

PROFESSIONAL TRAINING REPORT
At
Sathyabama Institute of Science and Technology
(Deemed to be University)

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Technology Degree in

Information Technology

By

SUWETHA.S
(REG.NO.37120901)



DEPARTMENT OF INFORMATION TECHNOLOGY
SCHOOL OF COMPUTING
SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600119. TAMILNADU.

AUGUST2019



SATHYABAMA
Institute of science and technology
(DEEMED TO BE UNIVERSITY)
(Established under Section 3 of UGC Act,
1956) Jeppiaar Nagar, Rajiv Gandhi Salai,
Chennai 600119
www.sathyabamauniversity.ac.in



SCHOOL OF COMPUTING

BONAFIDE CERTIFICATE

This is to certify that this Professional Training Report is the bonafide work of **SUWETHA.S (REG.NO.37120901)** who underwent the professional training in **“NEAREST SHOPPING LOCATION”** under our supervision from June 2019 to august2019.

INTERNAL GUIDE

Dr.P. JEYANTHI.M.E., Ph.D.,

HEAD OF THE DEPARTMENT

Dr. R. SUBHASHINI, M.E., Ph.D.

Submitted for Viva voce Examination heldon_____

InternalExaminer

ExternalExaminer

DECLARATION

ISUWETHA.S(Reg.No.37120901) hereby declare that the Professional Training Report on **“NEAREST SHOPPING LOCATION ”** done by me under the guidance of **DR.JEYANTHI,M.E., Ph.D.**, at Sathyabama Institute of Science and Technology is submitted in partial fulfillment of the requirements for the award of Bachelor of Technology degree in Information Technology

DATE:

PLACE:

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management** Of **SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T.Sasikala M.E., Ph.D, Dean**, School of Computing , **Dr.SUBHASHINI M.E., Ph.D.** , Head of the Department of Information and Technology for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. Jeberson Retna Raj M.E., Ph.D.**, for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Information Technology** who were helpful in many ways for the completion of the project.

TRAINING CERTIFICATE



YOHO TECHNOLOGIES PVT LTD
No.29, 2nd Floor, Ambal Nagar Main Road
Keelakattalai, Chennai - 600117.
www.yohotechnologies.com
+91 7639978189



INTERNSHIP CERTIFICATION

To whom it may concern,

This is to certify that the student SUWETHA.S worked in YOHO TECHNOLOGIES PVT LTD with excellent performance during her internship in WEB DEVELOPMENT from 12th May to 22nd May 2020.

During the internship period, we found that she could bear hardship, having strong sense of responsibility and emphasizing team work, and she gets along well with colleagues and generally obtained high form them.

We wish her good luck in all his future endeavors.

Regards



Mithun v
HR Manager

CHAPTER	TITLE	PAGE NO.
	ABSTRACT	i
	LIST OF ABBREVIATIONS	ii
1.	INTRODUCTION	14
2.	SYSTEM ANALYSIS	15
	2.1 EXISTING SYSTEM	
	2.2 PROPOSED SYSEM	
	2.3 FEASIBILITY ANALYSIS	
3.	SOFTWARE REQUIREMENT SPECIFICATION	19
4.	SOFTWARE DESCRIPTION	20
5.	SYSTEM DESIGN	36
	5.1 SEQUENCE DIAGRAM	
	5.2 CLASS DIAGRAM	
	5.3 COMPONENT DIAGRAM	
	5.4 USECASE DIAGRAM	
	5.5 ACTIVITY DIAGRAM	
	5.6 ER DIAGRAM	
6.	IMPLEMENTATION	42
	MODULES DESCRIPTION	
	6.1 AGGREGATE NEAREST LOCATION	
	6.2 APPROXIMATE NEAREST LOCATION	
	6.3 MINMAX NEAREST LOCATION	
	6.4 NEAREST LOCATION	
7.	SYSTEM TESTING	45
8.	INPUT OUTPUT DESIGN	49
9.	SCREEN SHOTS	51
10.	CONCLUSION	61
11.	BIBILIOGRAPHY	62

