zeus as well. They run on Windows, Unix/Linux, Maces, VMS, and IBM mainframe operating systems. They are the single most popular application of the Java programming language. They are arguably the most popular choice for developing medium to large Web applications. They are used by the airline

industry (most United Airlines and Delta Airlines Web sites), e-commerce (ofoto.com), online banking (First USA Bank, Blanco Popular de Puerto Rico), Web search engines/portals (excite.com), large financial sites (American Century Investments), and hundreds of other sites that you visit every day. Of course, popularity alone is no proof of good technology. Numerous

counter-examples abound. But our point is that you are not experimenting with a

new and unproven technology when you work with server-side Java.

**The Role of JSP**

A somewhat oversimplified view of servlets is that they are Java programs with HTML embedded inside of them. A somewhat oversimplified view of JSP documents is that they are HTML pages with Java code embedded inside of them. For example, compare the sample servlet shown earlier (Listing 1.1) with the JSP page shown below (Listing 1.2). They look totally different; the first looks mostly like a regular Java class, whereas the second looks mostly like a normal HTML page. The interesting thing is that, despite the huge apparent difference, behind the scenes they are the same. In fact, a JSP document is just another way of writing a servlet. JSP pages get translated into servlets, the servlets get compiled, and it is the servlets that run at request time. So, the question is, If JSP technology and servlet technology are essentially equivalent in power, does it matter which you use? The answer is, Yes, yes, yes! The issue is not power, but convenience, ease of use, and maintainability. For example, anything you can do in the Java programming language you could do in assembly language. Does this mean that it does not matter which you use? Hardly. JSP is discussed in great detail starting in Chapter 10. But, it is worthwhile mentioning now how servlets and JSP fit together. JSP is focused on simplifying the creation and maintenance of the HTML. Servlets are best at invoking the business logic and performing complicated operations. A quick rule of thumb is that servlets are best for tasks oriented toward processing, whereas JSP is best for tasks oriented toward presentation. For some requests, servlets are the right choice. For other requests, JSP is a better option. For still others, neither servlets alone nor JSP alone is best, and a combination of the two (see Chapter 15, “Integrating Servlets and JSP: The Model View Controller (MVC) Architecture”) is best. But the point is that you need both servlets and JSP in your overall project: almost no project will consist

entirely of servlets or entirely of JSP. You want both.

**8. Coding**

Registration.jsp:

<!DOCTYPE html>

<html>

<head>

<meta name=*"viewport"* content=*"width=device-width, initial-scale=1"*>

<style>

body {

font-family: *Arial, Helvetica, sans-serif*;

background-color: *white*;

}

\* {

box-sizing: *border-box*;

}

/\* Add padding to containers \*/

.container {

padding: *16px*;

background-color: *white*;

}

/\* Full-width input fields \*/

input[type=text], input[type=password] {

width: *100%*;

padding: *15px*;

margin: *5px 0 22px 0*;

display: *inline-block*;

border: *none*;

background: *#f1f1f1*;

}

select{

width: *100%*;

padding: *12px*;

font-size:*25px*;

border: *1px solid #ccc*;

border-radius: *4px*;

box-sizing: *border-box*;

margin-top: *6px*;

margin-bottom: *16px*;

resize: *vertical*;

}

input[type=text]:focus, input[type=password]:focus {

background-color: *#ddd*;

outline: *none*;

}

input[type=radio]{

height:*30px*;

width:*44px*;

}

/\* Overwrite default styles of hr \*/

hr {

border: *1px solid #f1f1f1*;

margin-bottom: *25px*;

}

/\* Set a style for the submit button \*/

.registerbtn {

background-color: *#4CAF50*;

color: *white*;

padding: *16px 20px*;

margin: *8px 0*;

border: *none*;

cursor: *pointer*;

width: *100%*;

opacity: *0.9*;

}

.registerbtn:hover {

opacity: *1*;

}

/\* Add a blue text color to links \*/

a {

color: *dodgerblue*;

}

/\* Set a grey background color and center the text of the "sign in" section \*/

.signin {

background-color: *#f1f1f1*;

text-align: *center*;

}

</style>

</head>

<body>

<jsp:include page=*"Header.jsp"*></jsp:include>

<jsp:include page=*"HomeMenu.jsp"*></jsp:include>

<center>

<form action=*"RegistrationServlet"*style=*"width:900px;"*>

<center><h2><font face=*"Yu Mincho Demibold"* style=*"color: #00BFFF;"*><font size=*"8"* >Register Here</font></font></h2></center>

<div class=*"container"* style=*"text-align: left"*>

<center><label><font color=*"red"* size=*"4"*><% String status=request.getParameter("status"); if(status!=null){out.println(status);} %></font></label> </center>

<hr>

<label for=*"email"* style=*"color:green"*><b>LOGINID</b></label>

<input type=*"text"* placeholder=*"Create LoginId"* name=*"uid"* required>

<label for=*"psw"* style=*"color:green"*><b>PASSWORD</b></label>

<input type=*"password"* placeholder=*"Create Password"* name=*"pass"* required><!--

<label for="psw-repeat" style="color:blue"><b>ROLE</b></label>

<select name="role" required>

<option value="User">User</option>

<option value="Admin">Admin</option>

</select>

-->

<label for=*"psw"* style=*"color:green"*><b>ROLE</b></label>

<input type=*"text"* placeholder=*"Create Password"* name=*"role"* value=*"User"* required>

<label for=*"email"* style=*"color:green"*><b>USERNAME</b></label>

<input type=*"text"* placeholder=*"Enter User Name"* name=*"uname"* required>

<label for=*"psw"* style=*"color:green"*><b>GENDER</b></label>

<br><br>&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;Male

<input type=*"radio"* size=*"54"* name=*"g"* id=*"g"* value=*"M"* checked=*"checked"* />

&nbsp; FeMale

<input type=*"radio"* name=*"g"* id=*"g"* value=*"F"* /><br><br>&nbsp;

<label for=*"psw"* style=*"color:green"*><b>MAIL</b></label>

<input type=*"text"* placeholder=*"Enter MailId"* name=*"email"* pattern=*"[a-z0-9.\_%+-]+@[a-z0-9.-]+\.[a-z]{2,3}$"* required>

<label for=*"psw-repeat"* style=*"color:green"*><b>MOBILE</b></label>

<input type=*"text"* placeholder=*"Enter Valid Mobile Number"* name=*"mob"* pattern=*"[7896][0-9]{9}"* required>

<label for=*"psw-repeat"* style=*"color:green"*><b>ADDRESS</b></label>

<input type=*"text"* placeholder=*"Enter Address"* name=*"uaddress"* required>

<hr>

<button type=*"submit"* class=*"registerbtn"*>Register</button>

</div>

<div class=*"container signin"*>

<p>Already have an account? <a href=*"Login.jsp"*>Sign in</a>.</p>

</div>

</form>

</center>

</body>

</html>

RegistrationServlet.java:

package com.automated.action;

import java.io.IOException;

import java.io.PrintWriter;

import java.util.Random;

import javax.servlet.RequestDispatcher;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import com.automated.dao.RegistrationDao;

import com.automated.dto.Profilebean;

public class RegistrationServlet extends HttpServlet {

/\*\*

\* The doGet method of the servlet. <br>

\*

\* This method is called when a form has its tag value method equals to get.

\* \* @param request the request send by the client to the server

\* @param response the response send by the server to the client

\* @throws ServletException if an error occurred

\* @throws IOException if an error occurred

\*/

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

response.setContentType("text/html");

PrintWriter out = response.getWriter();

doPost(request, response);

}

/\*\*

\* The doPost method of the servlet. <br>

\* \* This method is called when a form has its tag value method equals to post.

\* \* @param request the request send by the client to the server

\* @param response the response send by the server to the client

\* @throws ServletException if an error occurred

\* @throws IOException if an error occurred

\*/

public void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException { response.setContentType("text/html");

PrintWriter out = response.getWriter();

String loginid=request.getParameter("uid");

String password=request.getParameter("pass");

String username=request.getParameter("uname");

String email=request.getParameter("email");

String mobile=request.getParameter("mob");

String gender=request.getParameter("g");

String address=request.getParameter("uaddress");

String role=request.getParameter("role");

/\* String role="Admin";\*/

Profilebean pb=new Profilebean();

pb.setLoginid(loginid);

pb.setPassword(password);

pb.setUsername(username);

pb.setEmail(email);

pb.setMobileno(mobile);

pb.setGender(gender);

pb.setAddress(address);

pb.setRole(role);

RegistrationDao rdao=new RegistrationDao();

try{

int i=rdao.register(pb);

if(i!=0){

RequestDispatcher rd=request.getRequestDispatcher("Login.jsp?status=Registration Successfully Completed");

rd.include(request, response);

}

else {

RequestDispatcher rd=request.getRequestDispatcher("Registration.jsp?status=Registration Faild Plz Try Again");

rd.include(request, response);

}

}

catch (Exception e) {

RequestDispatcher rd=request.getRequestDispatcher("Registration.jsp?status=Registration Faild Plz Try Again");

rd.include(request, response);

}

}

}

RegistrationDao.java:

package com.automated.dao;

import java.sql.Connection;

import java.sql.PreparedStatement;

import com.automated.db.DbCon;

import com.automated.dto.Profilebean;

public class RegistrationDao extends DbCon{

public int register(Profilebean pb) {

int i=0;

Connection con=null;

String loginid=pb.getLoginid();

String password=pb.getPassword();

String username=pb.getUsername();

String email=pb.getEmail();

String mobile=pb.getMobileno();

String gender=pb.getGender();

String address=pb.getAddress();

String role=pb.getRole();

con=getConnection();

System.out.println("connection post\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"+con);

try {

PreparedStatement pstmt=con.prepareStatement("insert into USERINFO(LOGINID,PASSWORD,ROLE,USERNAME,EMAIL,MOBILE,GENDER,ADDRESS,STATUS) values(?,?,?,?,?,?,?,?,?)");

pstmt.setString(1, loginid);

pstmt.setString(2, password);

pstmt.setString(3, role);

pstmt.setString(4, username);

pstmt.setString(5, email);

pstmt.setString(6, mobile);

pstmt.setString(7, gender);

pstmt.setString(8, address);

pstmt.setString(9, "Pending");

i=pstmt.executeUpdate();

System.out.println(i+"Record is Inserted successfully");

con.close();

} catch (Exception e) {

e.printStackTrace();

}

return i;

}

}

ProfileBean.java

package com.automated.dto;

import java.io.Serializable;

import java.sql.Date;

public class Profilebean implements Serializable {

private String loginid;

private String password;

private String newpassword;

private String username;

private String email;

private String mobileno;

private String gender;

private String address;

private String role;

private String status;

private String candidatename;

private String desc;

private int cid;

private String date;

private String searchdate;

private String topicname;

private int accesspoint;

private int count;

private int size;

private int negative;

private int ppoolcount;

private int npoolcount;

public String getLoginid() {

return loginid;

}

public void setLoginid(String loginid) {

this.loginid = loginid;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public String getNewpassword() {

return newpassword;

}

public void setNewpassword(String newpassword) {

this.newpassword = newpassword;

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

public String getMobileno() {

return mobileno;

}

public void setMobileno(String mobileno) {

this.mobileno = mobileno;

}

public String getGender() {

return gender;

}

public void setGender(String gender) {

this.gender = gender;

}

public String getAddress() {

return address;

}

public void setAddress(String address) {

this.address = address;

}

public String getRole() {

return role;

}

public void setRole(String role) {

this.role = role;

}

public String getStatus() {

return status;

}

public void setStatus(String status) {

this.status = status;

}

public String getCandidatename() {

return candidatename;

}

public void setCandidatename(String candidatename) {

this.candidatename = candidatename;

}

public String getDesc() {

return desc;

}

public void setDesc(String desc) {

this.desc = desc;

}

public int getCid() {

return cid;

}

public void setCid(int cid) {

this.cid = cid;

}

public String getDate() {

return date;

}

public void setDate(String date) {

this.date = date;

}

public String getSearchdate() {

return searchdate;

}

public void setSearchdate(String searchdate) {

this.searchdate = searchdate;

}

public String getTopicname() {

return topicname;

}

public void setTopicname(String topicname) {

this.topicname = topicname;

}

public int getAccesspoint() {

return accesspoint;

}

public void setAccesspoint(int accesspoint) {

this.accesspoint = accesspoint;

}

public int getCount() {

return count;

}

public void setCount(int count) {

this.count = count;

}

public int getSize() {

return size;

}

public void setSize(int size) {

this.size = size;

}

public int getNegative() {

return negative;

}

public void setNegative(int negative) {

this.negative = negative;

}

public int getPpoolcount() {

return ppoolcount;

}

public void setPpoolcount(int ppoolcount) {

this.ppoolcount = ppoolcount;

}

public int getNpoolcount() {

return npoolcount;

}

public void setNpoolcount(int npoolcount) {

this.npoolcount = npoolcount;

}

}

**9. System Testing**

**9.1 Testing Methodologies**

Testing is the process of finding differences between the expected behavior specified by system models and the observed behavior implemented system. From modeling point of view , testing is the attempt of falsification of the system with respect to the system models. The goal of testing is to design tests that exercise defects in the system and to reveal problems.

The process of executing a program with intent of finding errors is called testing. During testing , the program to be tested is executed with a set of test cases , and the output of the program for the test cases is evaluated to determine if the program is performing as expected . Testing forms the first step in determining the errors in the program. The success of testing in revealing errors in program depends critically on test cases.

**Strategic Approach to Software Testing:**

The software engineering process can be viewed as a spiral. Initially system engineering defines the role of software and leads to software requirements analysis where the information domain , functions , behavior , performance , constraints and validation criteria for software are established. moving inward along the spiral , we come to design and finally to coding . To develop computer software we spiral in along streamlines that decreases the level of abstraction on each item.

A Strategy for software testing may also be viewed in the context of the spiral. Unit testing begins at the vertex of the spiral and concentrates on each unit of the software as implemented in source code. Testing will progress by moving outward along the spiral to integration testing , where the focus on the design and the concentration of the software architecture. Talking another turn on outward on the spiral we encounter validation testing where requirements established as part of software requirements analysis are validated against the software that has been constructed . Finally we arrive at system testing , where the software and other system elements are tested as a whole .

UNUNI

UNIT TESTING

MODULE

SUB-SYSTEM

**Component**

SYSTEM TESTING

**Integration Testing**

ACCEPTANCE

**User Testing**

**Different Levels of Testing**

Client Needs Acceptance Testing

Requirements System Testing

Design Integration Testing

Code Unit Testing

Testing is the process of finding difference between the expected behavior specified by system models and the observed behavior of the implemented system.

**Testing Activities**

Different levels of testing are used in the testing process , each level of testing aims to test different aspects of the system. the basic levels are:

Unit testing

Integration testing

System testing

Acceptance testing

**Unit Testing**

Unit testing focuses on the building blocks of the software system, that is, objects and sub system. There are three motivations behind focusing on components. First, unit testing reduces the complexity of the overall tests activities, allowing us to focus on smaller units of the system. Second, unit testing makes it easier to pinpoint and correct faults given that few components are involved in this test. Third, Unit testing allows parallelism in the testing activities, that is each component can be tested independently of one another. Hence the goal is to test the internal logic of the module.

