**MODELLING INFLUENCE DIFFUSION OVER SIGNED SOCIAL NETWORKS**

*A Mini Project Report submitted to*

*JNTU Hyderabad in partial fulfillment*

*of the requirements for the award of the degree*

**BACHELOR OF TECHNOLOGY**

In

**COMPUTER SCIENCE AND ENGINEERING**

***Submitted by***

**GADI PRATHYUSHA 19RG1A0516**

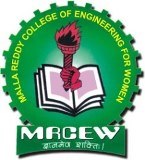
**GOTTAPU LAVANYA 19RG1A0519**

**MANDAVA HESMITHA REDDY 20RG5A0501 GOSIKONDA SHRUTHI 20RG5A0502**

*Under the Guidance of*

**Mrs. N. RADHIKA**

  Asst. Professor, B. Tech, M. Tech



***DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING***

**MALLA REDDY COLLEGE OF ENGINEERING FOR WOMEN**

*(Approved by AICTE New Delhi and Affiliated to JNTUH)*

*An ISO 9001: 2015 Certified Institution*

*(All B. Tech Programs Accredited by NBA)*

*Maisammaguda, Medchal (M), Hyderabad-500100, T. S.*

*JUNE 2022*

***DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING***

**MALLA REDDY COLLEGE OF ENGINEERING FOR WOMEN**

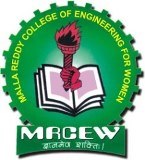
*(Approved by AICTE New Delhi and Affiliated to JNTUH)*

*An ISO 9001: 2015 Certified Institution*

*(All B. Tech Programs Accredited by NBA)*

*Maisammaguda, Medchal (M), Hyderabad-500100, T. S.*

*JUNE 2022*



***CERTIFICATE***

This is to certify that the mini project entitled **“MODELLING INFLUENCE DIFFUSION OVER SIGNED SOCIAL NETWORKS”** has been submitted by **GADI PRATHYUSHA (19RG1A0516), GOTTAPU LAVANYA (19RG1A0519), MANDAVA HESMITHA REDDY (20RG5A0501), GOSIKONDA SHRUTHI (20RG5A0502)** in partial fulfillment of the requirements for the award of **BACHELOR OF TECHNOLOGY** in **COMPUTER SCIENCE & ENGINEERING**. This record of bonafide work carried out by them under my guidance and supervision. ***The result embodied in this mini project report has not been submitted to any other University or Institute for the award of any degree.***

**Mrs. N. Radhika Dr. Kanaka Durga Returi**

*Assistant Professor Head of the Department*

*Project Guide*

***External Examiner***

**ACKNOWLEDGEMENT**

The Mini Project work carried out by our team in the Department of Computer Science and Engineering, Malla Reddy College of Engineering for Women, Hyderabad. ***This work is original and has not been submitted in part or full for any degree or diploma of any other university.***

We wish to acknowledge our sincere thanks to our project guide **Dr. Kanaka Durga Returi**, Professor & Head of the Department, Computer Science & Engineering for formulation of the problem, analysis, guidance and her continuous supervision during the course of work.

We acknowledge our sincere thanks to **Dr. Vaka Murali Mohan,** Principal, MRCEW and all our faculties,CSEDfor their kind cooperation in making this Major Project work a success.

We extend our gratitude to **Sri. Ch. Malla Reddy**, Founder Chairman and **Sri. Ch. Mahender Reddy,** Secretary for their kind cooperation in providing the infrastructure for completion of our Mini Project.

We acknowledge our special thanks to the entire teaching faculty and non-teaching staff members of the Computer Science & Engineering Department for their support in making this project work a success.

**GADI PRATHYUSHA 19RG1A0516**

**GOTTAPU LAVANYA 19RG1A0519**

**MANDAVA HESMITHA REDDY 20RG5A0501**

**GOSIKONDA SHRUTHI 20RG1A0502**

**INDEX**

**Chapter Page No.**

**ABSTRACT** vi

**LIST OF FIGURES** vii

**1. SYSTEM ANALYSIS 8**

1.1 Existing System 8

1.1.1 Disadvantages of Existing System 9

1.2 Proposed System 10

1.2.1 Advantages of Proposed System 11

1.3 Introduction 11

**2. LITERATURE SURVEY 14**

**3. SYSTEM DESIGN 17**

3.1 System Architecture 17

3.1.1 Modules 17

3.1.2 Block Diagram 19

3.2 System Requirements 20

3.2.1 Hardware Requirements 20

3.2.2 Software Requirements 20

3.3 UML Diagrams 21

3.3.1 Use Case Diagram 21

3.3.2 Class Diagram 22

3.3.3 Sequence Diagram 23

3.3.4 Data Flow Diagram 24

3.3.5 Flow Chart Diagram 25

**4. INPUT & OUTPUT DESIGN 27**

4.1 Input Design 27

4.2 Output Design 28

**5. SOFTWARE ENVIRONMENT 30**

5.1 Java Technology 30

5.1.1 Java as a programming language 30

5.1.2 Java as a platform language 32

5.2 JDBC 38

5.2.1 JDBC Goals 38

5.3 Networking 42

5.3.1 TCP/IP 42

5.3.2 IP Datagrams 42

5.3.3 UDP 43

5.3.4 Internet and Network Address 44

5.4 Sockets 45

5.5 J2ME 47

5.5.1 J2ME Architecture 48

**6. SYSTEM STUDY 52**

6.1 Economic Feasibility53

6.2 Technical Feasibility 53

6.3 Social Feasibility 54

**7. SYSTEM TESTING** **55**

7.1 Types of Tests 55

7.1.1 Unit testing 55

7.1.2 Integration testing 55

7.1.3 Functional testing 56

7.1.4 System testing 56

7.1.5 Acceptance testing 57

**8. RESULTS 60**

**9. CONCLUSION & FUTURE ENHANCEMENT 70**

**10**. **BIBLIOGRAPHY**  **72**

**ABSTRACT**

In offline or online worlds, many social systems can be represented as signed social networks including both positive and negative relationships. Although a variety of studies on signed social networks have been conducted motivated by the great application value of unique polarity characteristics, how to model the process of influence propagation over signed social networks is still an important problem that remains pretty much open. Currently, a few studies extended traditional diffusion

models (*e.g.,* Independent Cascade model and Linear Threshold model) from unsigned social networks to signed social networks for estimating positive and negative influence of user sets. However, all of above extension models are stochastic and descriptive models. In order to ensure the accuracy of estimated influence, existing models require a significant number of Monte-Carlo simulations which are very time-consuming and not scalable. Aiming at this issue, we propose the Polarity-related Linear Influence Diffusion (PLID) model which can quickly and accurately calculate polarity-related influence of user sets without simulations. To validate effectiveness and efficiency of our proposed model, we make use of our PLID model to solve the positive influence maximization problem in signed social networks under rigorous mathematical proofs. Extensive experiments demonstrate that our PLID model and approximation algorithm significantly outperform state-of-the-art methods in terms of positive influence spread and running time, using Epinions and Slashdot datasets.

**LIST OF FIGURES**

**Fig.No. Fig. Name Page No**

3.1.1 Block Diagram 2

3.3.1 Use Case Diagram 3

3.3.2 Class Diagram 19

3.3.3 Sequence Diagram 21

3.3.4 Data Flow Diagram 22

3.3.5 Flow Chart Diagram 30

8.1 Admin Login 60

8.2 Admin Menu 61

8.3 All Users List 62

8.4 Add Product 63

8.5 Reviewed Users 64

8.6 Reviewed Posts 65

8.7 All Product Details 66

8.8 View Dislike Results 66

8.9 View Like Results 67

8.10 User Login 67

8.11 Welcome User Menu 68

8.12 Search Products 68

8.13 Search Results 69

8.14 Reviewed Posts 69

8.15 User Search History 70

8.16 KD Tree based Reviewed Users 70

**CHAPTER-1: SYSTEM ANALYSIS**

* 1. **Existing System**

Domingo’s *et al* first introduced the influence maximizationproblem in 2001. Kempe *et al* first define the influencemaximization as a discrete optimization problem.They also proposed a greedy solution, and illustratedthat it performed better than some other naïvemethods. There have been a lot of studies devotedto address the efficiency issue of the method inLeskovec *et al* developed the Cost-Effective LazyForward (CELF) selection strategy by leveraging thesubmodular property of the influence function fully,which runs about seven hundred times faster than theoriginal greedy solution. Kim *et al* developed aindependent path algorithm to approximate influenceestimated by IC model. Similarly, Chen *et al*. presentedPMIA and DegreeDiscountIC heuristicmethods to estimate influence of seed node sets usingnetwork structures. Cheng *et al* presenteda StaticGreedy algorithm to avoid a lot of Monte-Carlo simulations using snapshots. They *et al* also found an iterative ranking method which ownsadvantages of greedy method and heuristic method.Liu *et al* developed the Group-PageRank methodto calculate the upper bound of influence quickly inthe IM problem. Tang *et al* proposed a martingalemethod to solve the influence maximization problemin near-linear time.

The extensive studies on influence maximization problem have been conducted from various aspects. Influence diffusion is a temporal dynamic process, some studies focused on how to identify influential nodes from social networks under the time constraint. Competitive influence maximization of multiple information has been actively studied in the literature. Furthermore, Lu

*et al* extended competition to complementarity, and studied comparative influence diffusion and maximization. Chen *et al*, Aslay *et al* and Chen *et*

*al* explored topic-aware influence maximization problem. Lee *et al* and Li *et al* attempted to maximize influence spread in specific users of social networks. Guo *et al* studied location related influence maximization in trajectory databases. Although influence maximization has attracted tremendous attention, all of above studies were based on unsigned social networks, and ignored the positive and negative polarities of relationships between users.

**Research of signed social networks.** Victor *et al* and Tang *et al* made use of both positive and negative relationships to strength the performance of the recommendation systems. Chen *et al* proposed a user model integrating trust and distrust relations, which is helpful to address the cold start problem in recommendation systems. Yang *et al* and Liu *et* *al* studied how to detect communities from signed social networks. Tang *et al* investigated an interesting problem that is negative relationships prediction only based on positive relationships and interaction content. Song *et al* proposed a generalized AUC (GAUC) to quantify the ranking performance of all kinds of potential links based on partially observed signed social networks. Leskovec *et al*. studied the classical structural balance theory based on online social networks. Wu *et al*. proposed the Troll-Trust model to rank users in signed social networks.

**1.1.1 Disadvantages of Existing System**

1) The system is not implemented by polarity influence calculation under the PLID model.

2) The system doesn’t implement POSITIVE INFLUENCE MAXIMIZATION BASED ON PLID MODEL.

**1.2 Proposed System**

To this end, we provide a new influence diffusion model based on the linear iterative mind for effective and efficient polarity-related influence estimation. Different from the stochastic and descriptive models in, our proposed model is a computational model which directly calculates polarity-related influence of user sets without a large number of simulations. Because the ground truth of influence of user sets is unavailable, it is difficult to evaluate different diffusion models directly. Therefore, we adopt an indirect evaluation way in this paper. Specifically, we combine our model with a greedy solution to address the positive influence maximization (PIM) problem under rigorous mathematical proofs.” *The* *PIM problem is the extension of the traditional influence* *maximization problem in signed social networks”*, and it owns complete and standard evaluation criterions. If our proposed model performs better than other models in terms of solving the PIM problem, this means the efficiency and effectiveness of our model

are validated.

• We propose a novel Polarity-related Linear Influence Diffusion (PLID) model in signed social networks, which is a computational model and has the capability of estimating positive and negative influence of node sets accurately and quickly. The PLID model fuses the social principle that” *the* *friend of my friend is my friend, and the friend of my* *enemy is my enemy, and the enemy of my friend is my* *enemy, and the enemy of my enemy is my friend*”.