a set of points P and a query set Q , a *group enclosing query* (GEQ) fetches the point $p \in P$ such that the maximum distance of p to all points in Q is minimized. This problem is equivalent to the Min-Max case (minimizing the maximum distance) of aggregate nearest neighbor queries for spatial databases. This work first designs a new exact solution by exploring new geometric insights, such as the minimum enclosing ball, the convex hull and the furthest voronoi diagram of the query group. To further reduce the query cost, especially when the dimensionality increases, we turn to approximation algorithms. Our main approximation algorithm has a worst case p_2 -approximation ratio if one can find the exact nearest neighbor of a point. In practice, its approximation ratio never exceeds 1.05 for a large number of data sets up to six dimensions. We also discuss how to extend it to higher dimensions (up to 74 in our experiment) and show that it still maintains a very good approximation quality (still close to 1) and low query cost. In fixed dimensions, we extend the p_2 -approximation algorithm to get a $(1 + \epsilon)$ -approximate solution for the GEQ problem. Both approximation algorithms have $O(\log N + M)$ query cost in any fixed dimension, where N and M are the sizes of the data set P and query group Q . Extensive experiments on both synthetic and real data sets, up to 10 million points and 74 dimensions, confirm the efficiency, effectiveness and scalability of the proposed algorithms, especially their significant improvement over the state-of-the-art method.

LIST OF FIGURES

FIGURE NO.	FIGURENAME	PAGENO.
5.1	Sequence diagram	36
5.3	Class diagram	37
5.4	Component diagram	38
5.5	Usecase diagram	39
5.6	Entity relationship diagram	41

LIST OF TABLES

TABLE NO.	TABLERNAME	PAGE NO
6.1.1	Insertion page	24
6.1.2	search page	24
6.1.3	delete page	25

LIST OF ABBREVIATION

ABBREVIATION

EXPANSION

GEQ	Group Enclosing Query
MBM	Minimum Bonding Method
SQL	STURCTURE QUERYLANGUAGE
JVM	Java Virtual Machine

CHAPTER 1

INTRODUCTION

Aggregate nearest neighbor queries represent a class of important query types that are defined by taking the minimum over the combination of some aggregate operators, such as the SUM, AVG, MAX, and the fundamental nearest neighbor search. These queries are defined over two data sets, the data set P and a query data set Q . Such queries could be generally interpreted as a minimizing optimization problem defined on the nearest neighbor search. For example, when the aggregate operator is the SUM, the aggregate nearest neighbor query is equivalent to the group nearest neighbor query where the goal is to find a point from P that has the minimum SUM distance to all query points in Q . Papadis et al. have given a complete, thorough treatment to a number of aggregate nearest neighbor queries in spatial databases. While being general enough to cover different aggregate operators, its generality also means that important opportunities could be overlooked to optimize query algorithms for specific operators. For instance, the state of the art is limited by heuristics that may yield very high query cost in certain cases, especially for data sets and queries in higher (more than two) dimensions. Motivated by this observation, this work focuses on one specific aggregate operator, namely the MAX, for the aggregate nearest neighbor queries in large databases and designs methods that significantly reduce the query cost compared to the MBM (Minimum Bounding Method) algorithm from, Following the previous instance when studying a specific aggregate type for aggregate nearest neighbor queries (e.g., group nearest neighbor queries for the SUM operator, we designate a query name, the group enclosing query (GEQ), for an aggregate nearest neighbor query with the MAX operator.

CHAPTER 2

2. SYSTEM ANALYSIS

2.1.1 EXISTING SYSTEM

While being general enough to cover different aggregate operators, its generality also means that important opportunities could be overlooked to optimize query algorithms for specific operators. For instance, the state of the art is limited by heuristics that may yield very high query cost in certain cases, especially for data sets and queries in higher (more than two) dimensions. Motivated by this observation, this work focuses on one specific aggregate operator, namely the MAX, for the aggregate nearest neighbor queries in large databases and designs methods that significantly reduce the query cost compared to the MBM (Minimum Bounding Method) algorithm from.