**Integration Testing**

In the integration testing, many test modules are combined into sub systems , which are then tested . The goal here is to see if the modules can be integrated properly, the emphasis being on testing module interaction.

After structural testing and functional testing we get error free modules. These modules are to be integrated to get the required results of the system. After checking a module, another module is tested and is integrated with the previous module. After the integration, the test cases are generated and the results are tested.

**System Testing**

In system testing the entire software is tested . The reference document for this process is the requirement document and the goal is to see whether the software meets its requirements. The system was tested for various test cases with various inputs.

**Acceptance Testing**

Acceptance testing is sometimes performed with realistic data of the client to demonstrate that the software is working satisfactory. Testing here focus on the external behavior of the system , the internal logic of the program is not emphasized . In acceptance testing the system is tested for various inputs.

**Types of Testing**

1. Black box or functional testing
2. White box testing or structural testing

**Black box testing**

This method is used when knowledge of the specified function that a product has been designed to perform is known . The concept of black box is used to represent a system whose inside workings are not available to inspection . In a black box the test item is a "Black" , since its logic is unknown , all that is known is what goes in and what comes out , or the input and output.

Black box testing attempts to find errors in the following categories:

Incorrect or missing functions

Interface errors

Errors in data structure

Performance errors

Initialization and termination errors

As shown in the following figure of Black box testing , we are not thinking of the internal workings , just we think about

What is the output to our system?

What is the output for given input to our system?

**?**

Input Output

The Black box is an imaginary box that hides its internal workings

**White box testing**

White box testing is concerned with testing the implementation of the program. the intent of structural is not to exercise all the inputs or outputs but to exercise the different programming and data structure used in the program. Thus structural testing aims to achieve test cases that will force the desire coverage of different structures . Two types of path testing are statement testing coverage and branch testing coverage.

**INTERNAL WORKING**

Input Output

The White Box testing strategy, the internal workings

**Test Plan**

Testing process starts with a test plan. This plan identifies all the testing related activities that must be performed and specifies the schedules, allocates the resources, and specified guidelines for testing . During the testing of the unit the specified test cases are executed and the actual result compared with expected output. The final output of the testing phase is the test report and the error report.

**Test Data:**

Here all test cases that are used for the system testing are specified. The goal is to test the different functional requirements specified in Software Requirements Specifications (SRS) document.

**Unit Testing:**

Each individual module has been tested against the requirement with some test data.

**Test Report:**

The module is working properly provided the user has to enter information. All data entry forms have tested with specified test cases and all data entry forms are working properly.

**Error Report:**

If the user does not enter data in specified order then the user will be prompted with error messages. Error handling was done to handle the expected and unexpected errors.

**TEST CASES:**

Test cases can be divided in to two types. First one is Positive test cases and second one is negative test cases. In positive test cases are conducted by the developer intention is to get the output. In negative test cases are conducted by the developer intention is to don’t get the output.

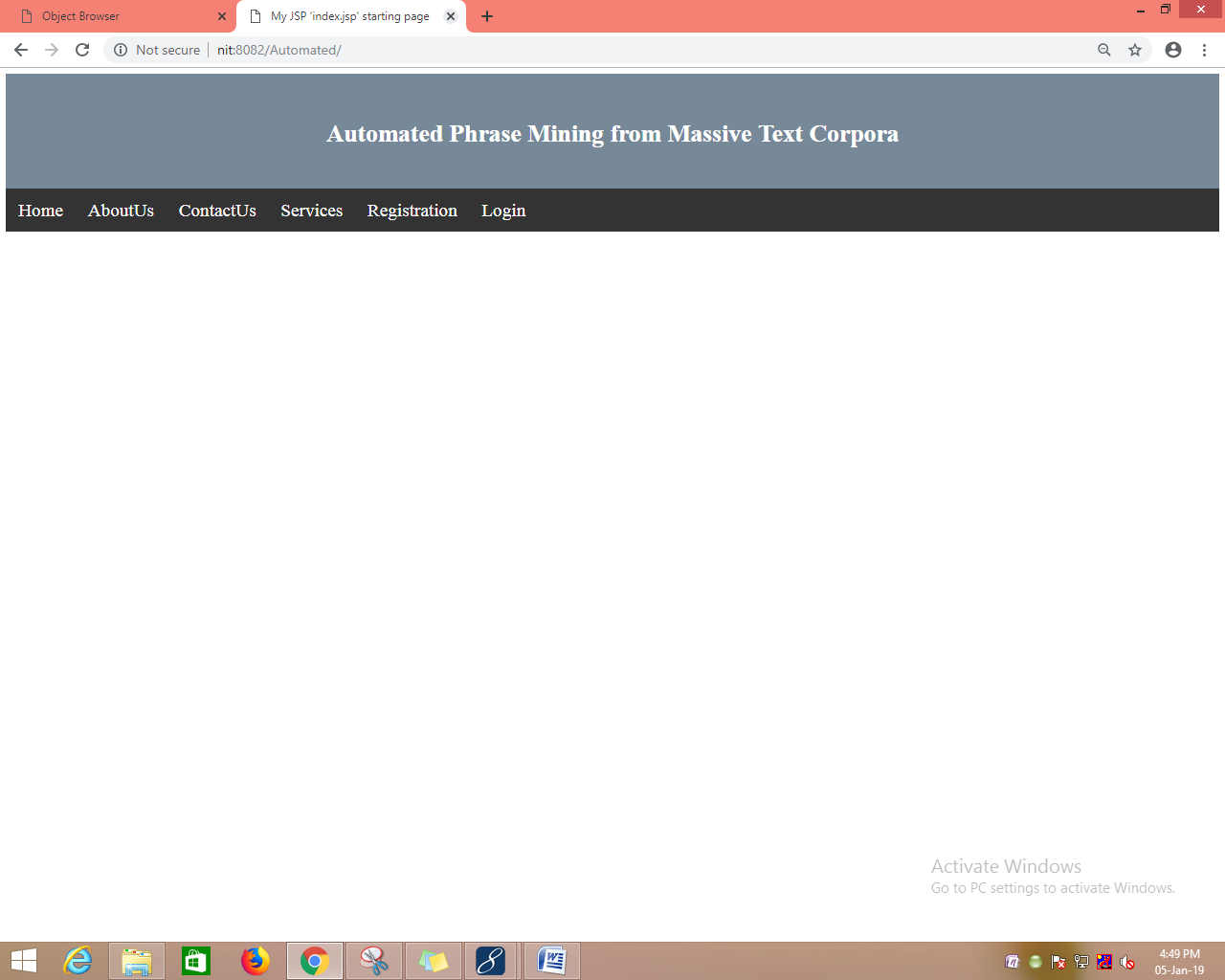
**+VE TEST CASES**

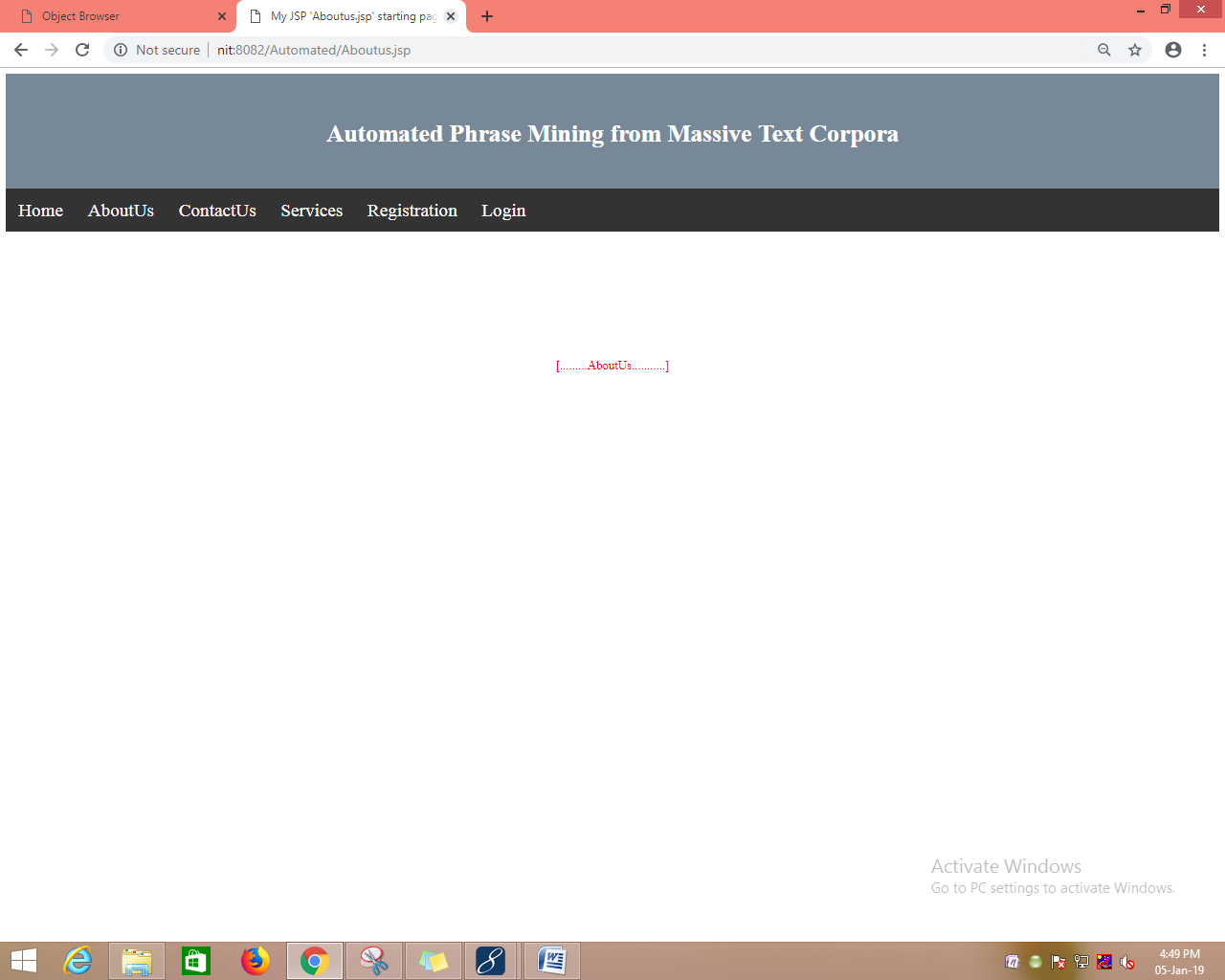
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S .No** | **Test case Description** | **Actual value** | **Expected value** | **Result** |
| **1** | Create new user registration process | Enter the personal info and address info. | Update personal info and address info in to oracle database successfully | True |
| **2** | Enter the username and password | Verification of login details. | Login Successfully | True |
| **3** | Add candidates Details | Enter all fields | Web data uploaded successfully | True |
| **4** | Search for Candidates | Enter Name. | Display proper Candidates details | True |