• We make use of our PLID model to solve the PIM problem. Specifically, we mathematically prove that the objective function of the PIM problem maintains monotonicity and sub modularity under the PLID model, which enables a greedy

seed nodes selection strategy to provide 1 − 1/e approximation for the PIM problem.

• We conduct a comprehensive set of experiments on Opinions and Slashdot datasets to evaluate our method. The experimental results indicate that our method achieves better positive influence spread performance in solving the PIM problem, and runs 7-35 times faster, than the state-of-the art approach.

**1.2.1 Advantages of Proposed System**

* The system is more effective due to presence of POLARITY RELATED LINEAR INFLUENCE DIFFUSION MODEL.
* The system is more secure due to presence of Modeling Signed Social Networks.

**1.3 Introduction**

*”The social system is a complex system of reciprocal actions and social relationships involving a plurality of individuals”*. In recent years, a variety of onlinenetworking sites spring up and provide new platformsof friends making, information propagation andcrowd-sourcing for users around the world, whichcan be regarded as online social systems. In some online social systems (*e.g., Epinions*, *Slashdot*), users are allowed to mark positive signs or negative signs on the relationships with other users. In some other systems (*e.g., Twitter*, *Weibo*), polarity signs are not available directly, however, can be detected from the interactive data between users. The abov e described systems including positive and negative relationships are usually modeled as signed social networks, which can describe the real social systems more accurately than unsigned social networks ignoring the polarities of the relationships.

The polarity properties of relationships between users are helpful to understand network evolution, analyze user preferences, predict user behaviors. Besides that, the unprecedented availability of online social data provides the opportunity for the signed social network research. Therefore, so far, a series of classic research problems (*e.g.,* community detection, link prediction, users ranking and recommendation system) have been conducted on signed social networks. However, modeling the process of influence propagation over signed social networks is still an important problem that remains pretty much open. Diffusion models can be used to estimate influence of user sets which plays a crucial role in solving some real application problems, such as viral marketing, rumor control. The positive and negative polarities of relationships existing in signed social networks cause that the non-polarity influence of a user set will be divided into positive influence and negative influence. How to distinguish these two types of influence and estimate them exactly and quickly is a critical challenge in modeling influence diffusion over signed social networks.

Currently, a few studies attempted to solve the above challenge by extending some classic models (*e.g.* Independent Cascade model, Voter model and Linear Threshold model) from unsigned social networks to signed social networks. However, all of above extended models are stochastic and descriptive models. To estimate positive and negative influence of user sets, these models must run a certain number of Monte-Carlo simulations. When the number of simulations is small, the estimated influence is not accurate. When the number of simulations (10000- 20000) is large enough to obtain accurate influence, the simulation process consumes a great deal of time, which leads that these models are not applicable to large scale social networks. Therefore, existing diffusion models over signed social networks are notable to take into account of accuracy and efficiency in terms of polarity-related influence estimation.

To this end, we provide a new influence diffusion model based on the linear iterative mind for effective and efficient polarity-related influence estimation. Different from the stochastic and descriptive models in, our proposed model is a computational model which directly calculates polarity-related influence of user sets without a large number of simulations. Because the ground truth of influence of user sets is unavailable, it is difficult to evaluate different diffusion models directly. Therefore, we adopt an indirect evaluation way in this paper. Specifically,we combine our model with greedy solution to address the positive influence maximization (PIM) problem under rigorous mathematical proofs.” *The* *PIM problem is the extension of the traditional influence* *maximization problem in signed social networks”*, and it owns complete and standard evaluation criterions. If our proposed model performs better than other models in terms of solving the PIM problem, this means the efficiency and effectiveness of our model are validated.

In summary, this paper makes the following contributions:

• We propose a novel Polarity-related Linear Influence Diffusion (PLID) model in signed social networks, which is a computational model and has the capability of estimating positive and negative influence of node sets accurately and quickly. ThePLID model fuses the social principle that” the *friend of my friend is my friend, and the friend of my enemy is my enemy, and the enemy of my friend is my enemy, and the enemy of my enemy is my friend*”.

• We make use of our PLID model to solve the PIMproblem. Specifically, we mathematically provethat the objective function of the PIM problem maintains monotonicity and sub modularity under the PLID model, which enables a greedy seed nodes selection strategy to provide 1 − 1/e approximation for the PIM problem.

• We conduct a comprehensive set of experiments on Epinions and Slashdot datasets to evaluate our method. The experimental results indicate that our method achieves better positive influence spread performance in solving the PIM problem, and runs 7-35 times faster, than the state-of-the art approach.

**CHAPTER-2: LITERATURE SURVEY**

Domingo’s *et al* first introduced the influence maximizationproblem in 2001. Kempe *et al* first define the influencemaximization as a discrete optimization problem.They also proposed a greedy solution, and illustratedthat it performed better than some other naïvemethods.

Leskovec *et al* developed the Cost-Effective LazyForward (CELF) selection strategy by leveraging thesubmodular property of the influence function fully,which runs about seven hundred times faster than theoriginal greedy solution. Kim *et al* developed aindependent path algorithm to approximate influenceestimated by IC model.

Chen *et al*. presentedPMIA and DegreeDiscountIC heuristicmethods to estimate influence of seed node sets usingnetwork structures. Cheng *et al* presenteda StaticGreedy algorithm to avoid a lot of Monte-Carlo simulations using snapshots. They *et al* also found an iterative ranking method which ownsadvantages of greedy method and heuristic method.

Liu *et al* developed the Group-PageRank methodto calculate the upper bound of influence quickly inthe IM problem. Tang *et al* proposed a martingalemethod to solve the influence maximization problemin near-linear time.

The extensive studies on influence maximization problem have been conducted from various aspects. Influence diffusion is a temporal dynamic process, some studies focused on how to identify influential nodes from social networks under the time constraint. Competitive influence maximization of multiple information has been actively studied in the literature.

Lu *et al* extended competition to complementarity, and studied comparative influence diffusion and maximization. Chen *et al*, Aslay *et al* and Chen *et*

*al* explored topic-aware influence maximization problem.

Lee *et al* and Li *et al* attempted to maximize influence spread in specific users of social networks. Guo *et al* studied location related influence maximization in trajectory databases. Although influence maximization has attracted tremendous attention, all of above studies were based on unsigned social networks, and ignored the positive and negative polarities of relationships between users.

Victor *et al* and Tang *et al* made use of both positive and negative relationships to strength the performance of the recommendation systems. Chen *et al* proposed a user model integrating trust and distrust relations, which is helpful to address the cold start problem in recommendation systems. Yang *et al* and Liu *et* *al* studied how to detect communities from signed social networks. Tang *et al* investigated an interesting problem that is negative relationships prediction only based on positive relationships and interaction content. Song *et al* proposed a generalized AUC (GAUC) to quantify the ranking performance of all kinds of potential links based on partially observed signed social networks. Leskovec *et al*. studied the classical structural balance theory based on online social networks. Wu *et al*. proposed the Troll-Trust model to rank users in signed social networks.

**CHAPTER-3: SYSTEM DESIGN**

**3.1 System Architecture**

**3.1.1 Modules:**

**Admin**

In this module, the admin has to login by using valid user name and password. After login successful he can perform some operations such as Login, View All Users, Add Products, View All Product Posts, View All Reviewed Posts, View All Users Search History, View All Search Ratio, View All Reviewed Users, View Like Results, View Dislike Results.

**Friend Request & Response**

In this module, the admin can view all the friend requests and responses. Here all the requests and responses will be displayed with their tags such as Id, requested user photo, requested user name, user name request to, status and time & date. If the user accepts the request then the status will be changed to accepted or else the status will remains as waiting.

**Social Network Friends**

In this module, the admin can see all the friends who are all belongs to the same site. The details such as, Request From, Requested user’s site, Request To Name, Request To user’s site.

**All Recommended Posts**

In this module, the admin can see all the posts which are shared among the friends in same and other network sites. The details such as post image, title, description, recommend by name and recommend to name.

* **User**

In this module, there are n numbers of users are present. User should register before performing any operations. Once user registers, their details will be stored to the database. After registration successful, he has to login by using authorized user name and password. Once Login is successful user can perform some operations like Register and Login, Search Product, Search Reviewed Products, My Search History, My Reviewed Posts, Reviews with KD Tree.

**Searching Users**

In this module, the user searches for users in Same Site and in Different Sites and sends friend requests to them. The user can search for users in other sites to make friends only if they have permission.

**Adding Posts**

In this module, the user adds posts details such as title, description and the image of the post. The post details such as title and description will be encrypted and stores into the database.

**3.1.1 Block Diagram:**

**3.1.1 Block Diagram**

Login,

View All Users,

Add Products,

View All Product Posts,

View All Reviewed Posts,

View All Users Search History,

View All Search Ratio,

View All Reviewed Users,

View Like Results,

View DisLike Resultss.

**Store and retrievals**

**User**

Register and Login,

Search Product,

Search Reviewed Products,

My Search History,

My Reviewed Posts,

Reviews with KD Tree.

**WEB Database**

* 1. **System Requirements**
     1. **Hardware Requirements**

Processor - Pentium –IV

RAM - 4 GB (min)

Hard Disk - 20 GB

Key Board - Standard Windows Keyboard

Mouse - Two or Three Button Mouse

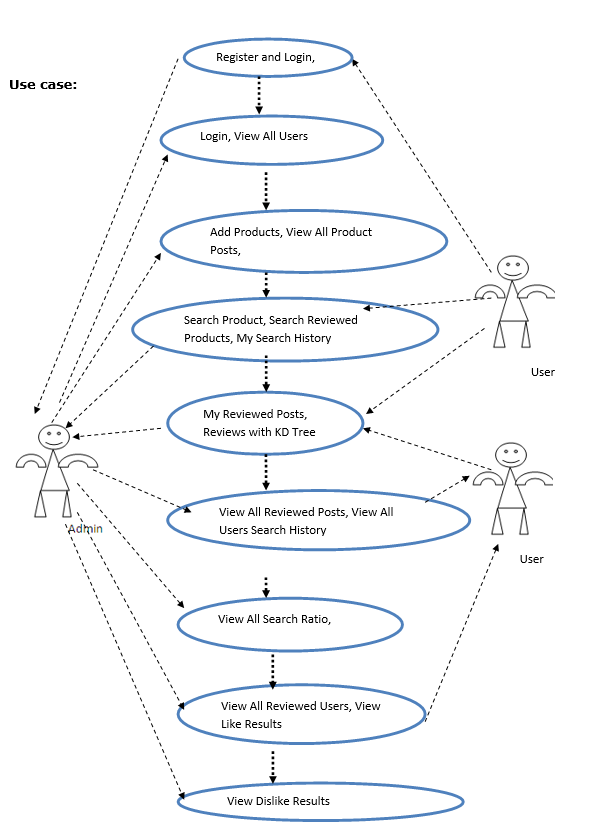
Monitor - SVGA

**3.3.2 Software Requirements**

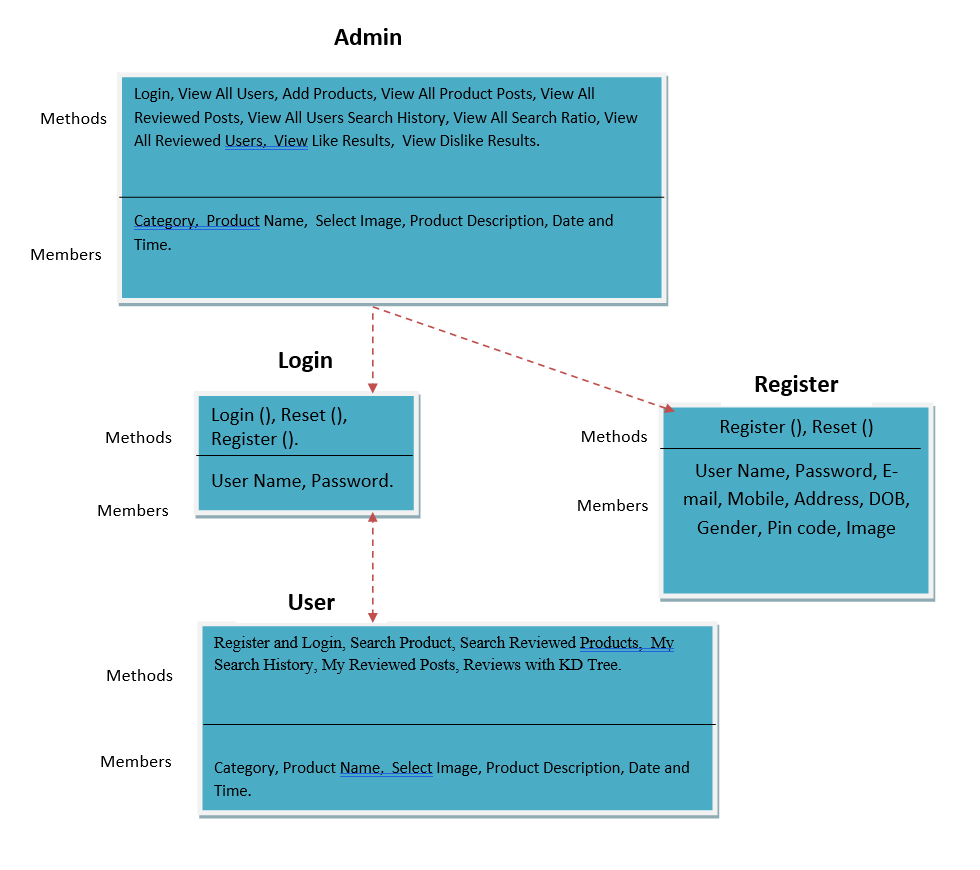
* Operating System - Windows XP
* Coding Language - Java/J2EE (JSP, Servlet)
* Front End - J2EE
* Back End - MySQL

**3.3 UML Diagrams**

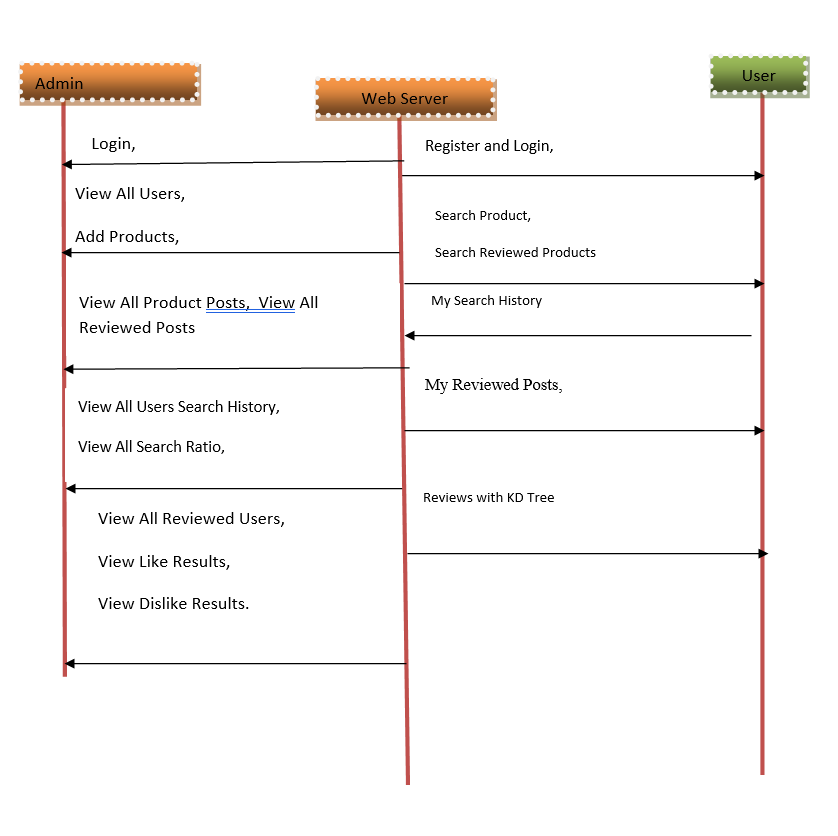
**3.3.1 Use Case Diagram**



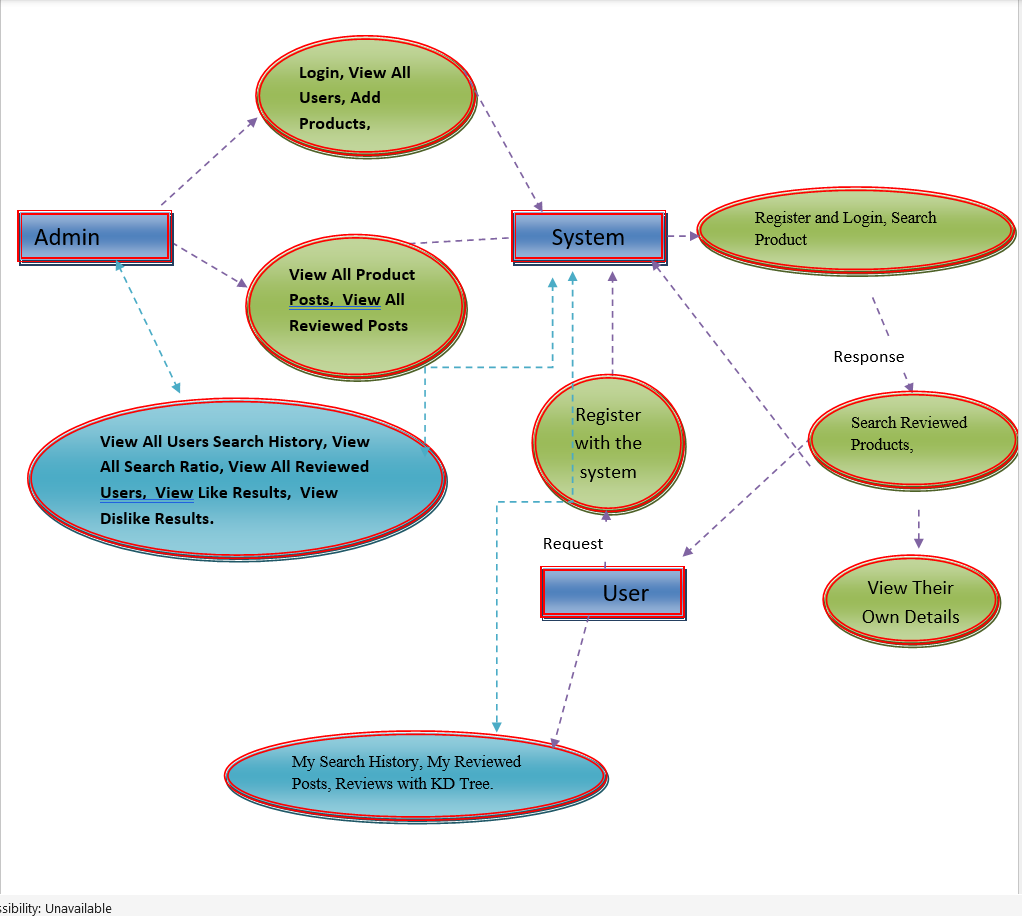
* + 1. **Class Diagram**



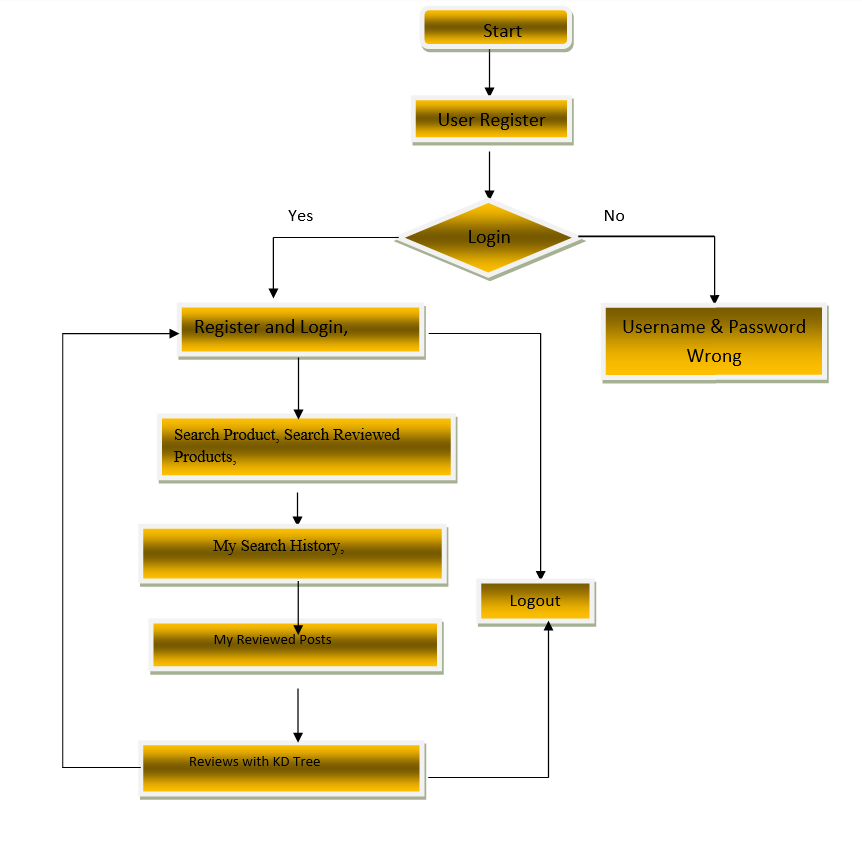
* + 1. **Sequence Diagram**

****

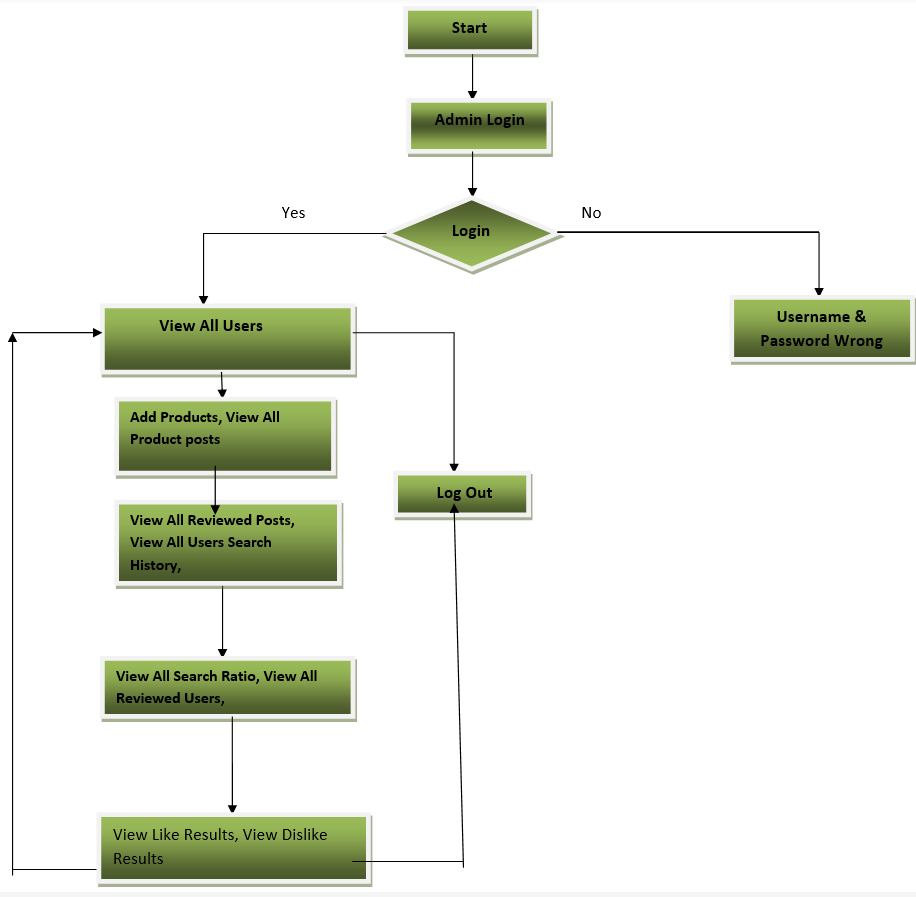
* + 1. **Data Flow Diagram**



**3.3.5 Flow Chart Diagram**



**ADMIN LOGIN**



**CHAPTER-4: INPUT AND OUTPUT DESIGN**

**4.1 Input Design**

Input Design plays a vital role in the life cycle of software development, it requires very careful attention of developers. The input design is to feed data to the application as accurate as possible. So inputs are supposed to be designed effectively so that the errors occurring while feeding are minimized. According to Software Engineering Concepts, the input forms or screens are designed to provide to have a validation control over the input limit, range and other related validations.