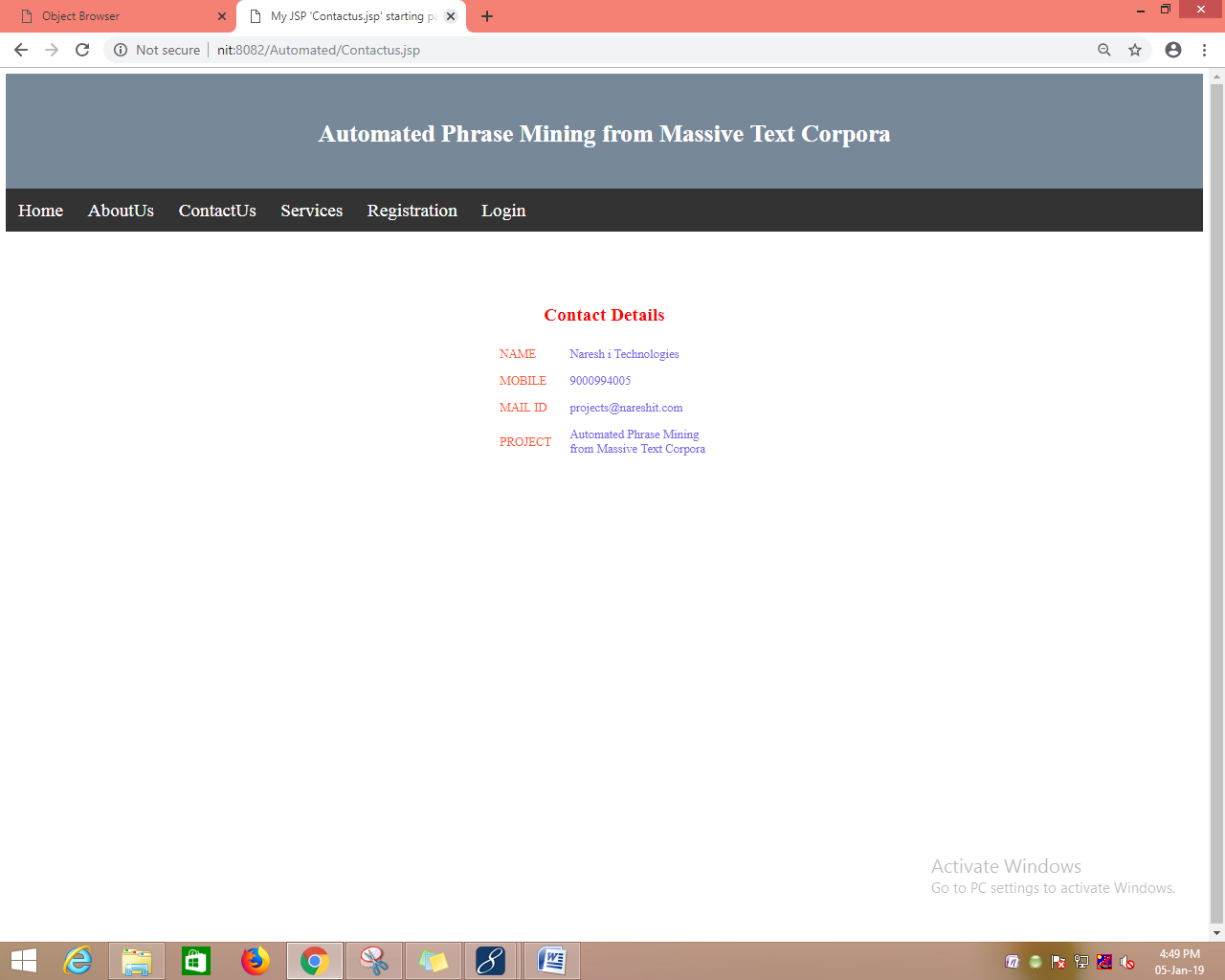
**-VE TEST CASES**

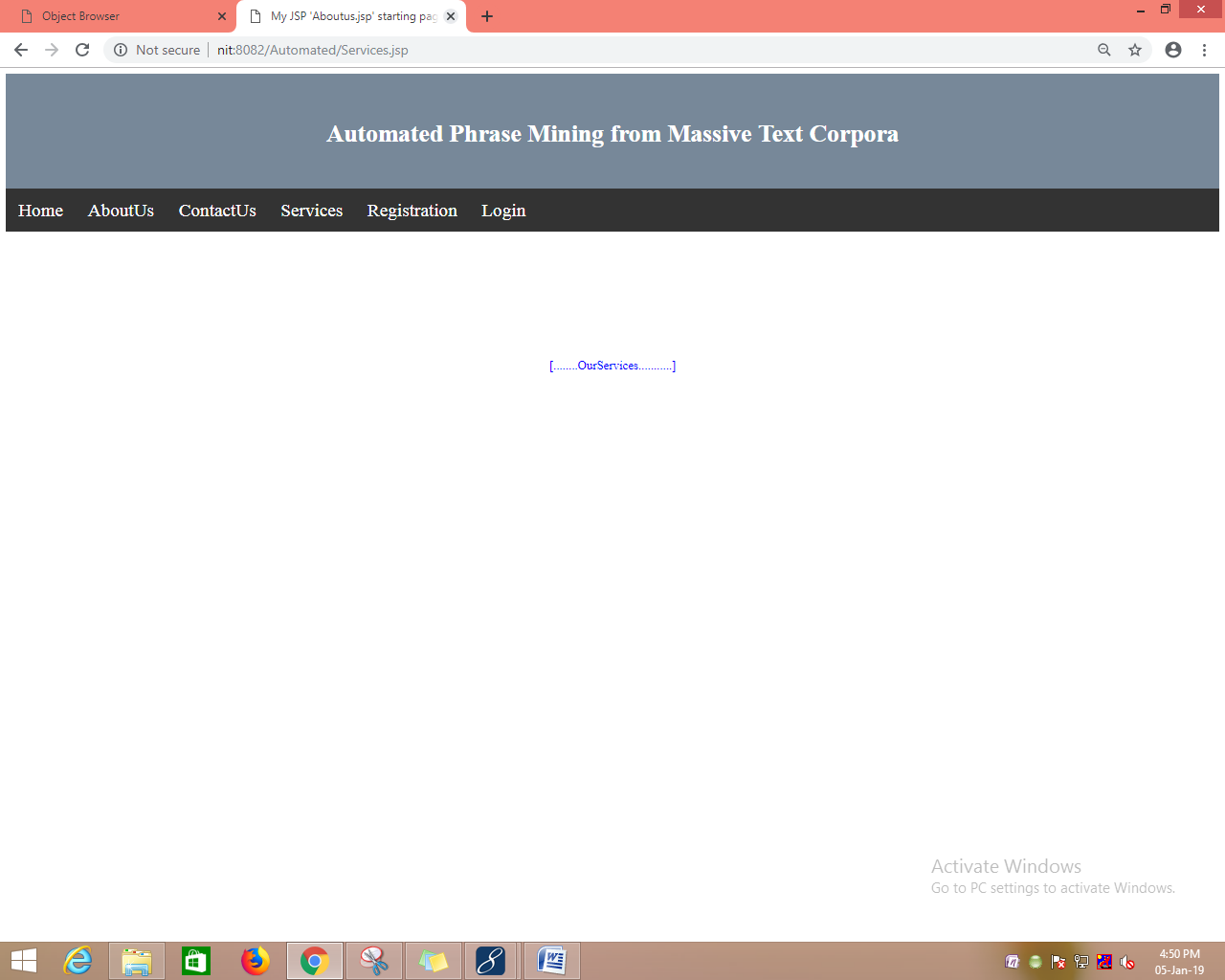
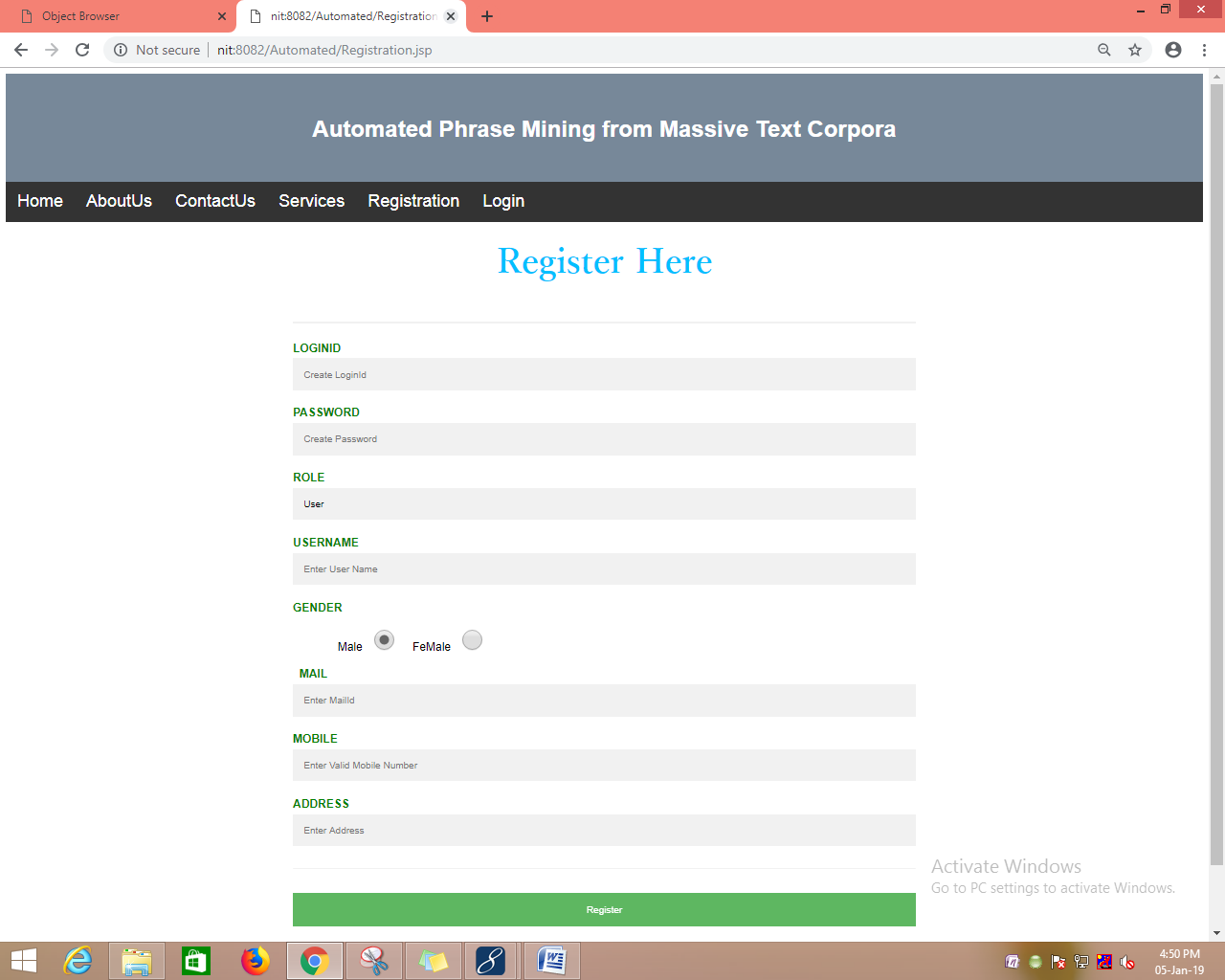
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S .No** | **Test case Description** | **Actual value** | **Expected value** | **Result** |
| **1** | Create the new user registration process | Enter the personal info and address info. | Personal info and address info its not update into database successfully. | False |
| **2** | Enter the username and password | Verification of login details. | Login failed | False |
| **3** | View Candidates Details. | Enter Proper Candidate name. | Name is Mismatch in database. | False |
| **4** | Add Candidates Details. | Enter Proper Details. | Data is not Stored in database | False |

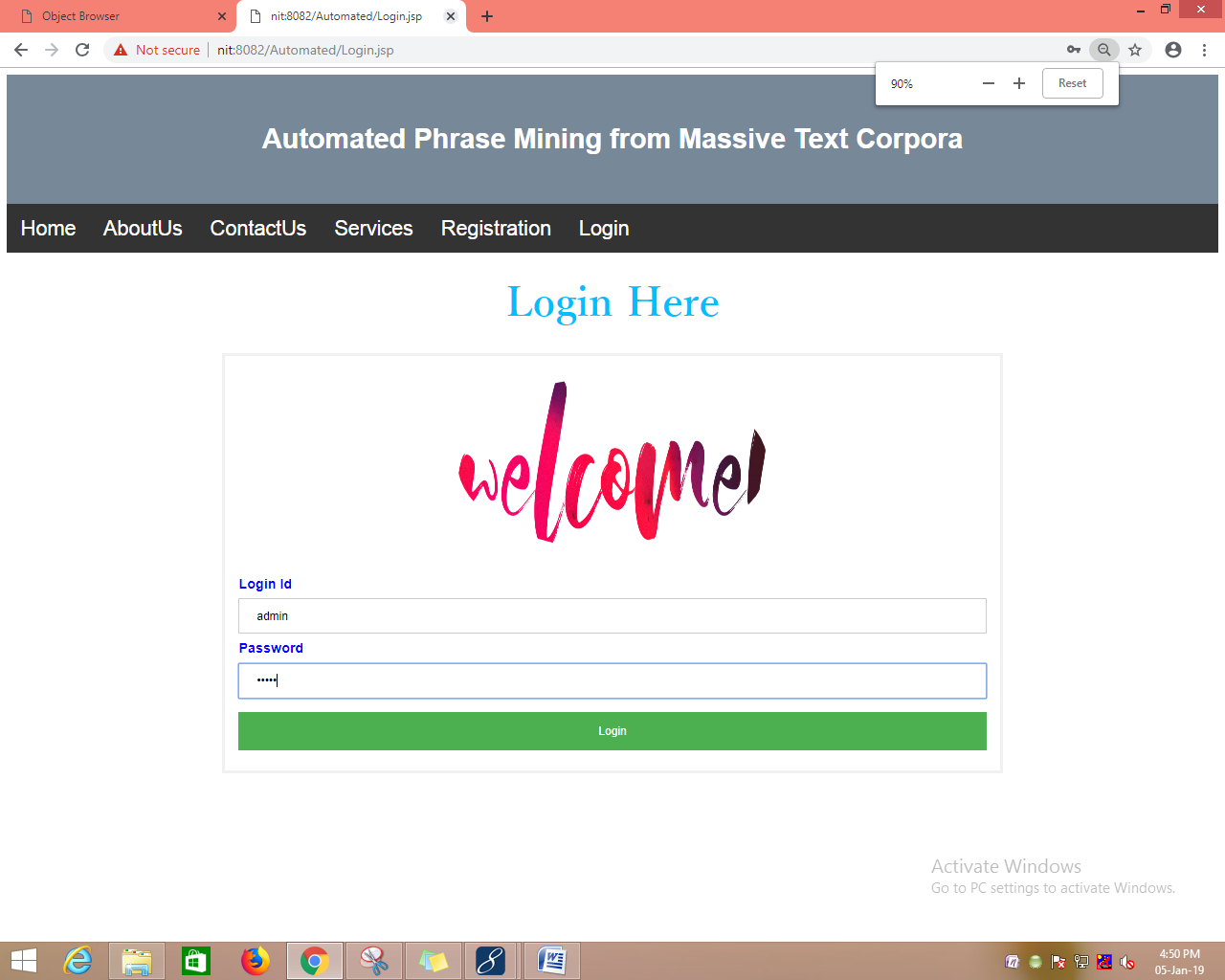
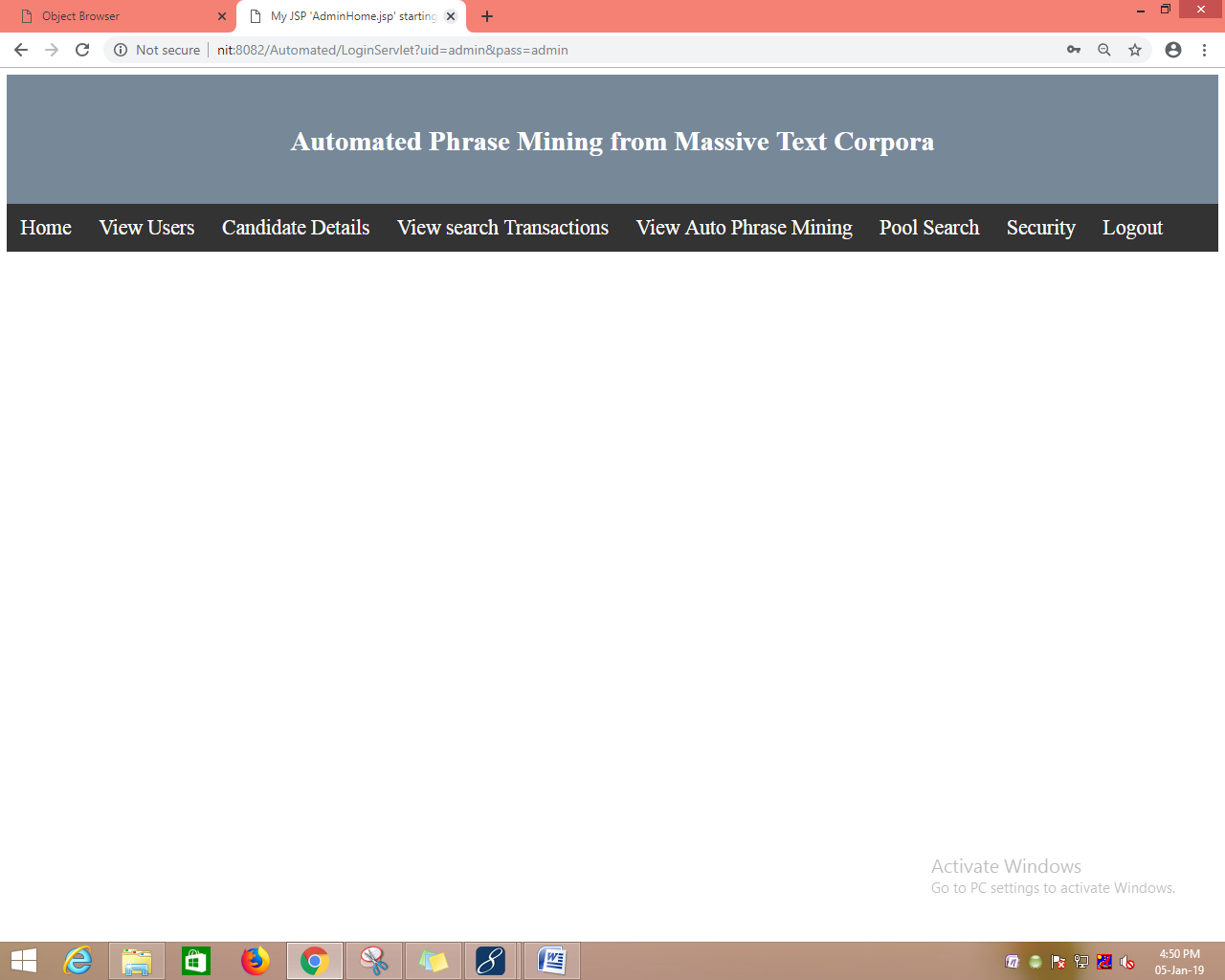
**10. Output Screens**