This system has input screens in almost all the modules. Error messages are developed to alert the user whenever he commits some mistakes and guides him in the right way so that invalid entries are not made. Let us see deeply about this under module design.

Input design is the process of converting the user created input into a computer-based format. The goal of the input design is to make the data entry logical and free from errors. The error is in the input are controlled by the input design. The application has been developed in user-friendly manner. The forms have been designed in such a way during the processing the cursor is placed in the position where must be entered. The user is also provided with in an option to select an appropriate input from various alternatives related to the field in certain cases.

Validations are required for each data entered. Whenever a user enters an erroneous data, error message is displayed and the user can move on to the subsequent pages after completing all the entries in the current page.

**4.2 Output Design**

The Output from the computer is required to mainly create an efficient method of communication within the company primarily among the project leader and his team members, in other words, the administrator and the clients. The output of VPN is the system which allows the project leader to manage his clients in terms of creating new clients and assigning new projects to them, maintaining a record of the project validity and providing folder level access to each client on the user side depending on the projects allotted to him. After completion of a project, a new project may be assigned to the client. User authentication procedures are maintained at the initial stages itself. A new user may be created by the administrator himself or a user can himself register as a new user but the task of assigning projects and validating a new user rests with the administrator only.

The application starts running when it is executed for the first time. The server has to be started and then the internet explorer in used as the browser. The project will run on the local area network so the server machine will serve as the administrator while the other connected systems can act as the clients. The developed system is highly user friendly and can be easily understood by anyone using it even for the first time.

**CHAPTER-5: SYSTEM ENVIRONMENT**

## **5.1Java Technology**

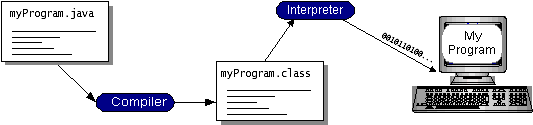
Java technology is both a programming language and a platform.

### 5.1.2The Java Programming Language

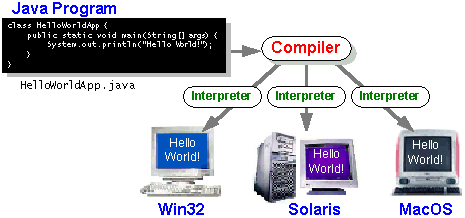
### The Java programming language is a high-level language that can be characterized by all of the following buzzwords:

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

With most programming languages, you either compile or interpret a program so that you can run it on your computer. The Java programming language is unusual in that a program is both compiled and interpreted. With the compiler, first you translate a program into an intermediate language called Java byte codes —the platform-independent codes interpreted by the interpreter on the Java platform. The interpreter parses and runs each Java byte code instruction on the computer. Compilation happens just once; interpretation occurs each time the program is executed. The following figure illustrates how this works.



You can think of Java byte codes as the machine code instructions for the Java Virtual Machine (Java VM). Every Java interpreter, whether it’s a development tool or a Web browser that can run applets, is an implementation of the Java VM. Java byte codes help make “write once, run anywhere” possible. You can compile your program into byte codes on any platform that has a Java compiler. The byte codes can then be run on any implementation of the Java VM. That means that as long as a computer has a Java VM, the same program written in the Java programming language can run on Windows 2000, a Solaris workstation, or on an iMac.



### 5.1.2The Java Platform

A platform is the hardware or software environment in which a program runs. We’ve already mentioned some of the most popular platforms like Windows 2000, Linux, Solaris, and MacOS. Most platforms can be described as a combination of the operating system and hardware. The Java platform differs from most other platforms in that it’s a software-only platform that runs on top of other hardware-based platforms.

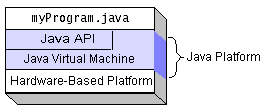
The Java platform has two components:

* The Java Virtual Machine (Java VM)
* The Java Application Programming Interface (Java API)

You’ve already been introduced to the Java VM. It’s the base for the Java platform and is ported onto various hardware-based platforms.

The Java API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets. The Java API is grouped into libraries of related classes and interfaces; these libraries are known as packages. The next section, What Can Java Technology Do? Highlights what functionality some of the packages in the Java API provide.

The following figure depicts a program that’s running on the Java platform. As the figure shows, the Java API and the virtual machine insulate the program from the hardware.



Native code is code that after you compile it, the compiled code runs on a specific hardware platform. As a platform-independent environment, the Java platform can be a bit slower than native code. However, smart compilers, well-tuned interpreters, and just-in-time byte code compilers can bring performance close to that of native code without threatening portability.

## What Can Java Technology Do?

The most common types of programs written in the Java programming language are applets and applications. If you’ve surfed the Web, you’re probably already familiar with applets. An applet is a program that adheres to certain conventions that allow it to run within a Java-enabled browser.

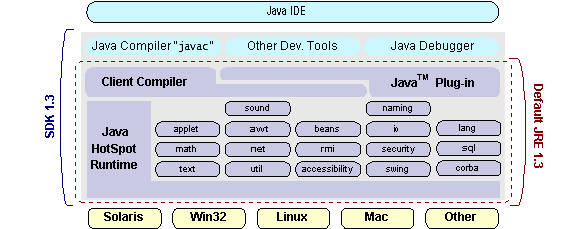
However, the Java programming language is not just for writing cute, entertaining applets for the Web. The general-purpose, high-level Java programming language is also a powerful software platform. Using the generous API, you can write many types of programs.

An application is a standalone program that runs directly on the Java platform. A special kind of application known as a server serves and supports clients on a network. Examples of servers are Web servers, proxy servers, mail servers, and print servers. Another specialized program is a servlet. A servlet can almost be thought of as an applet that runs on the server side. Java Servlets are a popular choice for building interactive web applications, replacing the use of CGI scripts. Servlets are similar to applets in that they are runtime extensions of applications. Instead of working in browsers, though, servlets run within Java Web servers, configuring or tailoring the server.

How does the API support all these kinds of programs? It does so with packages of software components that provides a wide range of functionality. Every full implementation of the Java platform gives you the following features:

* **The essentials**: Objects, strings, threads, numbers, input and output, data structures, system properties, date and time, and so on.
* **Applets**: The set of conventions used by applets.
* **Networking**: URLs, TCP (Transmission Control Protocol), UDP (User Data gram Protocol) sockets, and IP (Internet Protocol) addresses.
* **Internationalization**: Help for writing programs that can be localized for users worldwide. Programs can automatically adapt to specific locales and be displayed in the appropriate language.
* **Security**: Both low level and high level, including electronic signatures, public and private key management, access control, and certificates.
* **Software components**: Known as JavaBeansTM, can plug into existing component architectures.
* **Object serialization**: Allows lightweight persistence and communication via Remote Method Invocation (RMI).
* **Java Database Connectivity (JDBCTM)**: Provides uniform access to a wide range of relational databases.

The Java platform also has APIs for 2D and 3D graphics, accessibility, servers, collaboration, telephony, speech, animation, and more. The following figure depicts what is included in the Java 2 SDK.



## How Will Java Technology Change My Life?

We can’t promise you fame, fortune, or even a job if you learn the Java programming language. Still, it is likely to make your programs better and requires less effort than other languages. We believe that Java technology will help you do the following:

* **Get started quickly**: Although the Java programming language is a powerful object-oriented language, it’s easy to learn, especially for programmers already familiar with C or C++.
* **Write less code**: Comparisons of program metrics (class counts, method counts, and so on) suggest that a program written in the Java programming language can be four times smaller than the same program in C++.
* **Write better code**: The Java programming language encourages good coding practices, and its garbage collection helps you avoid memory leaks. Its object orientation, its JavaBeans component architecture, and its wide-ranging, easily extendible API let you reuse other people’s tested code and introduce fewer bugs.
* **Develop programs more quickly**: Your development time may be as much as twice as fast versus writing the same program in C++. Why? You write fewer lines of code and it is a simpler programming language than C++.
* **Avoid platform dependencies with 100% Pure Java**: You can keep your program portable by avoiding the use of libraries written in other languages. The 100% Pure JavaTM Product Certification Program has a repository of historical process manuals, white papers, brochures, and similar materials online.
* **Write once, run anywhere**: Because 100% Pure Java programs are compiled into machine-independent byte codes, they run consistently on any Java platform.
* **Distribute software more easily**: You can upgrade applets easily from a central server. Applets take advantage of the feature of allowing new classes to be loaded “on the fly,” without recompiling the entire program.

### ODBC

Microsoft Open Database Connectivity (ODBC) is a standard programming interface for application developers and database systems providers. Before ODBC became a *de facto* standard for Windows programs to interface with database systems, programmers had to use proprietary languages for each database they wanted to connect to. Now, ODBC has made the choice of the database system almost irrelevant from a coding perspective, which is as it should be. Application developers have much more important things to worry about than the syntax that is needed to port their program from one database to another when business needs suddenly change.

Through the ODBC Administrator in Control Panel, you can specify the particular database that is associated with a data source that an ODBC application program is written to use. Think of an ODBC data source as a door with a name on it. Each door will lead you to a particular database. For example, the data source named Sales Figures might be a SQL Server database, whereas the Accounts Payable data source could refer to an Access database. The physical database referred to by a data source can reside anywhere on the LAN.

The ODBC system files are not installed on your system by Windows 95. Rather, they are installed when you setup a separate database application, such as SQL Server Client or Visual Basic 4.0. When the ODBC icon is installed in Control Panel, it uses a file called ODBCINST.DLL. It is also possible to administer your ODBC data sources through a stand-alone program called ODBCADM.EXE. There is a 16-bit and a 32-bit version of this program and each maintains a separate list of ODBC data sources.

From a programming perspective, the beauty of ODBC is that the application can be written to use the same set of function calls to interface with any data source, regardless of the database vendor. The source code of the application doesn’t change whether it talks to Oracle or SQL Server. We only mention these two as an example. There are ODBC drivers available for several dozen popular database systems. Even Excel spreadsheets and plain text files can be turned into data sources. The operating system uses the Registry information written by ODBC Administrator to determine which low-level ODBC drivers are needed to talk to the data source (such as the interface to Oracle or SQL Server). The loading of the ODBC drivers is transparent to the ODBC application program. In a client/server environment, the ODBC API even handles many of the network issues for the application programmer.

The advantages of this scheme are so numerous that you are probably thinking there must be some catch. The only disadvantage of ODBC is that it isn’t as efficient as talking directly to the native database interface. ODBC has had many detractors make the charge that it is too slow. Microsoft has always claimed that the critical factor in performance is the quality of the driver software that is used. In our humble opinion, this is true. The availability of good ODBC drivers has improved a great deal recently. And anyway, the criticism about performance is somewhat analogous to those who said that compilers would never match the speed of pure assembly language. Maybe not, but the compiler (or ODBC) gives you the opportunity to write cleaner programs, which means you finish sooner. Meanwhile, computers get faster every year.

**5.2 JDBC**

In an effort to set an independent database standard API for Java; Sun Microsystems developed Java Database Connectivity, or JDBC. JDBC offers a generic SQL database access mechanism that provides a consistent interface to a variety of RDBMSs. This consistent interface is achieved through the use of “plug-in” database connectivity modules, or *drivers*. If a database vendor wishes to have JDBC support, he or she must provide the driver for each platform that the database and Java run on.

To gain a wider acceptance of JDBC, Sun based JDBC’s framework on ODBC. As you discovered earlier in this chapter, ODBC has widespread support on a variety of platforms. Basing JDBC on ODBC will allow vendors to bring JDBC drivers to market much faster than developing a completely new connectivity solution.

JDBC was announced in March of 1996. It was released for a 90 day public review that ended June 8, 1996. Because of user input, the final JDBC v1.0 specification was released soon after.

The remainder of this section will cover enough information about JDBC for you to know what it is about and how to use it effectively. This is by no means a complete overview of JDBC. That would fill an entire book.

### 5.2.1 JDBC Goals

Few software packages are designed without goals in mind. JDBC is one that, because of its many goals, drove the development of the API. These goals, in conjunction with early reviewer feedback, have finalized the JDBC class library into a solid framework for building database applications in Java.

The goals that were set for JDBC are important. They will give you some insight as to why certain classes and functionalities behave the way they do. The eight design goals for JDBC are as follows:

1. ***SQL Level API***

The designers felt that their main goal was to define a SQL interface for Java. Although not the lowest database interface level possible, it is at a low enough level for higher-level tools and APIs to be created. Conversely, it is at a high enough level for application programmers to use it confidently. Attaining this goal allows for future tool vendors to “generate” JDBC code and to hide many of JDBC’s complexities from the end user.

1. ***SQL Conformance***

SQL syntax varies as you move from database vendor to database vendor. In an effort to support a wide variety of vendors, JDBC will allow any query statement to be passed through it to the underlying database driver. This allows the connectivity module to handle non-standard functionality in a manner that is suitable for its users.

1. ***JDBC must be implemental on top of common database interfaces***The JDBC SQL API must “sit” on top of other common SQL level APIs. This goal allows JDBC to use existing ODBC level drivers by the use of a software interface. This interface would translate JDBC calls to ODBC and vice versa.
2. ***Provide a Java interface that is consistent with the rest of the Java system***

Because of Java’s acceptance in the user community thus far, the designers feel that they should not stray from the current design of the core Java system.

1. ***Keep it simple***

This goal probably appears in all software design goal listings. JDBC is no exception. Sun felt that the design of JDBC should be very simple, allowing for only one method of completing a task per mechanism. Allowing duplicate functionality only serves to confuse the users of the API.

1. ***Use strong, static typing wherever possible***

Strong typing allows for more error checking to be done at compile time; also, less error appear at runtime.

1. ***Keep the common cases simple***

Because more often than not, the usual SQL calls used by the programmer are simple SELECT’s, INSERT’s, DELETE’s and UPDATE’s, these queries should be simple to perform with JDBC. However, more complex SQL statements should also be possible.

Finally, we decided to proceed the implementation using Java

**5.3 Networking.**

And for dynamically updating the cache table we go for MS Access database.

Java ha two things: a programming language and a platform.

Java is a high-level programming language that is all of the following

Simple Architecture-neutral

Object-oriented Portable

Distributed High-performance

Interpreted multithreaded

Robust Dynamic

Secure

Java is also unusual in that each Java program is both compiled and interpreted. With a compile you translate a Java program into an intermediate language called Java byte codes the platform-independent code instruction is passed and run on the computer.

Compilation happens just once; interpretation occurs each time the program is executed. The figure illustrates how this works.

**Java Program**

**Compilers**

**Interpreter**

**My Program**

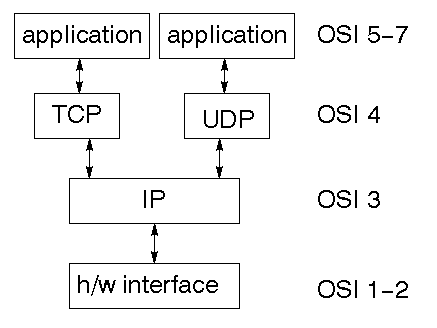
You can think of Java byte codes as the machine code instructions for the Java Virtual Machine (Java VM). Every Java interpreter, whether it’s a Java development tool or a Web browser that can run Java applets, is an implementation of the Java VM. The Java VM can also be implemented in hardware.

Java byte codes help make “write once, run anywhere” possible. You can compile your Java program into byte codes on my platform that has a Java compiler. The byte codes can then be run any implementation of the Java VM. For example, the same Java program can run Windows NT, Solaris, and Macintosh.

## Networking

### 5.3.1 TCP/IP stack

The TCP/IP stack is shorter than the OSI one:



TCP is a connection-oriented protocol; UDP (User Datagram Protocol) is a connectionless protocol.

### 5.3.2 IP datagram’s

The IP layer provides a connectionless and unreliable delivery system. It considers each datagram independently of the others. Any association between datagram must be supplied by the higher layers. The IP layer supplies a checksum that includes its own header. The header includes the source and destination addresses. The IP layer handles routing through an Internet. It is also responsible for breaking up large datagram into smaller ones for transmission and reassembling them at the other end.

### 5.3.3 UDP

UDP is also connectionless and unreliable. What it adds to IP is a checksum for the contents of the datagram and port numbers. These are used to give a client/server model - see later.

### TCP

TCP supplies logic to give a reliable connection-oriented protocol above IP. It provides a virtual circuit that two processes can use to communicate.

### 5.3.4 Internet addresses

In order to use a service, you must be able to find it. The Internet uses an address scheme for machines so that they can be located. The address is a 32-bit integer which gives the IP address. This encodes a network ID and more addressing. The network ID falls into various classes according to the size of the network address.

### Network address

Class A uses 8 bits for the network address with 24 bits left over for other addressing. Class B uses 16-bit network addressing. Class C uses 24-bit network addressing and class D uses all 32.

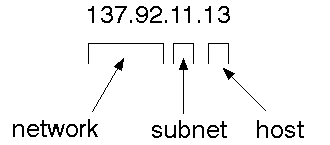
### Subnet address

Internally, the UNIX network is divided into sub networks. Building 11 is currently on one sub network and uses 10-bit addressing, allowing 1024 different hosts.

### Host address

8 bits are finally used for host addresses within our subnet. This places a limit of 256 machines that can be on the subnet.

### Total address



The 32-bit address is usually written as 4 integers separated by dots.

### Port addresses

A service exists on a host, and is identified by its port. This is a 16-bit number. To send a message to a server, you send it to the port for that service of the host that it is running on. This is not location transparency! Certain of these ports are "well known".

### 5.4 Sockets

A socket is a data structure maintained by the system to handle network connections. A socket is created using the call socket. It returns an integer that is like a file descriptor. In fact, under Windows, this handle can be used with Read File and Write File functions.

#include <sys/types.h>

#include <sys/socket.h>

int socket(int family, int type, int protocol);

Here "family" will be AF\_INET for IP communications, protocol will be zero, and type will depend on whether TCP or UDP is used. Two processes wishing to communicate over a network create a socket each. These are similar to two ends of a pipe - but the actual pipe does not yet exist.

**JFree Chart**

JFreeChart is a free 100% Java chart library that makes it easy for developers to display professional quality charts in their applications. JFreeChart's extensive feature set includes:

A consistent and well-documented API, supporting a wide range of chart types;

A flexible design that is easy to extend, and targets both server-side and client-side applications;

Support for many output types, including Swing components, image files (including PNG and JPEG), and vector graphics file formats (including PDF, EPS and SVG);

JFreeChart is "open source" or, more specifically, [free software](http://www.gnu.org/philosophy/free-sw.html). It is distributed under the terms of the [GNU Lesser General Public License](http://www.gnu.org/licenses/lgpl.html) (LGPL), which permits use in proprietary applications.

## 1. Map Visualizations

Charts showing values that relate to geographical areas. Some examples include: (a) population density in each state of the United States, (b) income per capita for each country in Europe, (c) life expectancy in each country of the world. The tasks in this project include:

Sourcing freely redistributable vector outlines for the countries of the world, states/provinces in particular countries (USA in particular, but also other areas);

Creating an appropriate dataset interface (plus default implementation), a rendered, and integrating this with the existing XYPlot class in JFreeChart;

Testing, documenting, testing some more, documenting some more.

## 2. Time Series Chart Interactivity

Implement a new (to JFreeChart) feature for interactive time series charts --- to display a separate control that shows a small version of ALL the time series data, with a sliding "view" rectangle that allows you to select the subset of the time series data to display in the main chart.

## 3. Dashboards

There is currently a lot of interest in dashboard displays. Create a flexible dashboard mechanism that supports a subset of JFreeChart chart types (dials, pies, thermometers, bars, and lines/time series) that can be delivered easily via both Java Web Start and an applet.

## 4. Property Editors

The property editor mechanism in JFreeChart only handles a small subset of the properties that can be set for charts. Extend (or reimplement) this mechanism to provide greater end-user control over the appearance of the charts.

**5.5 J2ME (Java 2 Micro edition):-**

Sun Microsystems defines J2ME as "a highly optimized Java run-time environment targeting a wide range of consumer products, including pagers, cellular phones, screen-phones, digital set-top boxes and car navigation systems." Announced in June 1999 at the JavaOne Developer Conference, J2ME brings the cross-platform functionality of the Java language to smaller devices, allowing mobile wireless devices to share applications. With J2ME, Sun has adapted the Java platform for consumer products that incorporate or are based on small computing devices.

**5.5.1 J2ME architecture**

**1. General J2ME architecture**



J2ME uses configurations and profiles to customize the Java Runtime Environment (JRE). As a complete JRE, J2ME is comprised of a configuration, which determines the JVM used, and a profile, which defines the application by adding domain-specific classes. The configuration defines the basic run-time environment as a set of core classes and a specific JVM that run on specific types of devices. We'll discuss configurations in detail in the profile defines the application; specifically, it adds domain-specific classes to the J2ME configuration to define certain uses for devices. We'll cover profiles in depth in the following graphic depicts the relationship between the different virtual machines, configurations, and profiles. It also draws a parallel with the J2SE API and its Java virtual machine. While the J2SE virtual machine is generally referred to as a JVM, the J2ME virtual machines, KVM and CVM, are subsets of JVM. Both KVM and CVM can be thought of as a kind of Java virtual machine -- it's just that they are shrunken versions of the J2SE JVM and are specific to J2ME.

**2.Developing J2ME applications**

Introduction In this section, we will go over some considerations you need to keep in mind when developing applications for smaller devices. We'll take a look at the way the compiler is invoked when using J2SE to compile J2ME applications. Finally, we'll explore packaging and deployment and the role preverification plays in this process.

**3.Design considerations for small devices**

Developing applications for small devices requires you to keep certain strategies in mind during the design phase. It is best to strategically design an application for a small device before you begin coding. Correcting the code because you failed to consider all of the "gotchas" before developing the application can be a painful process. Here are some design strategies to consider:

\* Keep it simple. Remove unnecessary features, possibly making those features a separate, secondary application.