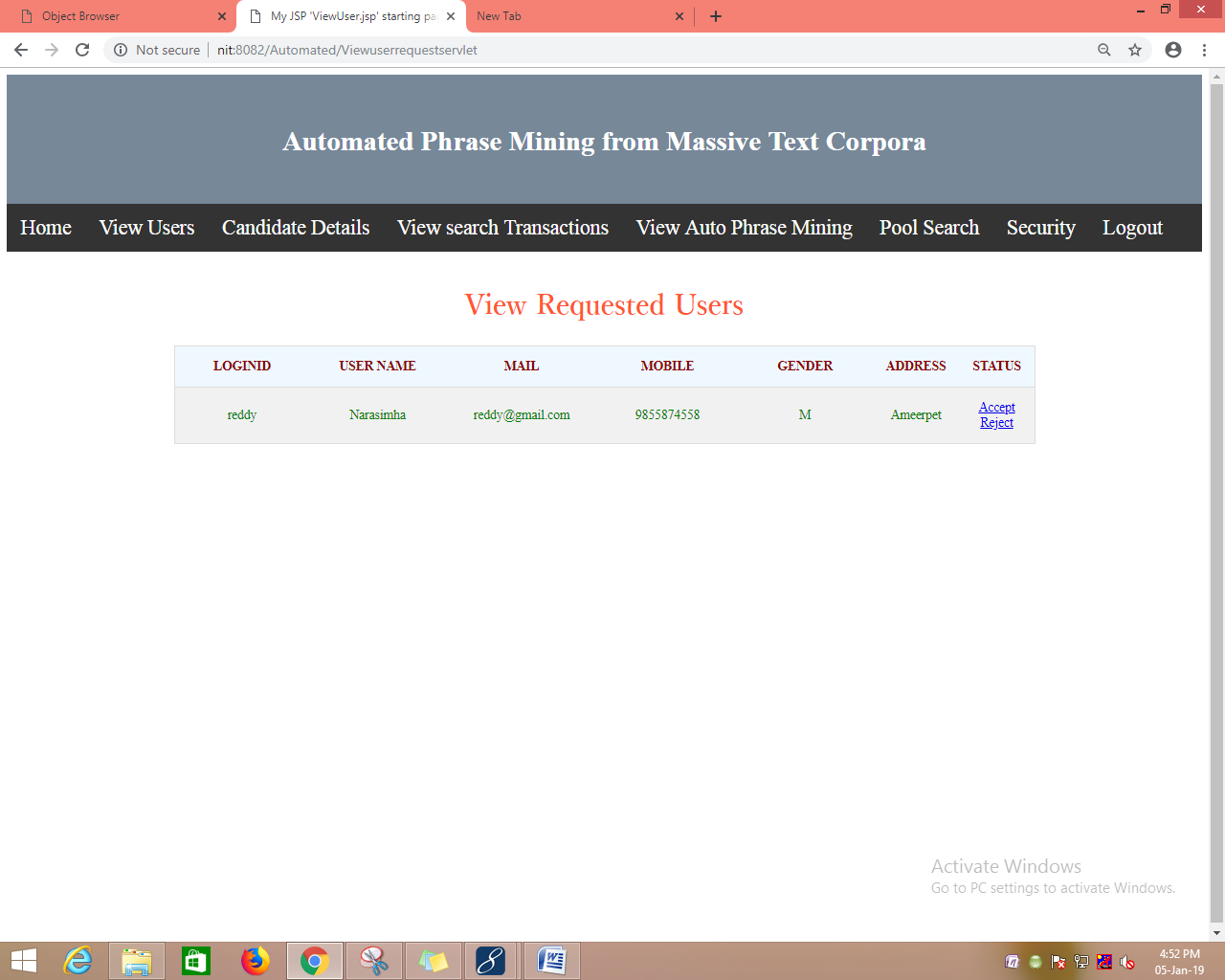
Home Page:

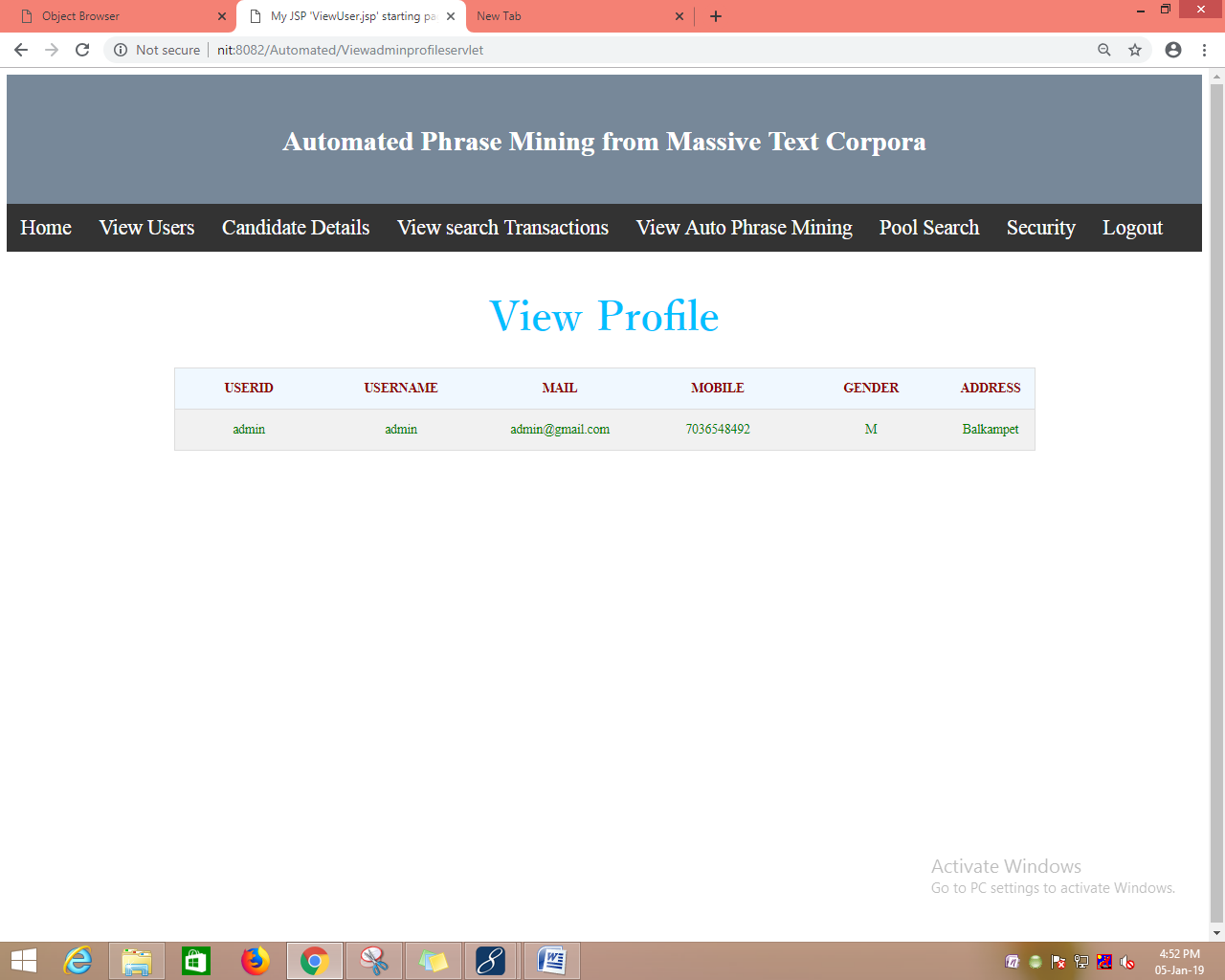
AboutUs Page:

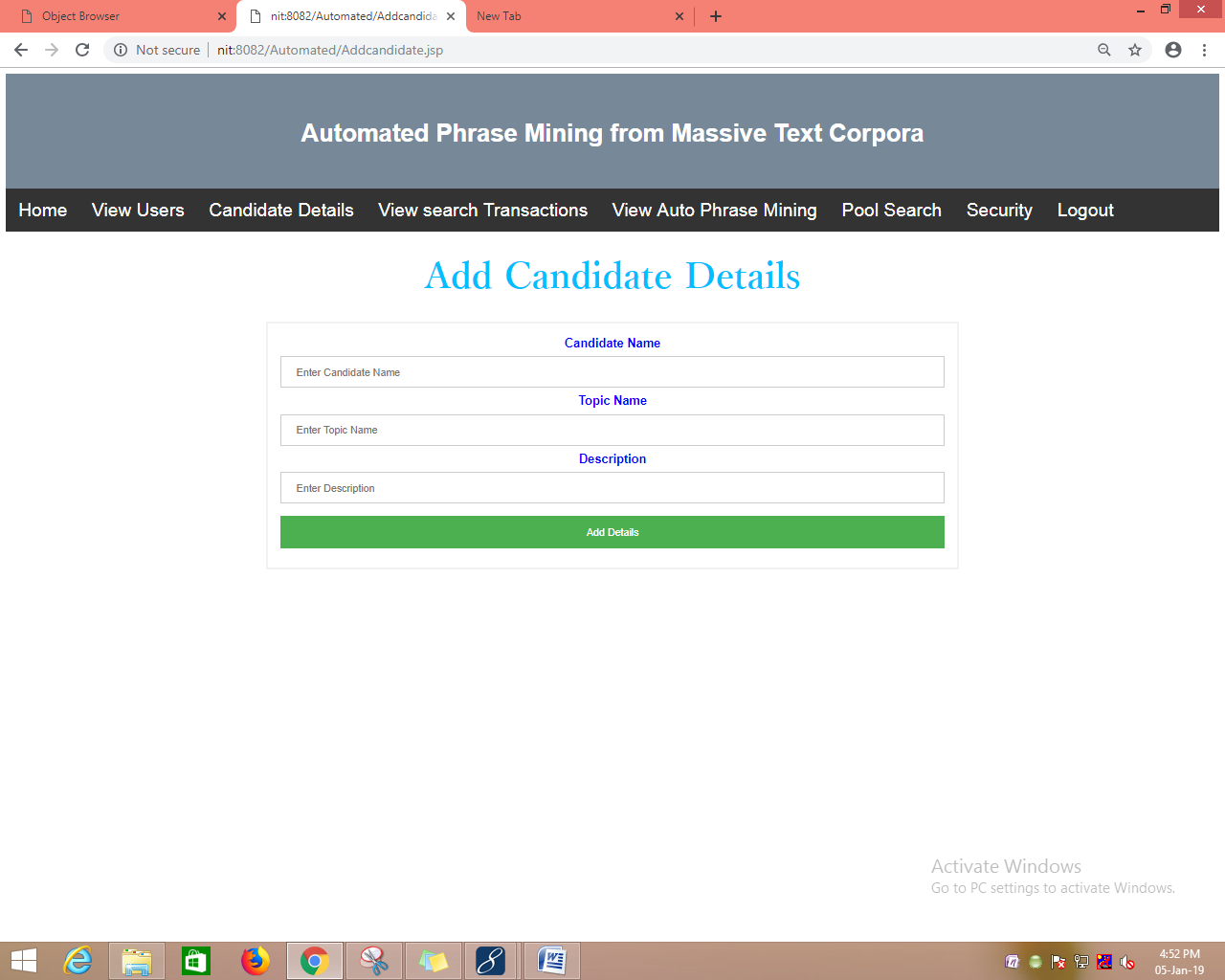
ContactUs Page:

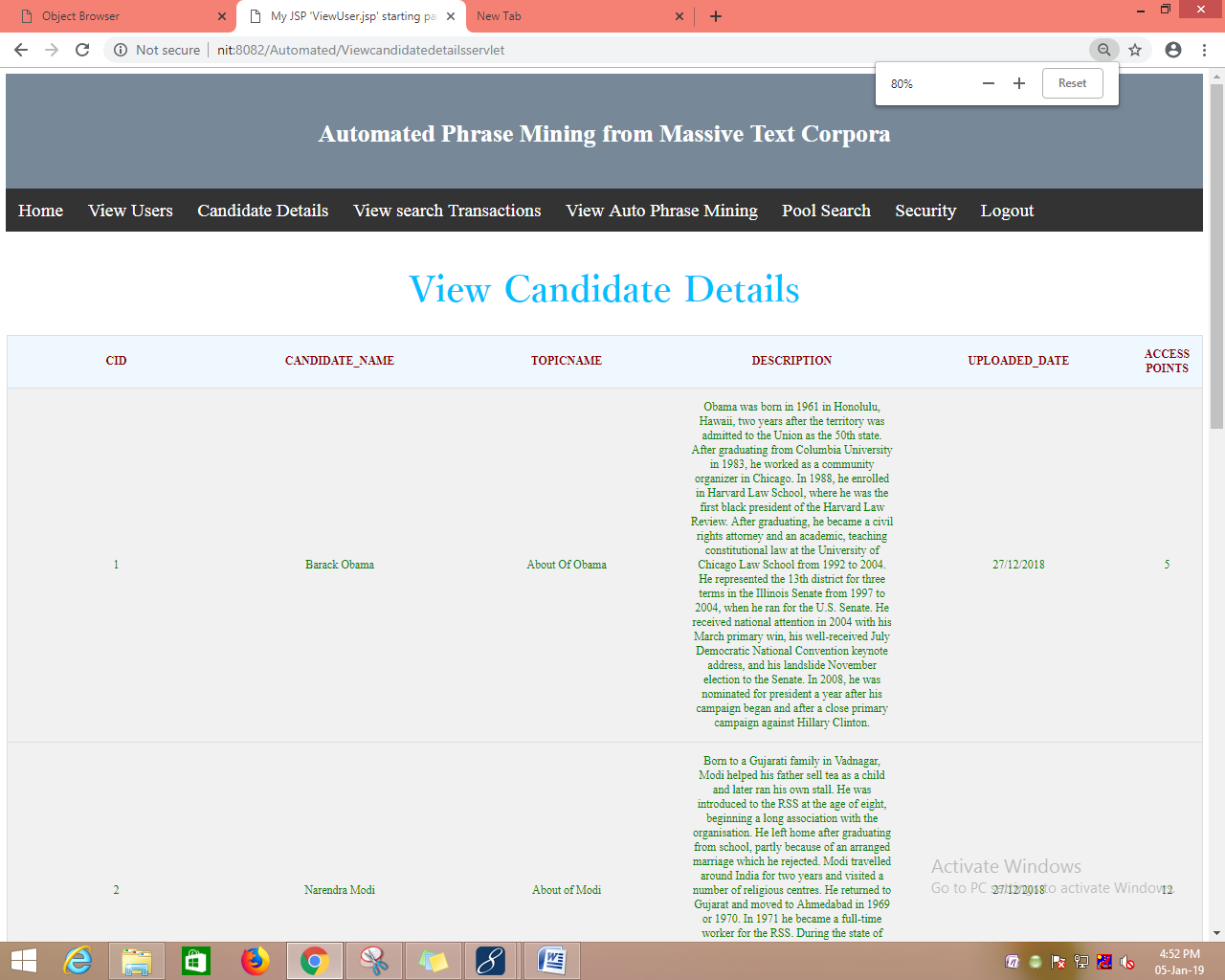
Services Page:User Registration Page:

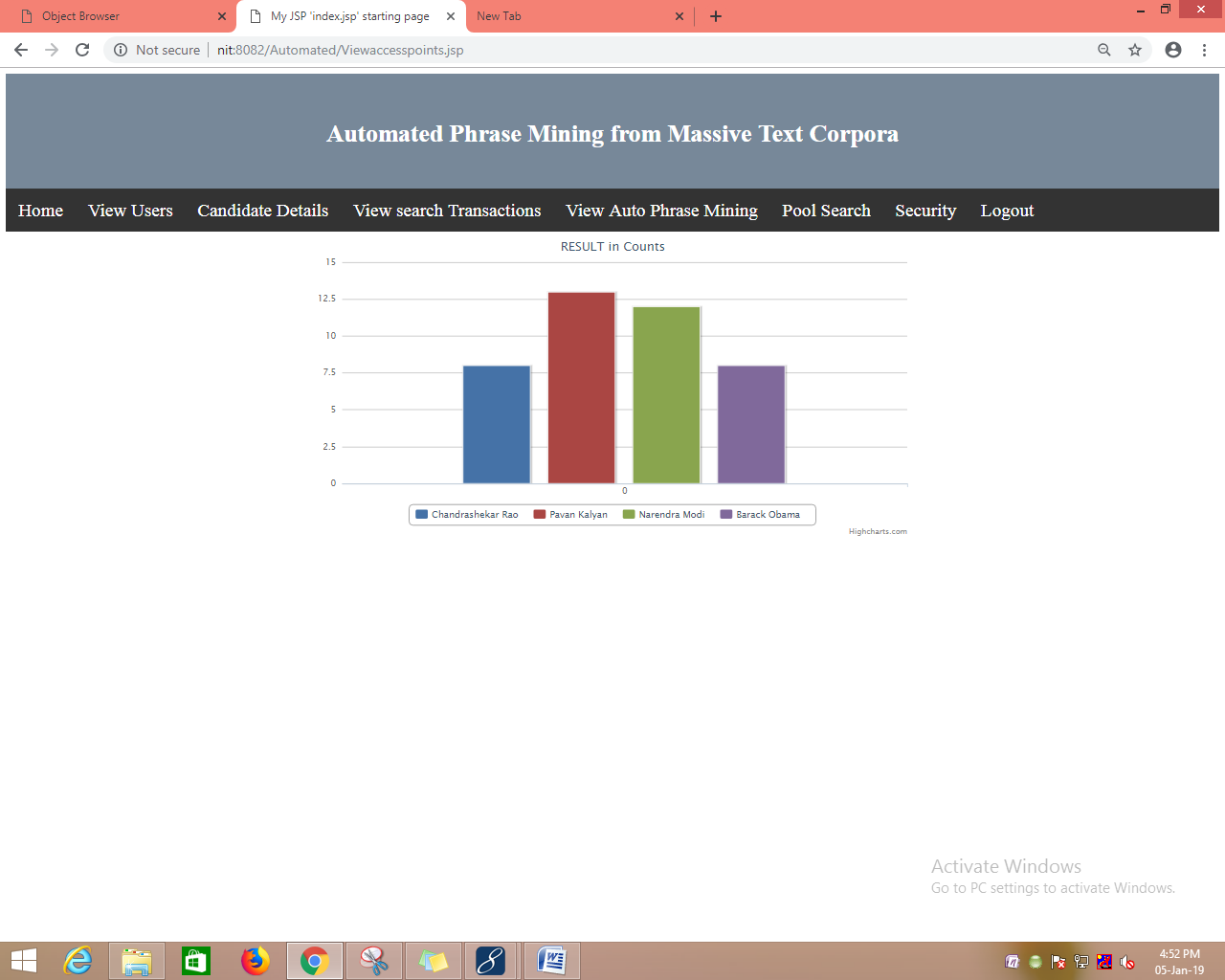
Admin Login Page:Admin Home Page:

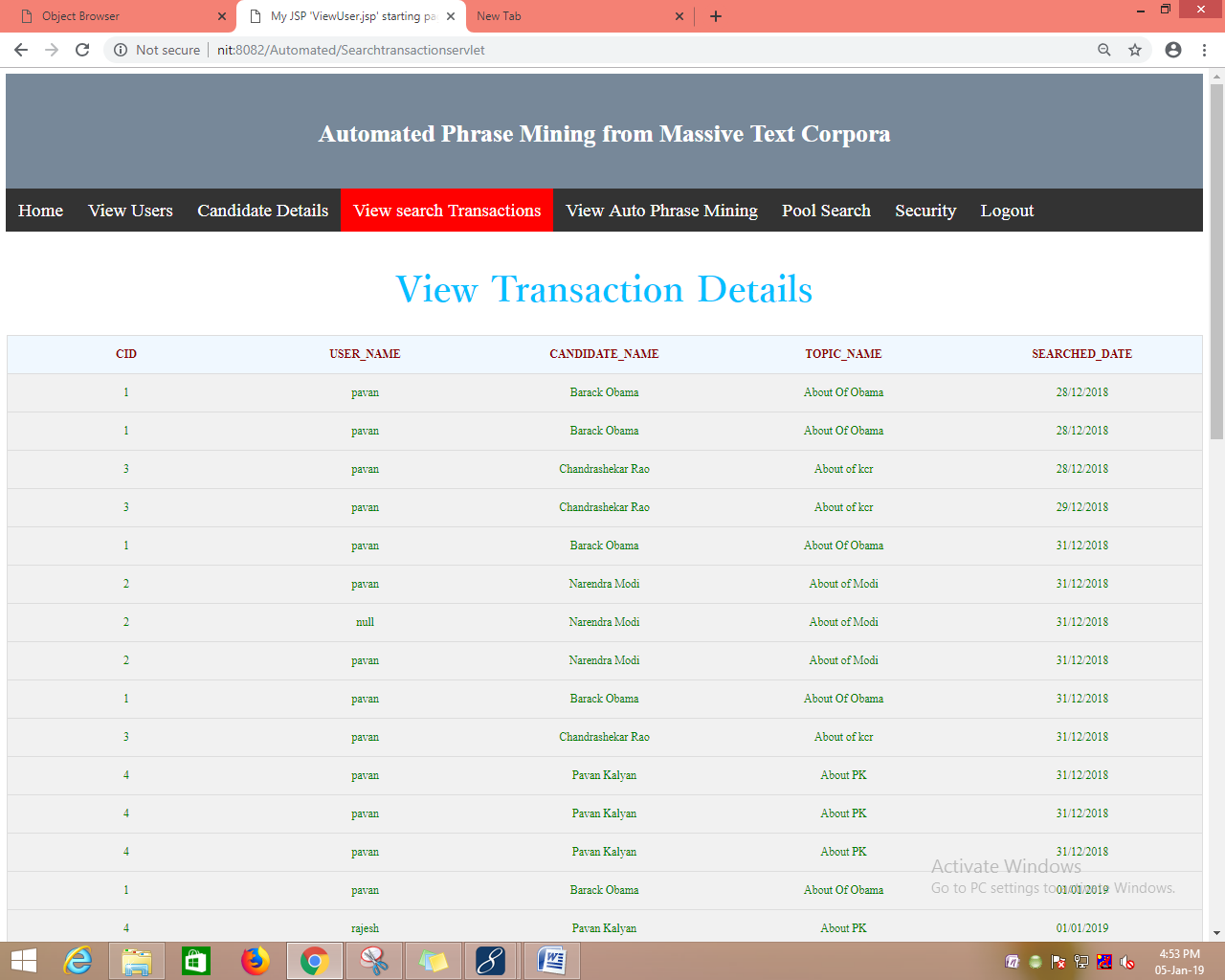
View Requested users:

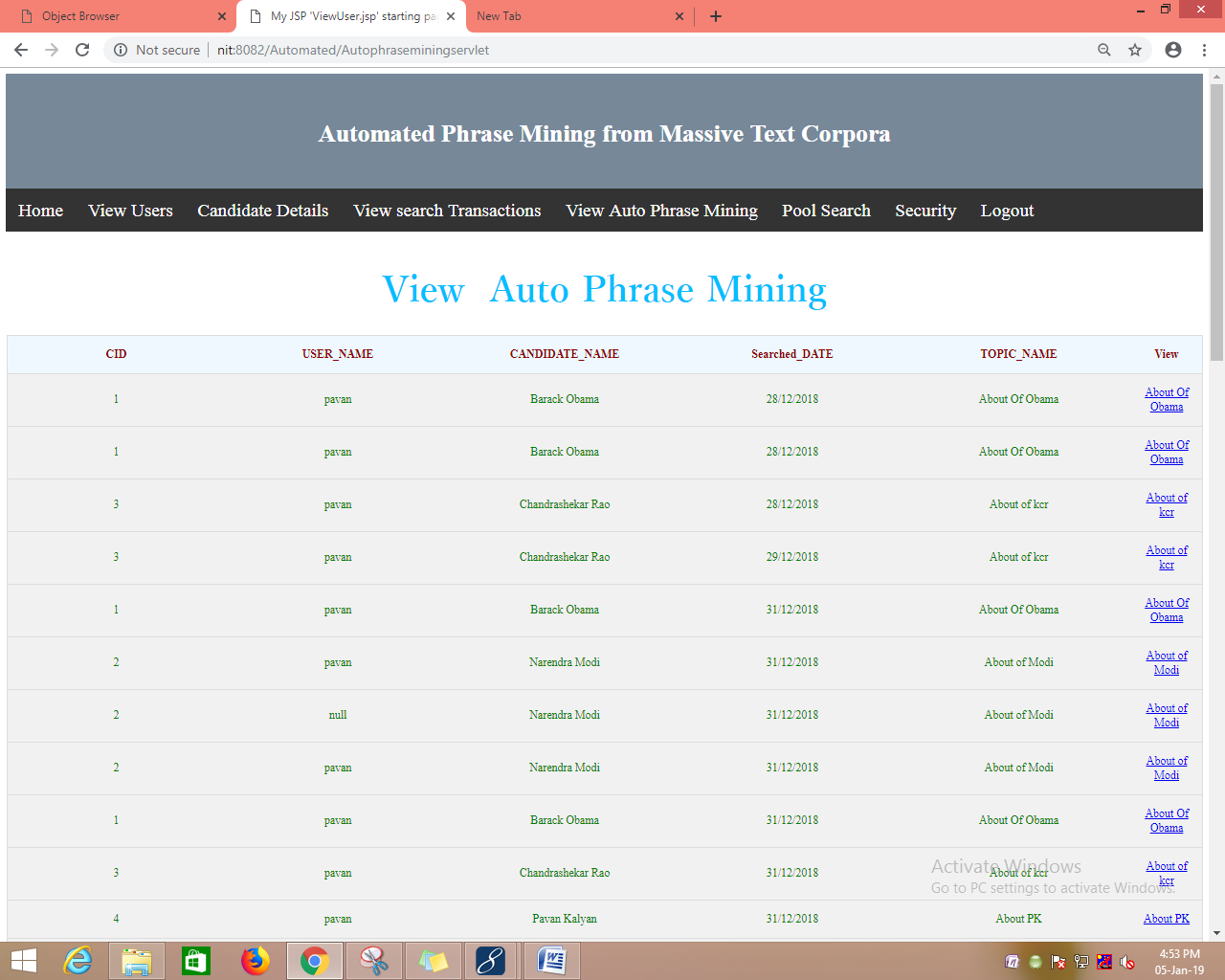
View Profile:

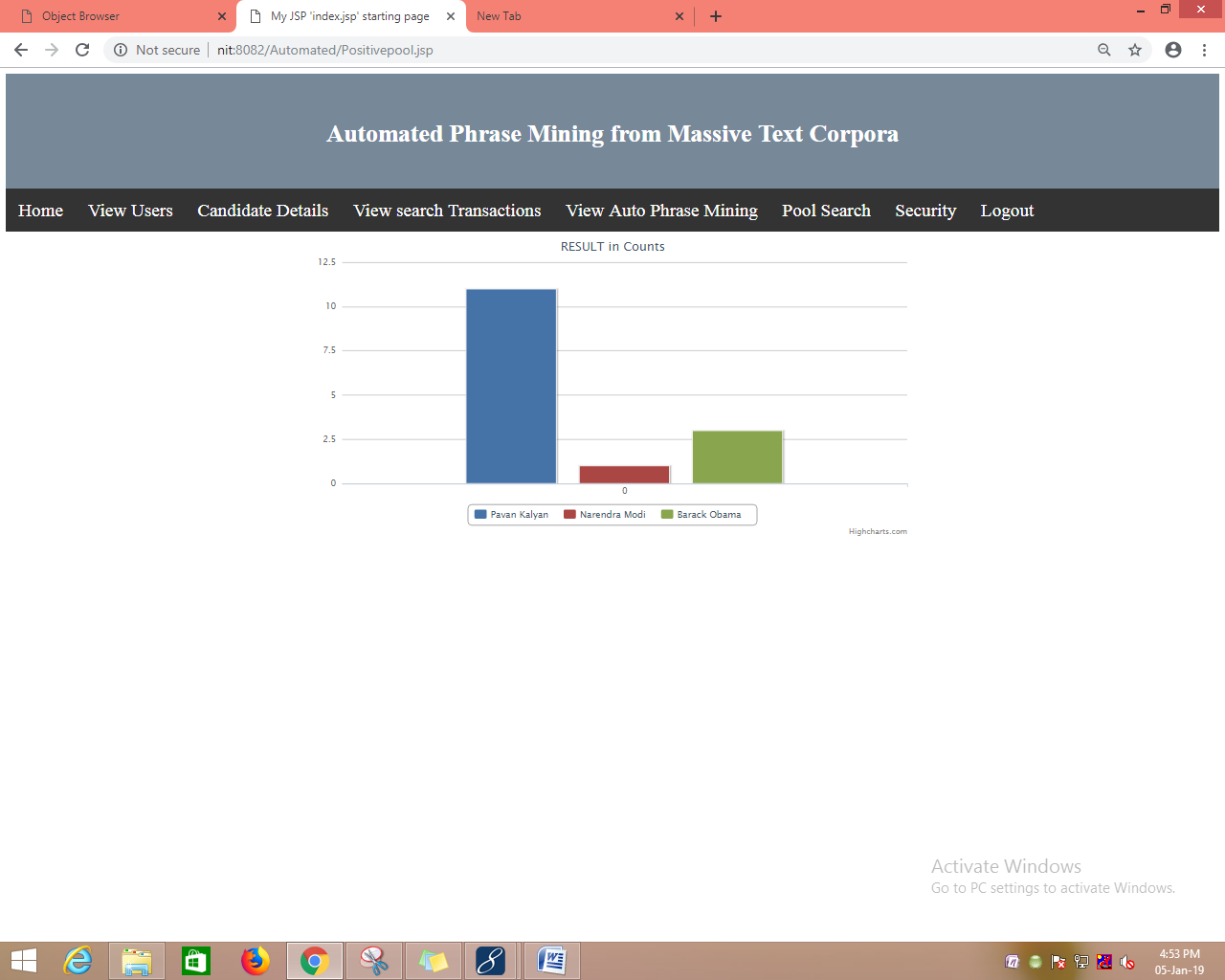
Add candidate Details:

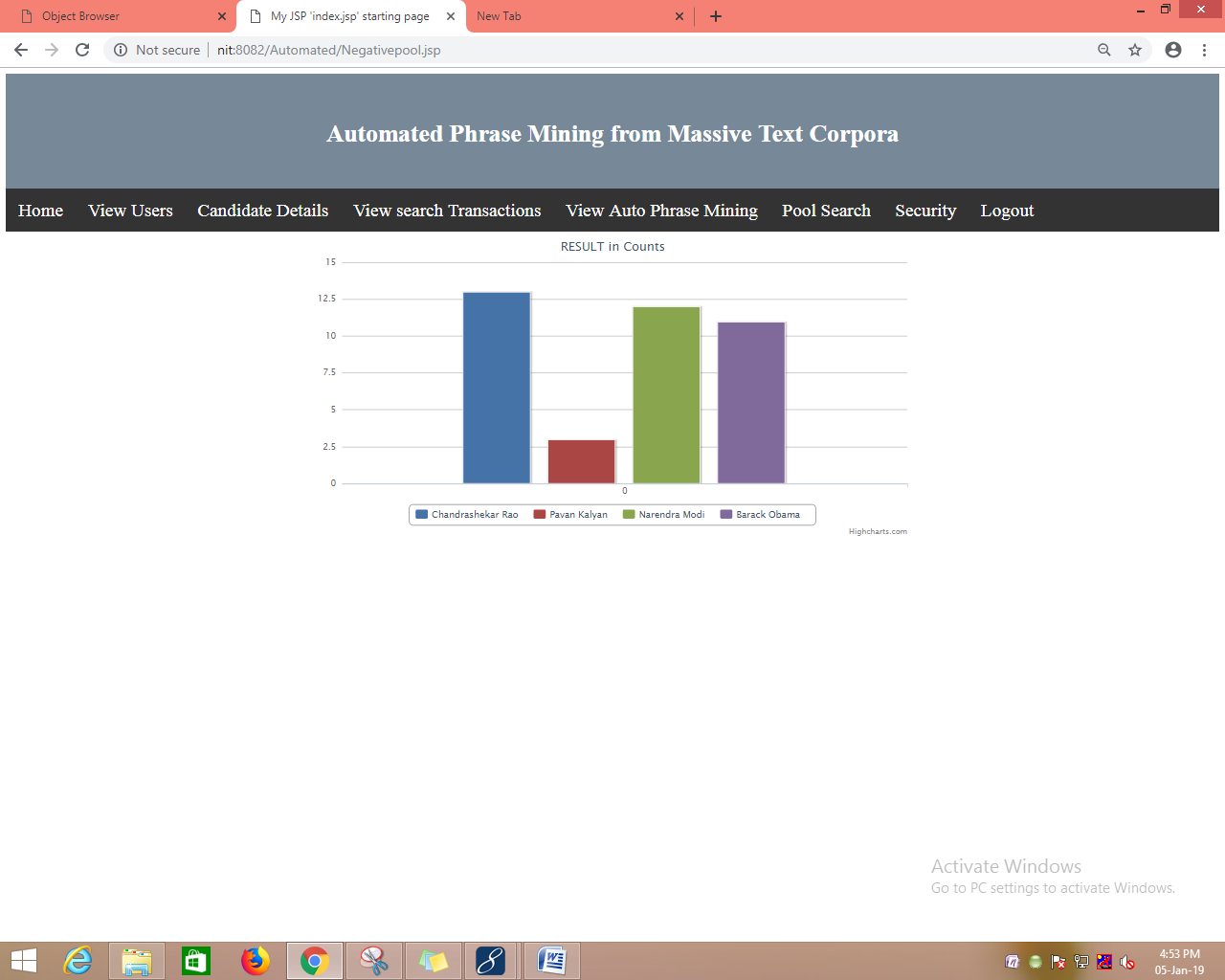
View Candidates Details:

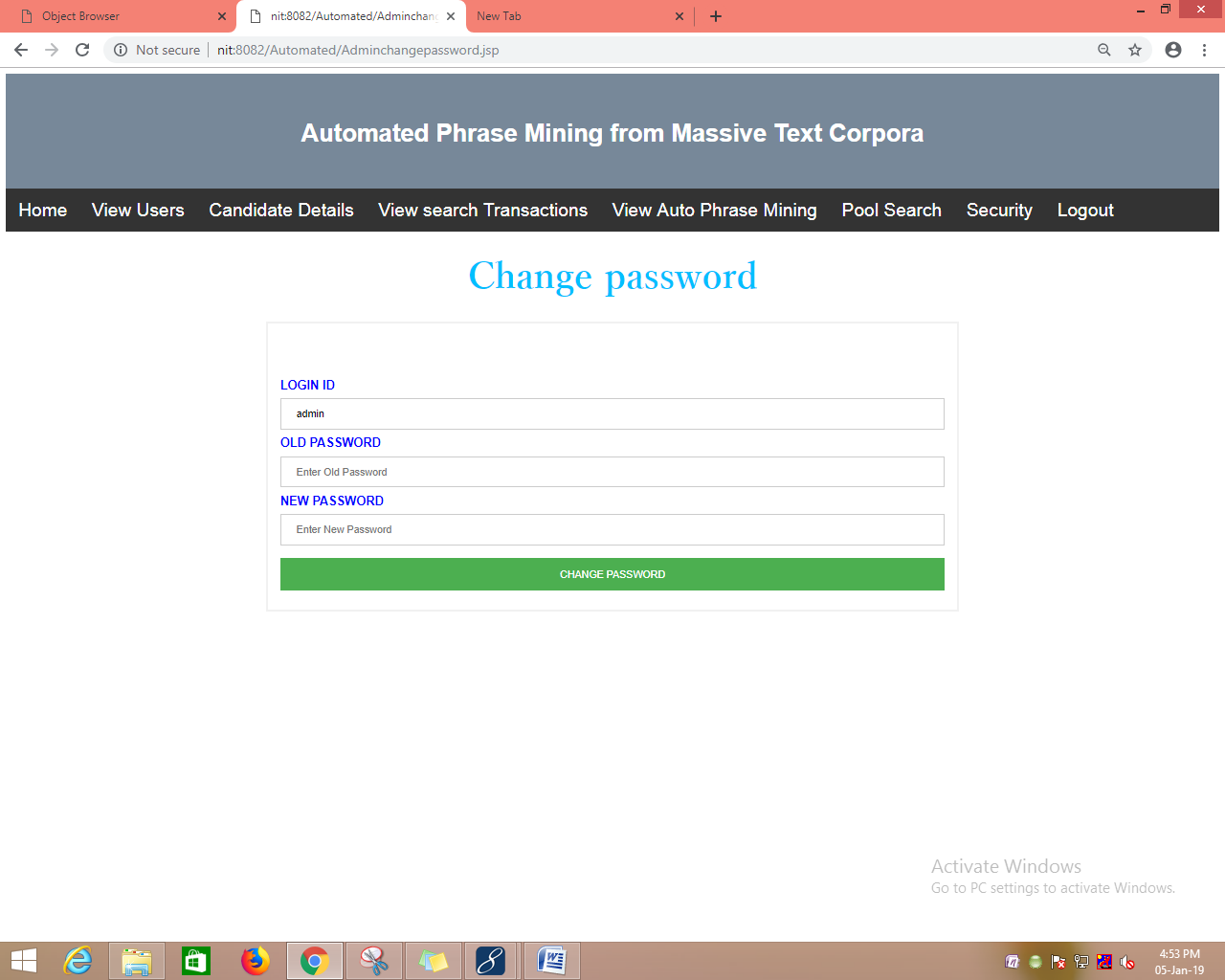
View Access Points:

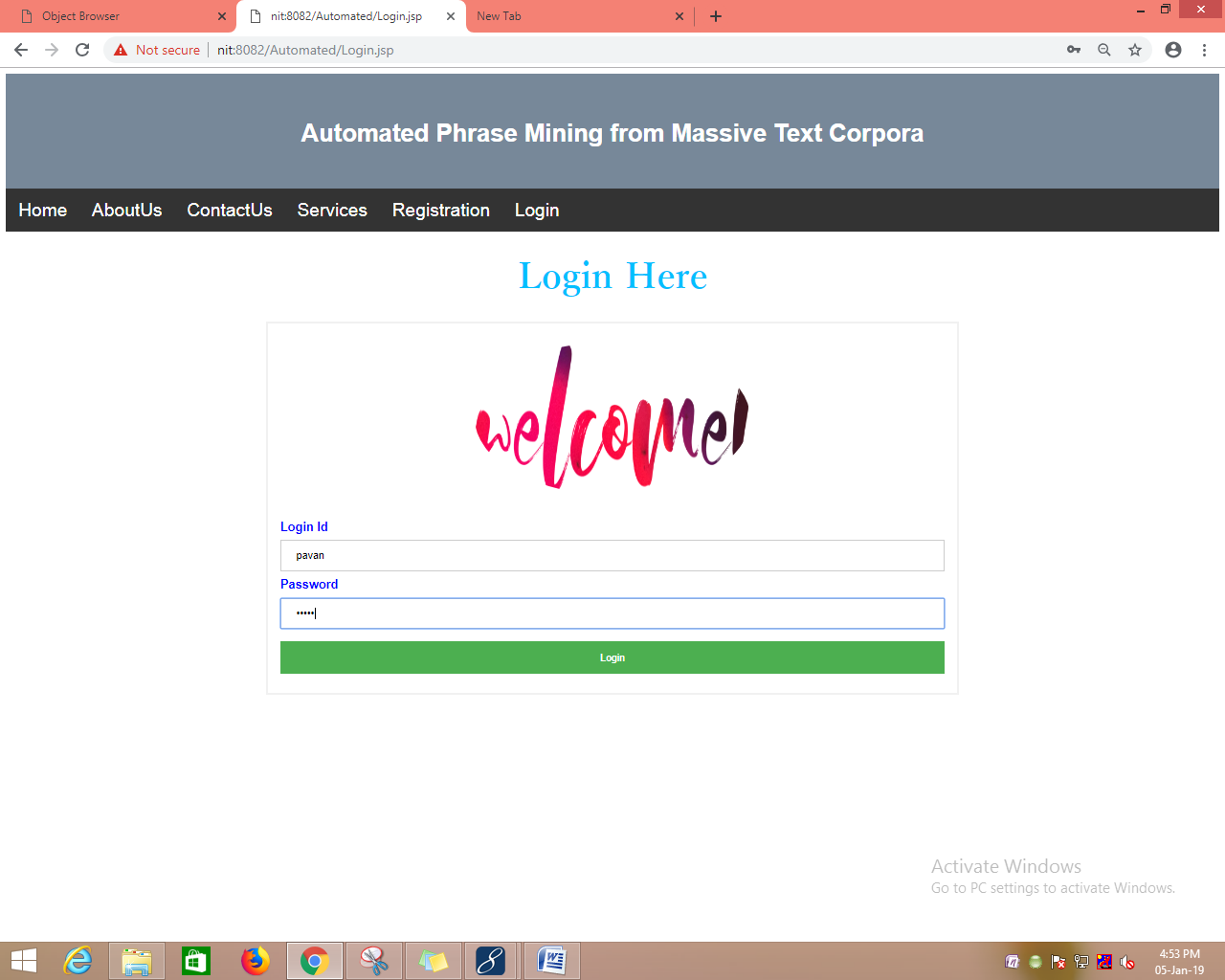
View Transaction Details:

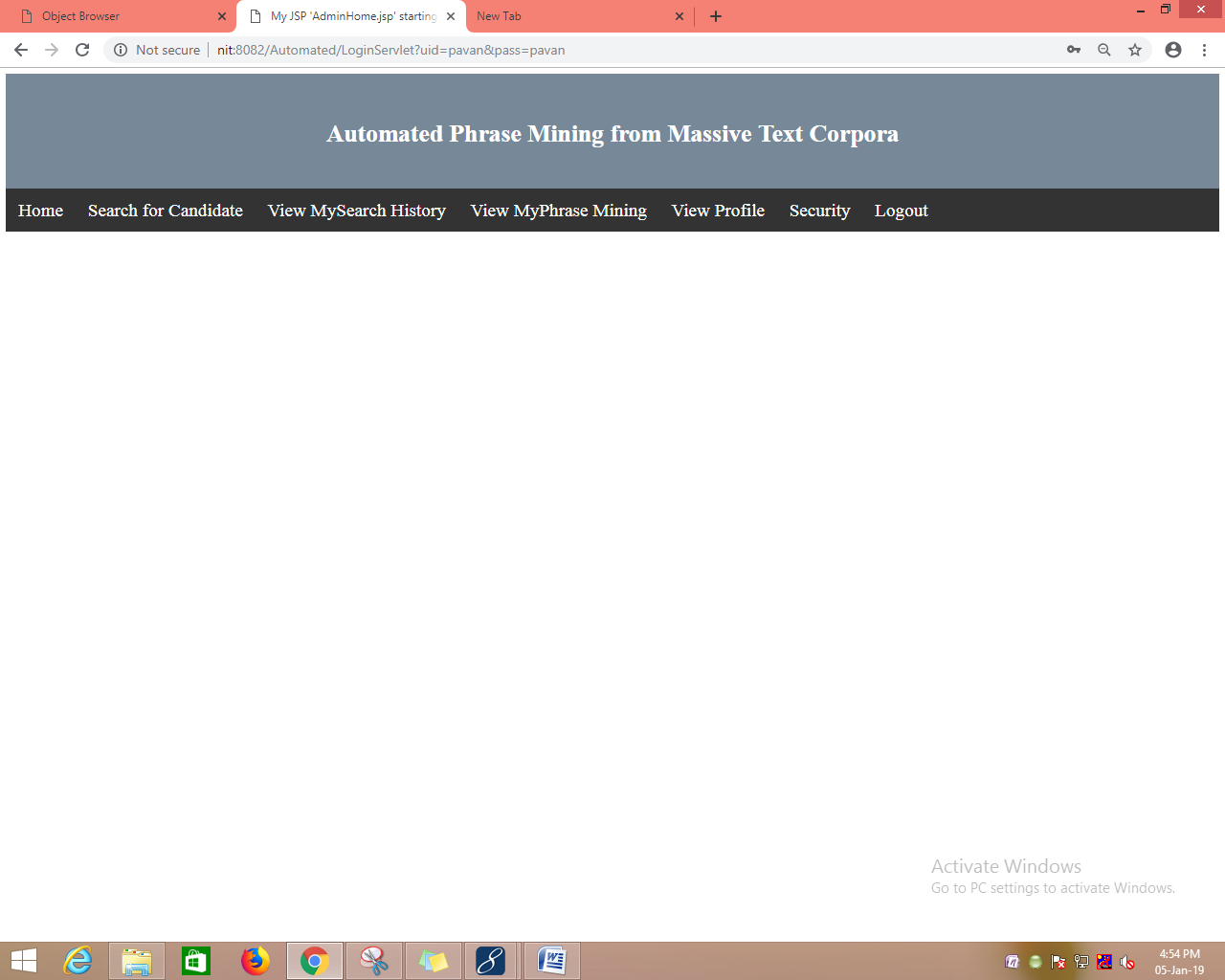
View Auto PhraseMining:

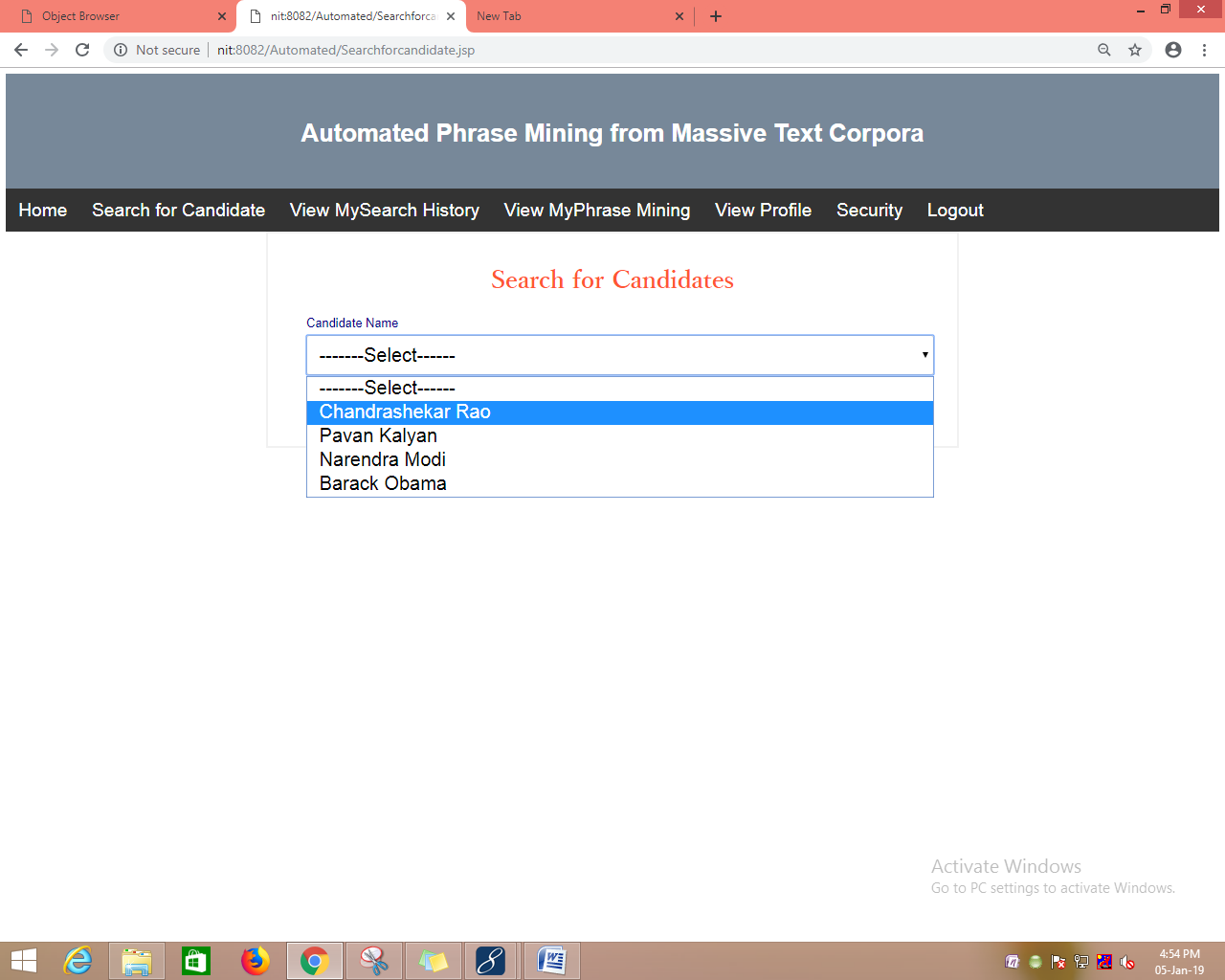
Positive Pool search:

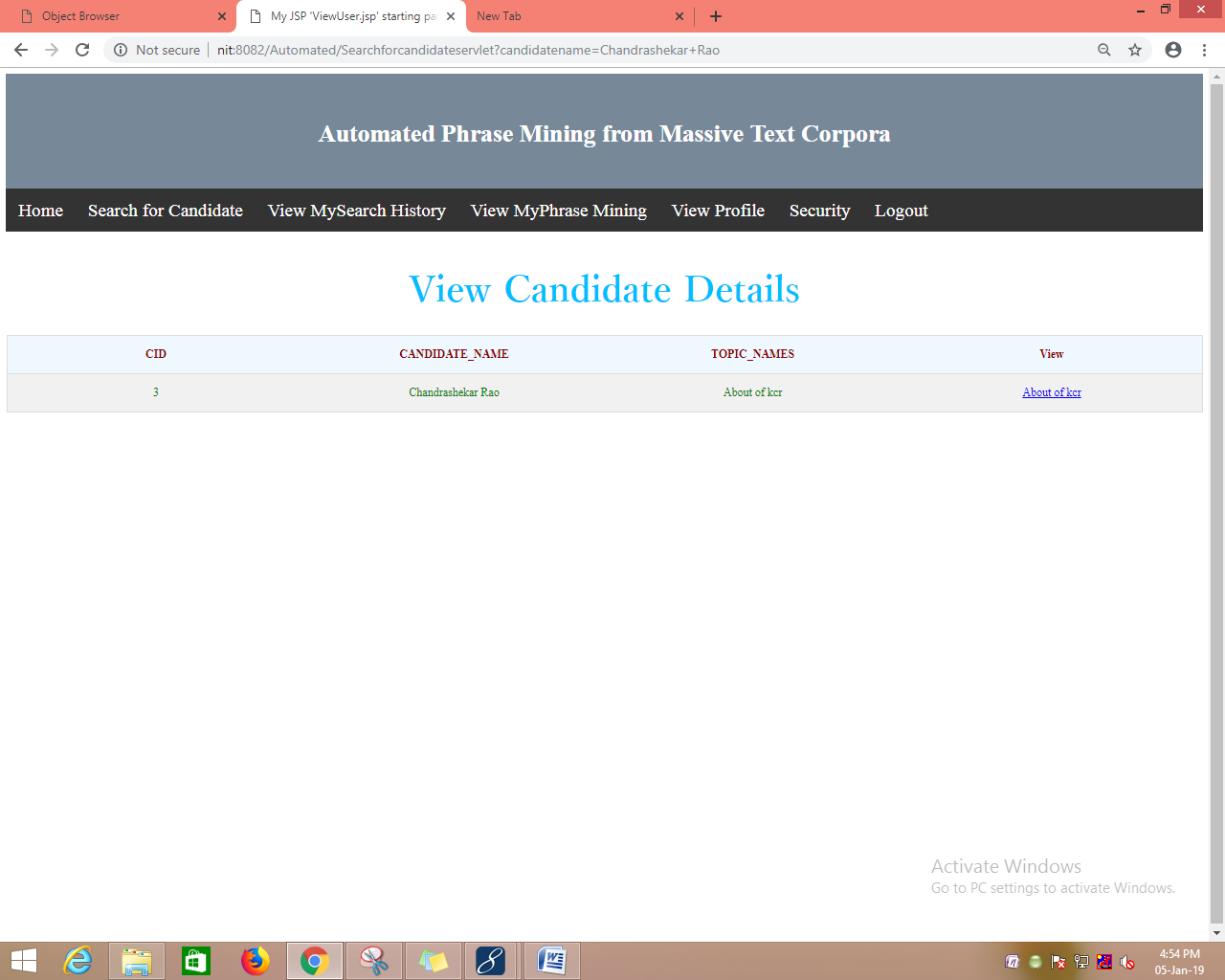
Negative Pool search:

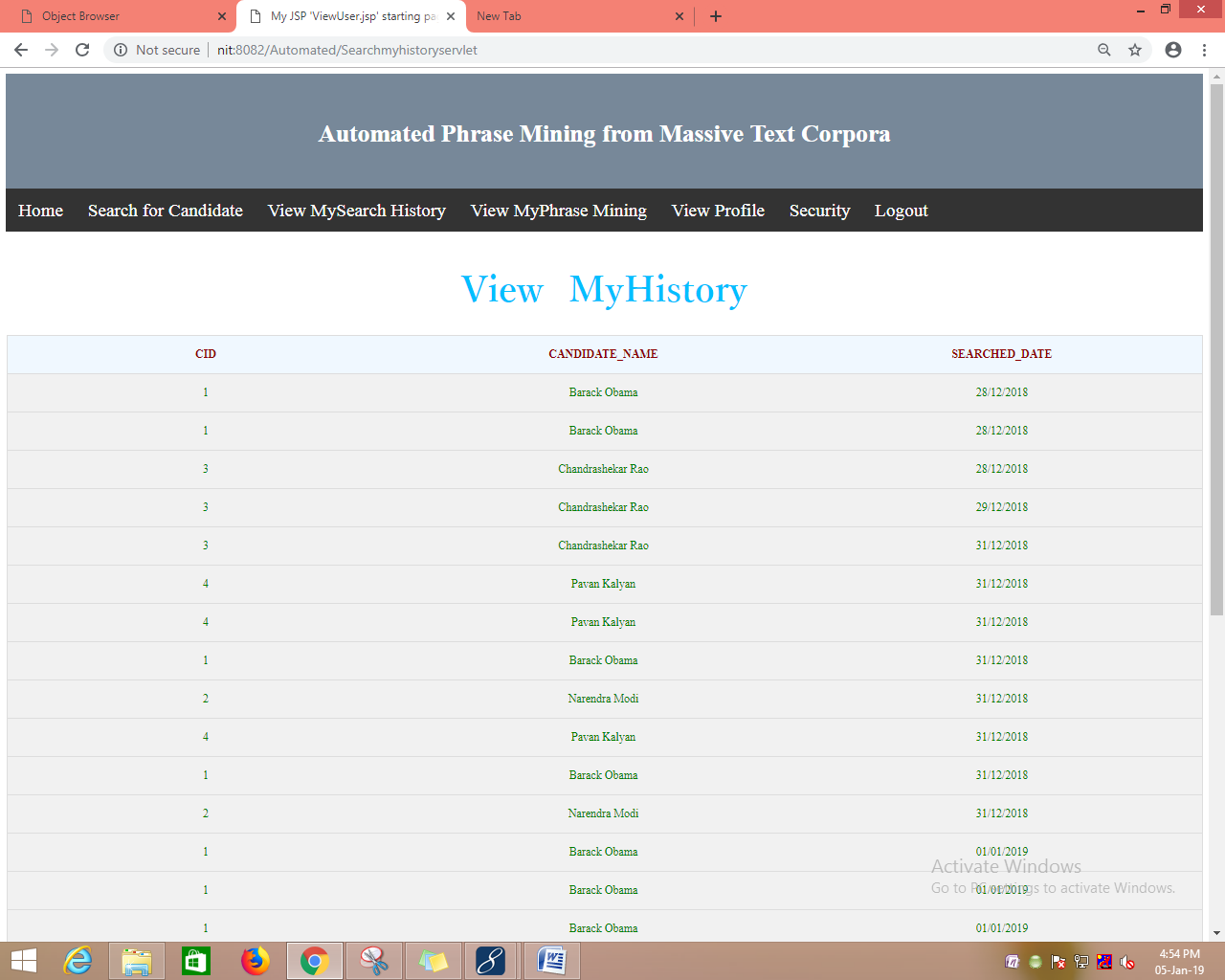
Admin Change Password:

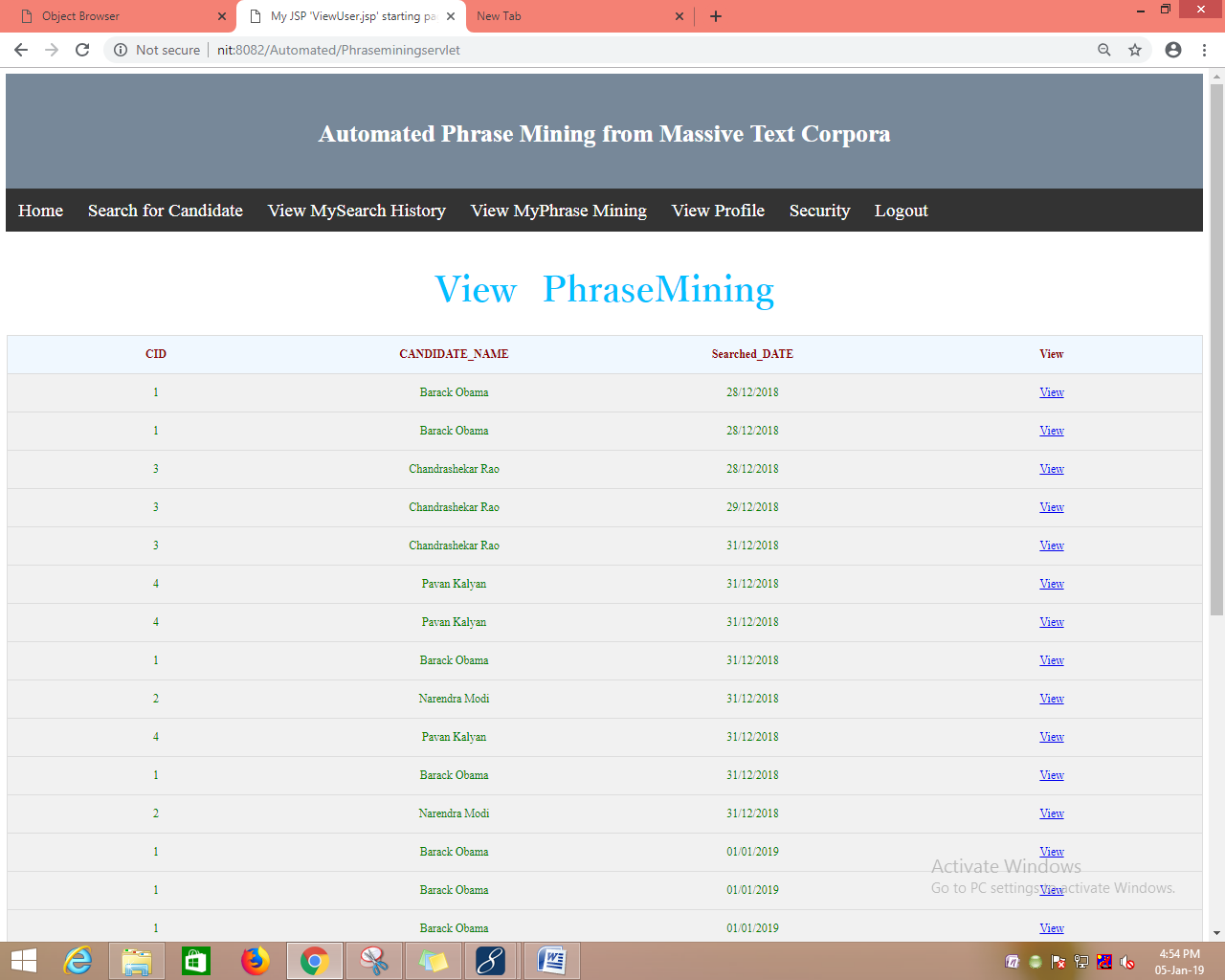
User LoginPage:

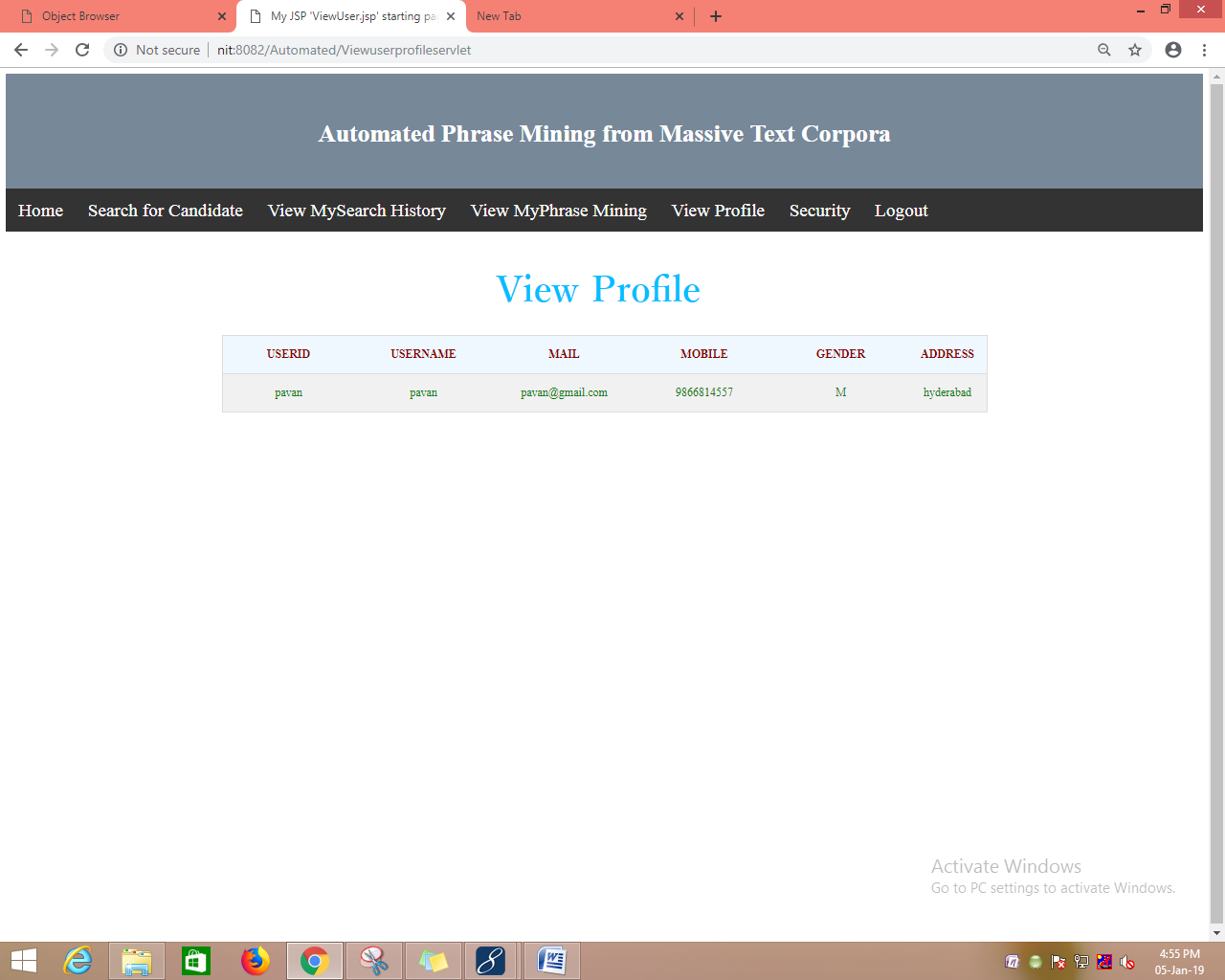
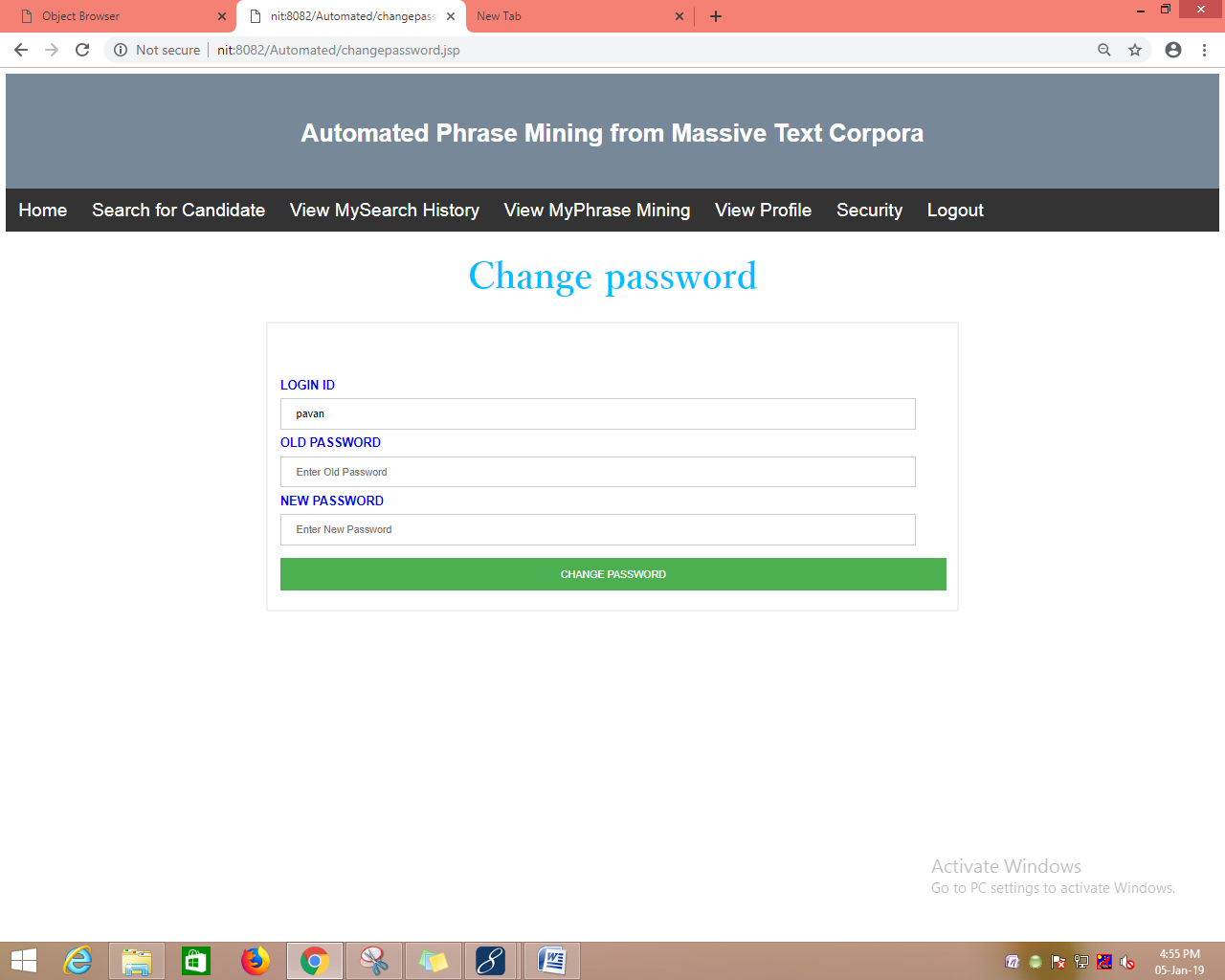
User Homepage:

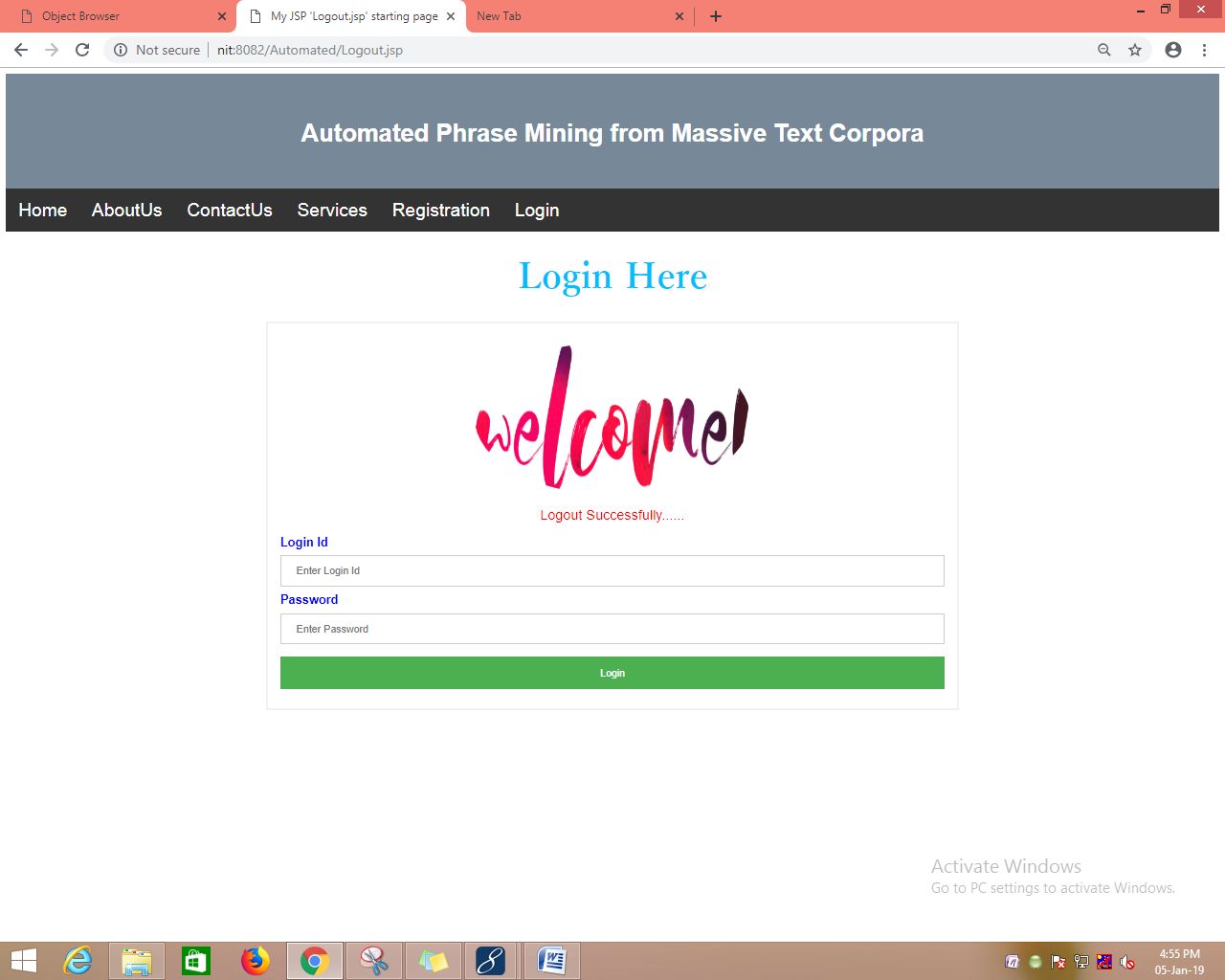
Search Candidates:

View Candidate Details:

View MyHistory:

View PhraseMining:

User Profile:User Change password:

Logout Page

**11. Reports:**

**12. Conclusion**

An automated phrase mining framework with two novel techniques: the robust positive only distant training and the POS-guided phrasal segmentation incorporating part-of-speech (POS) tags, for the development of an automated phrase mining framework Auto Phrase. Our extensive experiments show that Auto Phrase is domain-independent, outperforms other phrase mining methods, and supports multiple languages effectively, with minimal human effort.

**13. Future Enhancements:**

**14. References**

1.K. Ahmad, L. Gillam, L. Tostevin, et al. University of surrey participation in trec8: eirdness indexing for logical document extrapolation and retrieval (wilder). In TREC, pages 1–8, 1999.

2. A. Allahverdyan and A.Galstyan.Comparative analysis of viterbi training and maximum likelihood estimation for hmms. In NIPS, pages 1674–1682, 2011.

3. T. Baldwin and S. N. Kim. Multiword expressions. Handbook of Natural Language Processing, second edition. Morgan and Claypool, 2010.

4.S. Bedathur, K. Berberich, J. Dittrich, N. Mamoulis, and G. Weikum.Interesting-phrase mining for ad-hoc text analytics. Proc. VLDB Endow., 3(1-2):1348–1357, Sept. 2010.

5. L. Breiman. Randomizing outputs to increase prediction accuracy.Machine learning, 40(3):229–242, 2000.

6. K.-h. Chen and H.-H. Chen. Extracting noun phrases from largescale texts: A hybrid approach and its automatic evaluation. In Proceedings of the 32Nd Annual Meeting on Association for Computational Linguistics, ACL ’94, pages 234–241, Stroudsburg, PA, USA, 1994. Association for Computational Linguistics.

7.M. Danilevsky, C. Wang, N. Desai, X. Ren, J. Guo, and J. Han.Automatic construction and ranking of topical keyphrases on collections of short documents. In SDM, 2014.

8. M.-C. De Marneffe, B. MacCartney, C. D. Manning, et al. Generating typed dependency parses from phrase structure parses. In Proceedings of LREC, volume 6, pages 449–454, 2006.

9. P. Deane. A nonparametric method for extraction of candidate phrasal terms. In Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, ACL ’05, pages 605– 613, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics.

10.A. El-Kishky, Y. Song, C. Wang, C. R. Voss, and J. Han. Scalable topical phrase mining from text corpora. Proc. VLDB Endow.,8(3):305–316, Nov. 2014.

11. D. A. Evans and C. Zhai. Noun-phrase analysis in unrestricted text for information retrieval. In Proceedings of the 34th Annual Meeting on Association for Computational Linguistics, ACL ’96, pages 17–24, Stroudsburg, PA, USA, 1996. Association for Computational Linguistics.

12. G. Finch. Linguistic terms and concepts. Macmillan Press Limited, 2000.

13.K. Frantzi, S. Ananiadou, and H. Mima. Automatic recognition of multi-word terms: The c-value/nc-value method. JODL, 3(2):115– 130, 2000.

14. C. Gao and S. Michel. Top-k interesting phrase mining in ad-hoc collections using sequence pattern indexing. In Proceedings of the 15th International Conference on Extending Database Technology, EDBT ’12, pages 264–275, New York, NY, USA, 2012. ACM.

15. P. Geurts, D. Ernst, and L. Wehenkel. Extremely randomized trees.Machine learning, 63(1):3–42, 2006.

16. M. A. Halliday et al. Lexis as a linguistic level. In memory of JR Firth, 148:162, 1966.

17. K. S. Hasan and V. Ng. Conundrums in unsupervised keyphrase extraction: making sense of the state-of-the-art. In COLING, 2010.

18. T. Koo, X. Carreras, and M. Collins. Simple semi-supervised dependency parsing. ACL-HLT, 2008.

19. J. Leskovec, L. Backstrom, and J. Kleinberg. Meme-tracking and the dynamics of the news cycle. In Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD ’09, pages 497–506, New York, NY, USA, 2009. ACM.

20. R. Levy and C. Manning. Is it harder to parse Chinese, or the Chinese treebank? In Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1, ACL ’03, pages 439–446, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.

**15. Bibliography**

(1) Java Complete Reference by Herbert Shield

(2) Database Programming with JDBC and Java by George Reese

(3) Java and XML By Brett McLaughlin

(4) Wikipedia, URL: <http://www.wikipedia.org>.

(5) Answers.com, Online Dictionary, Encyclopedia and much more, URL: <http://www.answers.com>

(6) Google, URL: <http://www.google.co.in>

(7)Project Management URL: <http://www.startwright.com/project.html>