\* Smaller is better. This consideration should be a "no brainer" for all developers. Smaller applications use less memory on the device and require shorter installation times. Consider packaging your Java applications as compressed Java Archive (jar) files.

\* Minimize run-time memory use. To minimize the amount of memory used at run time, use scalar types in place of object types. Also, do not depend on the garbage collector. You should manage the memory efficiently yourself by setting object references to null when you are finished with them. Another way to reduce run-time memory is to use lazy instantiation, only allocating objects on an as-needed basis. Other ways of reducing overall and peak memory use on small devices are to release resources quickly, reuse objects, and avoid exceptions.

**4.Configurations overview**

The configuration defines the basic run-time environment as a set of core classes and a specific JVM that run on specific types of devices. Currently, two configurations exist for J2ME, though others may be defined in the future:

\* **Connected Limited Device Configuration (CLDC)** is used specifically with the KVM for 16-bit or 32-bit devices with limited amounts of memory. This is the configuration (and the virtual machine) used for developing small J2ME applications. Its size limitations make CLDC more interesting and challenging (from a development point of view) than CDC. CLDC is also the configuration that we will use for developing our drawing tool application. An example of a small wireless device running small applications is a Palm hand-held computer.

\* **Connected Device Configuration (CDC)** is used with the C virtual machine (CVM) and is used for 32-bit architectures requiring more than 2 MB of memory. An example of such a device is a Net TV box.

**5.J2ME profiles**

**What is a J2ME profile?**

As we mentioned earlier in this tutorial, a profile defines the type of device supported. The Mobile Information Device Profile (MIDP), for example, defines classes for cellular phones. It adds domain-specific classes to the J2ME configuration to define uses for similar devices. Two profiles have been defined for J2ME and are built upon CLDC: KJava and MIDP. Both KJava and MIDP are associated with CLDC and smaller devices. Profiles are built on top of configurations. Because profiles are specific to the size of the device (amount of memory) on which an application runs, certain profiles are associated with certain configurations.

A skeleton profile upon which you can create your own profile, the Foundation Profile, is available for CDC.

**Profile 1: KJava**

KJava is Sun's proprietary profile and contains the KJava API. The KJava profile is built on top of the CLDC configuration. The KJava virtual machine, KVM, accepts the same byte codes and class file format as the classic J2SE virtual machine. KJava contains a Sun-specific API that runs on the Palm OS. The KJava API has a great deal in common with the J2SE Abstract Windowing Toolkit (AWT). However, because it is not a standard J2ME package, its main package is com.sun.kjava. We'll learn more about the KJava API later in this tutorial when we develop some sample applications.

**Profile 2: MIDP**

MIDP is geared toward mobile devices such as cellular phones and pagers. The MIDP, like KJava, is built upon CLDC and provides a standard run-time environment that allows new applications and services to be deployed dynamically on end user devices. MIDP is a common, industry-standard profile for mobile devices that is not dependent on a specific vendor. It is a complete and supported foundation for mobile application

development. MIDP contains the following packages, the first three of which are core CLDC packages, plus three MIDP-specific packages.

\* java.lang

\* java.io

\* java.util

\* javax.microedition.io

\* javax.microedition.lcdui

\* javax.microedition.midlet

\* javax.microedition.rms

**CHAPTER-6: SYSTEM STUDY**

**FEASIBILITY STUDY**

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

* ECONOMICAL FEASIBILITY
* TECHNICAL FEASIBILITY
* SOCIAL FEASIBILITY

**6.1 ECONOMICAL FEASIBILITY**

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

### 6.2 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

**6.3 SOCIAL FEASIBILITY**

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

**CHAPTER-7: SYSTEM TESTING**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

**7.1 TYPES OF TESTS**

**7.1.1 Unit testing**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

**7.1.2 Integration testing**

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

**7.1.3 Functional test**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

**7.1.4 System Testing**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

**White Box Testing**

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

**Black Box Testing**

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

**7.1.5 Acceptance Testing**

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

**OTHER TESTING METHODOLOGIES**

**User Acceptance Testing**

User Acceptance of a system is the key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with the prospective system users at the time of developing and making changes wherever required. The system developed provides a friendly user interface that can easily be understood even by a person who is new to the system.

**Output Testing**

After performing the validation testing, the next step is output testing of the proposed system, since no system could be useful if it does not produce the required output in the specified format. Asking the users about the format required by them tests the outputs generated or displayed by the system under consideration. Hence the output format is considered in 2 ways – one is on screen and another in printed format.

**Validation Checking**

Validation checks are performed on the following fields.

**Text Field:**

The text field can contain only the number of characters lesser than or equal to its size. The text fields are alphanumeric in some tables and alphabetic in other tables. Incorrect entry always flashes and error message.

**Numeric Field:**

The numeric field can contain only numbers from 0 to 9. An entry of any character flashes an error messages. The individual modules are checked for accuracy and what it has to perform. Each module is subjected to test run along with sample data. The individually tested modules are integrated into a single system. Testing involves executing the real data information is used in the program the existence of any program defect is inferred from the output. The testing should be planned so that all the requirements are individually tested.

A successful test is one that gives out the defects for the inappropriate data and produces and output revealing the errors in the system.

**Preparation of Test Data**

Taking various kinds of test data does the above testing. Preparation of test data plays a vital role in the system testing. After preparing the test data the system under study is tested using that test data. While testing the system by using test data errors are again uncovered and corrected by using above testing steps and corrections are also noted for future use.

**Using Live Test Data:**

Live test data are those that are actually extracted from organization files. After a system is partially constructed, programmers or analysts often ask users to key in a set of data from their normal activities. Then, the systems person uses this data as a way to partially test the system. In other instances, programmers or analysts extract a set of live data from the files and have them entered themselves.

It is difficult to obtain live data in sufficient amounts to conduct extensive testing. And, although it is realistic data that will show how the system will perform for the typical processing requirement, assuming that the live data entered are in fact typical, such data generally will not test all combinations or formats that can enter the system. This bias toward typical values then does not provide a true systems test and in fact ignores the cases most likely to cause system failure.

**Using Artificial Test Data:**

Artificial test data are created solely for test purposes, since they can be generated to test all combinations of formats and values. In other words, the artificial data, which can quickly be prepared by a data generating utility program in the information systems department, make possible the testing of all login and control paths through the program.

The most effective test programs use artificial test data generated by persons other than those who wrote the programs. Often, an independent team of testers formulates a testing plan, using the systems specifications.

The package “Virtual Private Network” has satisfied all the requirements specified as per software requirement specification and was accepted.

**USER TRAINING**

Whenever a new system is developed, user training is required to educate them about the working of the system so that it can be put to efficient use by those for whom the system has been primarily designed. For this purpose the normal working of the project was demonstrated to the prospective users. Its working is easily understandable and since the expected users are people who have good knowledge of computers, the use of this system is very easy.

**MAINTAINENCE**

This covers a wide range of activities including correcting code and design errors. To reduce the need for maintenance in the long run, we have more accurately defined the user’s requirements during the process of system development. Depending on the requirements, this system has been developed to satisfy the needs to the largest possible extent. With development in technology, it may be possible to add many more features based on the requirements in future. The coding and designing is simple and easy to understand which will make maintenance easier.

**TESTING STRATEGY :**

A strategy for system testing integrates system test cases and design techniques into a well planned series of steps that results in the successful construction of software. The testing strategy must co-operate test planning, test case design, test execution, and the resultant data collection and evaluation .A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high level tests that validate major system functions against user requirements.

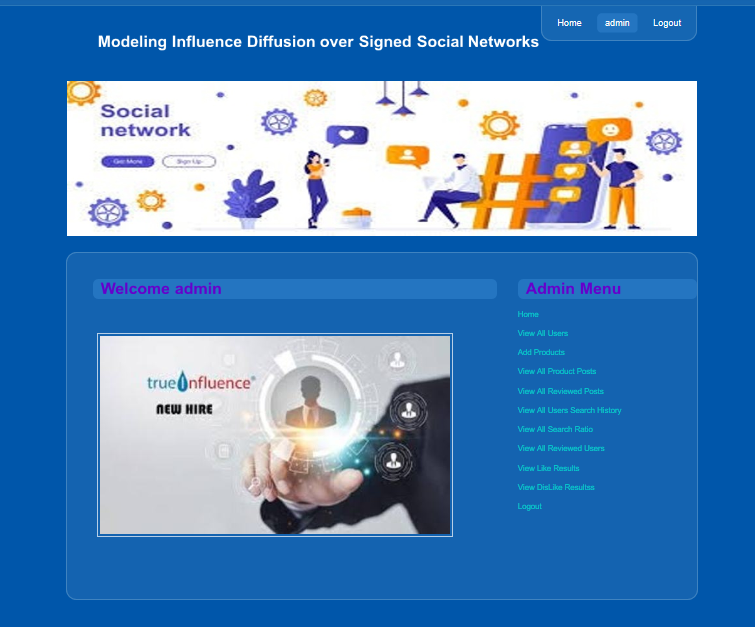
Software testing is a critical element of software quality assurance and represents the ultimate review of specification design and coding. Testing represents an interesting anomaly for the software. Thus, a series of testing are performed for the proposed system before the system is ready for user acceptance testing.

**CHAPTER-8: RESULTS**

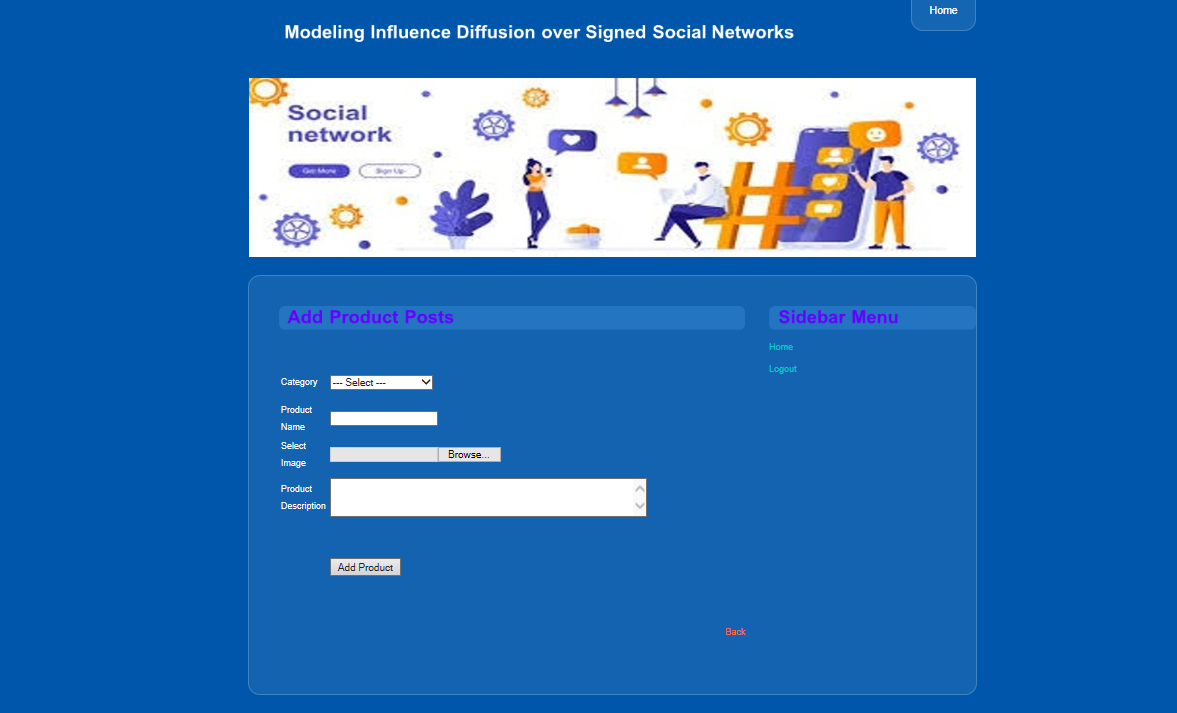
**8.1 Admin Login**



**8.2 Admin Menu**



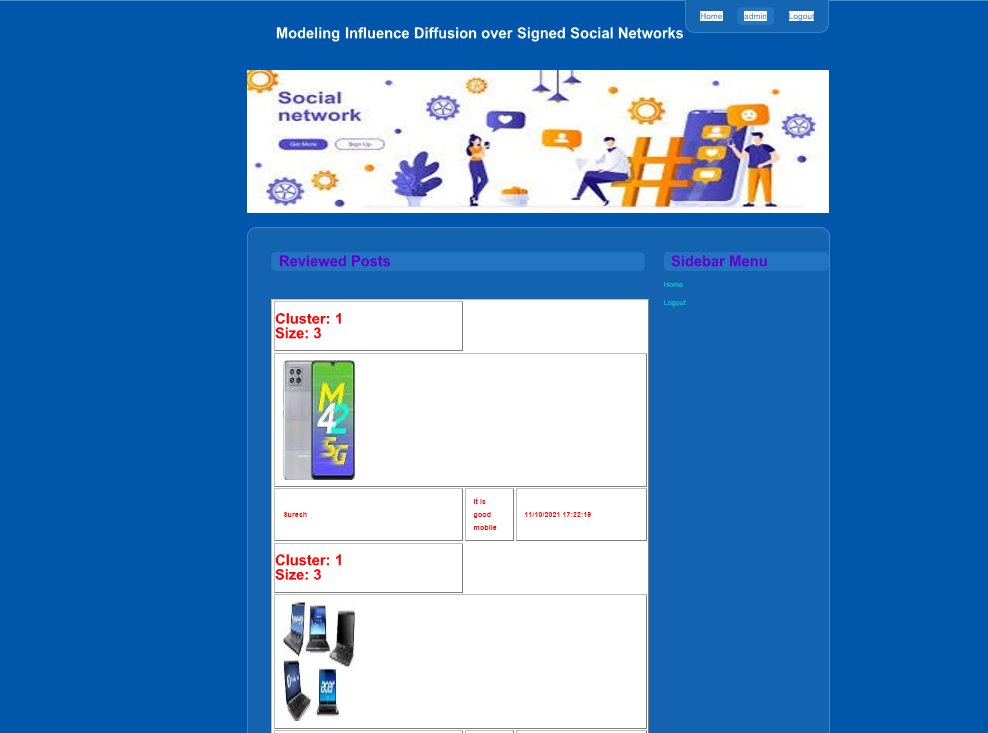
**8.3 All Users List**

**8.4 Add Product**

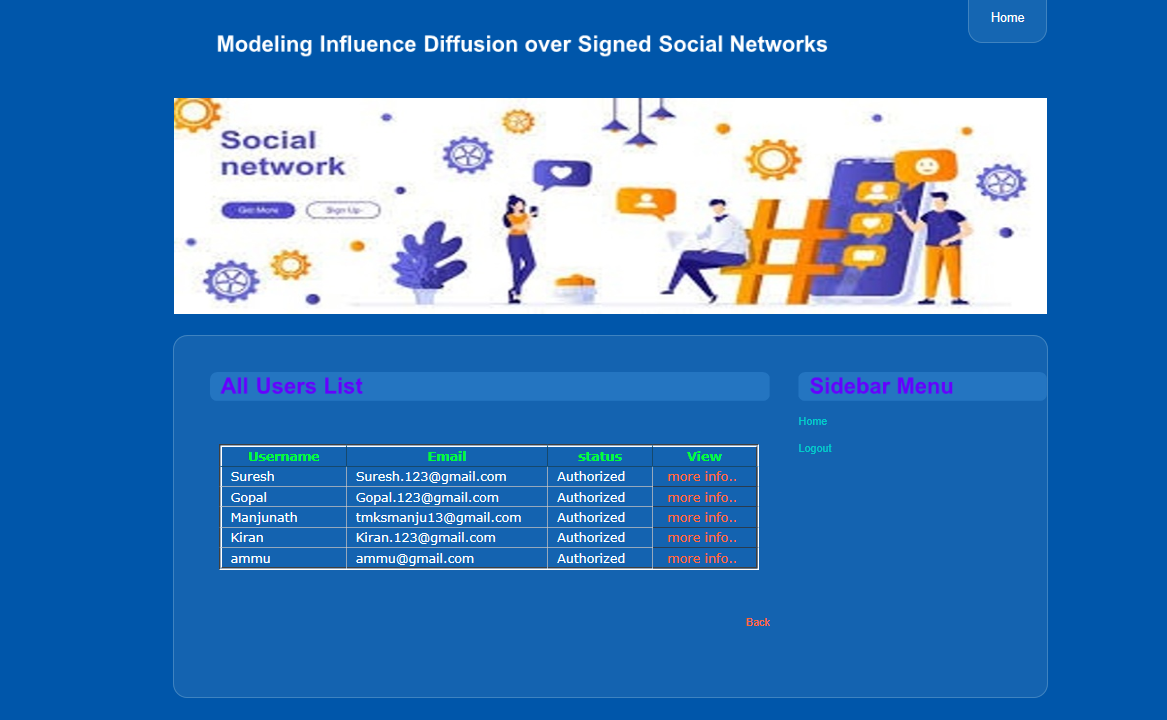
**8.5 Reviewed Users**



**8.6 Reviwed Posts**



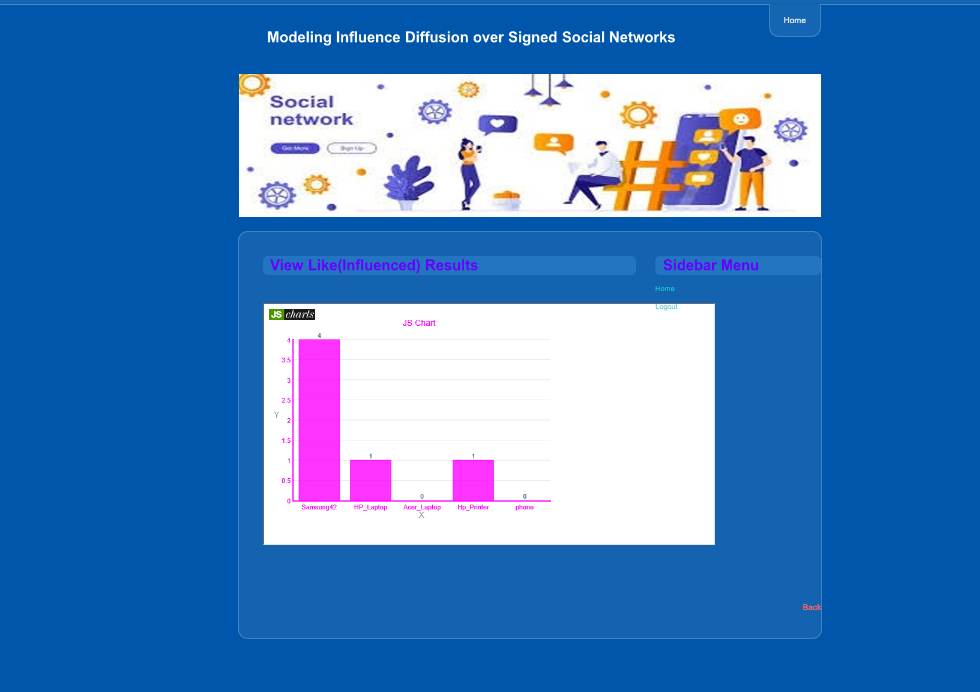
**8.7 All Product Details**



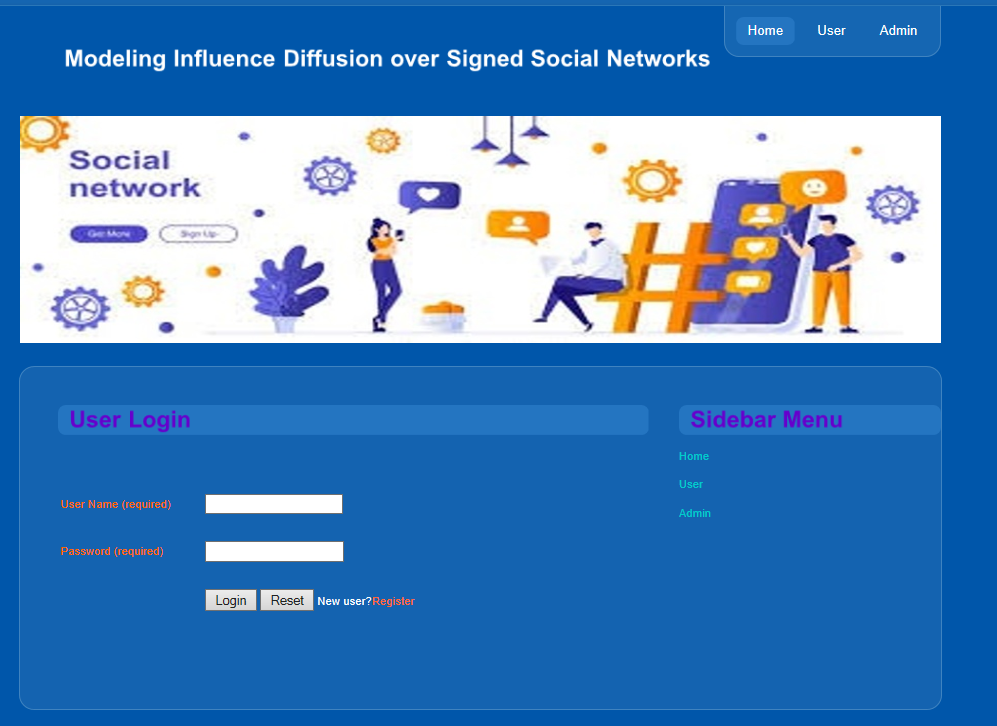
**8.8 View Dislike Results**



**8.9 View Like Results**



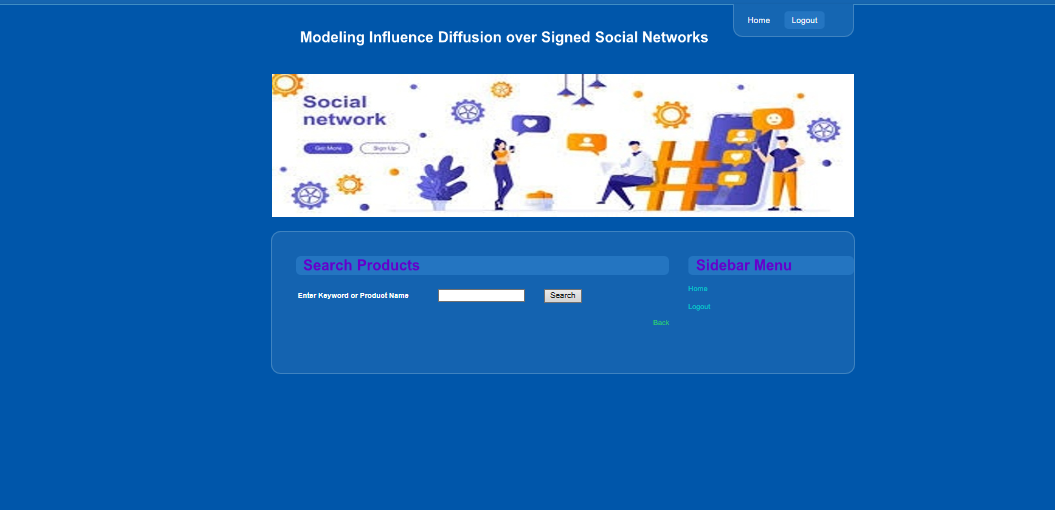
**8.10 User Login**



**8.11 Welcome User Menu**



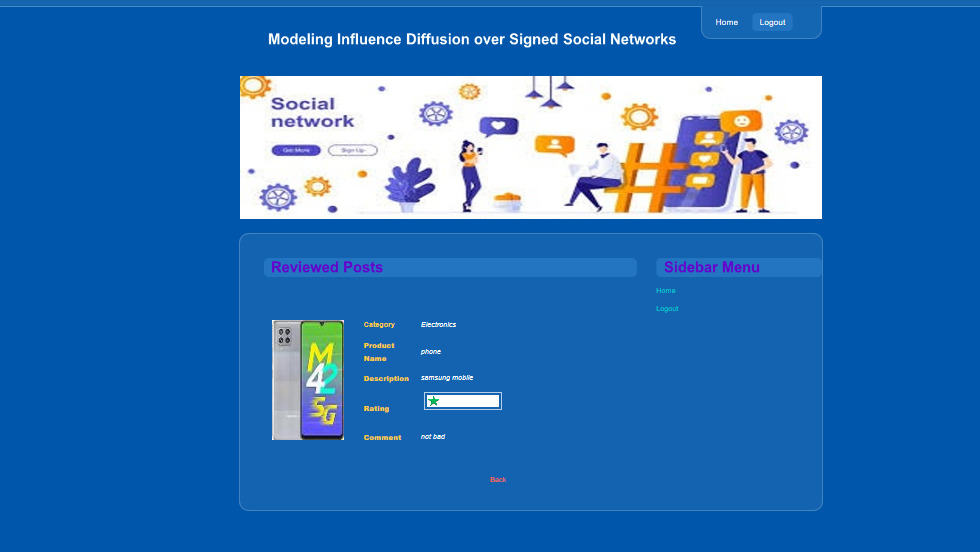
**8.12 Search Products**



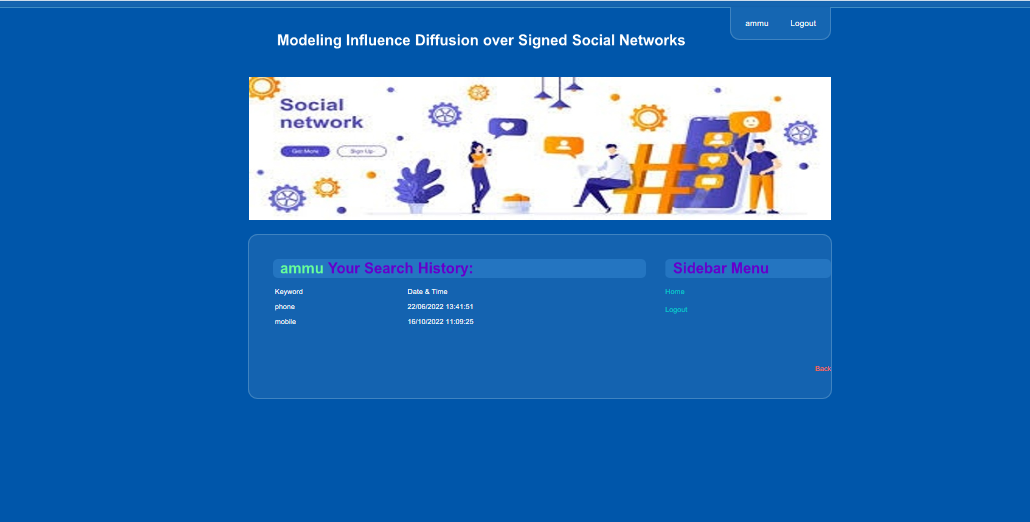
**8.13 Search Results**



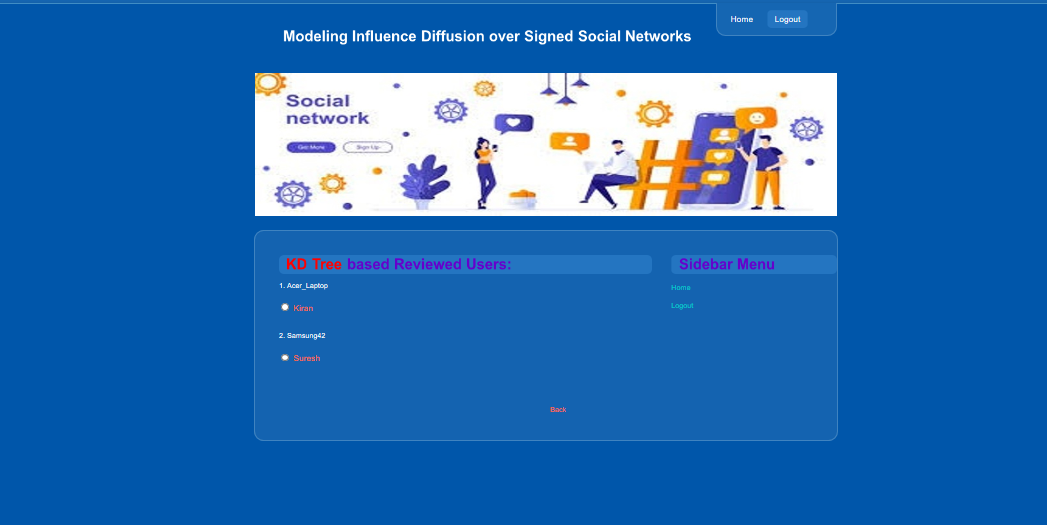
**8.14 Reviwed Posts**



**8.15 User Search History**



**8.16 KD Tree based Reviewed Users**



**CHAPTER-9: CONCLUSION & FUTURE ENHANCEMENT**

**Conclusion**

In this paper, we first proposed the Polarity-related Linear Influence Diffusion (PLID) model to estimate polarity related influence of user sets in online social systems. Nextly, we leverage our PLID model to address the PIM problem. Specifically, we proved that the objective function of the PIM problem maintains monotonicity and sub modularity, and presented a greedy solution to select seed nodes. Finally, comprehensive experiments on Epinions and Slashdot datasets demonstrate that our method achieves better performance than state-of-the-art approaches in terms of positive influence propagation and running time.

The polarity-related influence of users is also time related. Therefore, how to introduce time factor into our model for calculating temporal polarity-related influence will be an interesting problem. Besides that, although our PLID model is much faster than IC-P model, the greedy algorithm is still computationally expensive. Developing more efficient seed node selection strategies for the PIM problem will be one of our future research directions. Finally, online social systems provide a great quantity of diverse data for the social computing research , how to exploit the big data to reveal the implicit social phenomenon’s and serve the society is worth to explore.

**Future Enhancement**

* **PLID** (Polarity Related Linear Influence Diffusion Model) is quickly & accurately calculates polarity related influence of user sets without simulations.
* Present Project is developed in java. Later on python can be used to reduce the complexity of code.
* Directly users are allowed to mark positive signs or negative signs on the relationships with other users.

**CHAPTER 10: BIBLIOGRAPHY**

1. J. Bensman and I. Gerver, “Crime and punishment in the

factory: The function of deviancy in maintaining the social

system,” *American Sociological Review*, pp. 588–598, 1963.

2. B. Yang, X. Zhao, and X. Liu, “Bayesian approach to modeling

and detecting communities in signed network.” in *AAAI*, 2015,

pp. 1952–1958.

3. C. Liu, J. Liu, and Z. Jiang, “A multiobjective evolutionary

algorithm based on similarity for community detection

from signed social networks,” *IEEE transactions on cybernetics*,

vol. 44, no. 12, pp. 2274–2287, 2014.

4. J. Tang, S. Chang, C. Aggarwal, and H. Liu, “Negative link

prediction in social media,” in *Proceedings of the Eighth ACM*

*International Conference on Web Search and Data Mining*. ACM,

2015, pp. 87–96.

5. Z. Wu, C. C. Aggarwal, and J. Sun, “The troll-trust model for

ranking in signed networks,” in *Proceedings of the Ninth ACM*

*International Conference on Web Search and Data Mining*. ACM,

2016, pp. 447–456.

6. P. Victor, C. Cornelis, and et al., “Trust-and distrust-based

recommendations for controversial reviews,” in *Web Science*

*Conference (WebSci’09: Society On-Line)*, no. 161, 2009.

7. J. Tang, C. Aggarwal, and H. Liu, “Recommendations in

signed social networks,” in *Proceedings of the 25th International*

*Conference on World Wide Web*. International World Wide Web

Conferences Steering Committee, 2016, pp. 31–40.

8. H. Wang, Q. Yang, and et al., “Maximizing positive influence

in signed social networks,” in *International Conference on Cloud*

*Computing and Security*. Springer, 2015, pp. 356–367.

9. Y. Li, W. Chen, Y. Wang, and Z.-L. Zhang, “Influence diffusion

dynamics and influence maximization in social networks with

friend and foe relationships,” in *Proceedings of the sixth ACM*

*international conference on Web search and data mining*. ACM,

2013, pp. 657–666.

10. D. Li, Z.-M. Xu, N. Chakraborty, and et al., “Polarity related

influence maximization in signed social networks,” *PloS one*,

vol. 9, no. 7, p. e102199, 2014.

11. P. Domingos and M. Richardson, “Mining the network value

of customers,” in *Proceedings of the seventh ACM SIGKDD*

*international conference on Knowledge discovery and data mining*.

ACM, 2001, pp. 57–66.

12. D. Kempe, J. Kleinberg, and ´E. Tardos, “Maximizing the

spread of influence through a social network,” in *Proceedings*

*of the ninth ACM SIGKDD international conference on Knowledge*

*discovery and data mining*. ACM, 2003, pp. 137–146.

13. J. Leskovec, A. Krause, and et al., “Cost-effective outbreak

detection in networks,” in *Proceedings of the 13th ACM SIGKDD*

*international conference on Knowledge discovery and data mining*.

ACM, 2007, pp. 420–429.

14. J. Kim, S.-K. Kim, and H. Yu, “Scalable and parallelizable

processing of influence maximization for large-scale social

networks?” in *Data Engineering (ICDE), 2013 IEEE 29th International*

*Conference on*. IEEE, 2013, pp. 266–277 W. Chen, C. Wang, and Y.

for prevalent viral marketing in large-scale social

networks,” in *Proceedings of the 16th ACM SIGKDD international*

*conference on Knowledge discovery and data mining*. ACM, 2010,

pp. 1029–1038.

16. W. Chen, Y. Wang, and S. Yang, “Efficient influence maximization

in social networks,” in *Proceedings of the 15th ACM*

*SIGKDD international conference on Knowledge discovery and data*

*mining*. ACM, 2009, pp. 199–208.

17. S. Cheng, H. Shen, and et al., “Staticgreedy: solving the

scalability-accuracy dilemma in influence maximization,” in

*Proceedings of the 22nd ACM international conference on Information*

*& Knowledge Management*. ACM, 2013, pp. 509–518.

18. S. Cheng, H. Shen, J. Huang, W. Chen, and X. Cheng, “Imrank:

influence maximization via finding self-consistent ranking,” in

*Proceedings of the 37th international ACM SIGIR conference on*

*Research & development in information retrieval*. ACM, 2014,

pp. 475–484.

19. Q. Liu, B. Xiang, E. Chen, H. Xiong, and et al., “Influence

maximization over large-scale social networks: A bounded

linear approach,” in *Proceedings of the 23rd ACM International*

*Conference on Information and Knowledge Management*. ACM,

2014, pp. 171–180.

20. Y. Tang, Y. Shi, and X. Xiao, “Influence maximization in near linear

time: A martingale approach,” in *Proceedings of the 2015*