Following the previous instance when studying a specific aggregate type for aggregate nearest neighbor queries (e.g., group nearest neighbor queries for the SUM operator, we designate a query name, the group enclosing query (GEQ), for an aggregate nearest neighbor query with the MAX operator. An example of the GEQ problem is illustrated.

More applications could be listed to demonstrate the usefulness of this query and more examples are available from the prior study. Intuitively, in contrast to many existing variants of the nearest neighbor queries that ask for the best answer of the average case the GEQ problem searches for the best answer of the worst case. This problem becomes even more difficult if the query group is large as well.

2.1.2 PROPOSED SYSTEM

This work presents new, efficient algorithms, including both exact and approximate versions, for the GEQ problem that significantly outperform the state of the art, the MBM algorithm. Specifically, • We present a new exact search method for the GEQ problem in Section 4 that instantiates several new geometric insights, such as the minimum enclosing ball, the convex hull and the furthest voronoi diagram of the query group, to achieve higher pruning power than the MBM approach.

We design a $\sqrt{2}$ -approximation (worst case approximation ratio in *any dimensions*) algorithm in Section 5.1, if one can find the exact nearest neighbor of a point and the minimum enclosing ball of Q . Its asymptotic query cost is $O(\log N + M)$ in *any fixed dimensions*. Our idea is to reduce the GEQ problem to the classical nearest neighbor search by utilizing the center of the minimum enclosing ball for Q .

- We extend the above idea to a $(1+\phi)$ -approximation algorithm in any fixed dimension in Section 5.2. This algorithm has a strong theoretical interest and it also achieves the optimal $O(\log N+M)$ query cost in any fixed dimension.
- We extend the same idea from the $\sqrt{2}$ -approximate algorithm to much higher dimensions in Section 5.3, since it is impossible to find the exact nearest neighbor *efficiently* and the exact minimum enclosing ball in high dimensions in practice. We leverage on the latest, practical approximate nearest neighbor search method (the LSB-tree [31]) and the $(1 + \phi)$ - approximate minimum enclosing ball algorithm in high dimensions. It shows that we can still obtain an efficient $(\sqrt{5} + \sqrt{2}\phi)$ -approximation (with constant probability) in high dimensions that works extremely well in practice.
- We discuss the challenges when Q becomes large and disk-based in Section 6.1, and show how to adapt our algorithms to handle this case efficiently. We also present an interesting variation of the GEQ problem, the constrained GEQ.
- We demonstrate the efficiency, effectiveness and scalability of our algorithms with extensive experiments. These results show that both our exact and approximate methods have significantly outperformed the MBM method up to 6 dimensions. Beyond 6 dimensions and up to very high dimensions ($d = 74$), our approximate algorithm is still efficient and effective, with an average approximation ratio that is close to 1 and very low IO cost.

2.2 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

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.

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.

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 3

3. SOFTWARE REQUIREMENT SPECIFICATION:

H/W System Configuration:-

Processor	- Pentium –III
Speed	- 1.1 Ghz
RAM	- 256 MB(min)
Hard Disk	- 20 GB
Floppy Drive	- 1.44 MB
Key Board	- Standard Windows Keyboard
Mouse	- Two or Three Button Mouse
Monitor	- SVGA

S/W System Configuration:-

- ❖ Operating System : Windows95/98/2000/XP
- ❖ Application Server : Tomcat5.0/6.X
- ❖ Front End : HTML, Java, Jsp
- ❖ Scripts : JavaScript.
- ❖ Server side Script : Java Server Pages.
- ❖ Database Connectivity : Mysql.

CHAPTER 4

4. SOFTWARE DESCRIPTION

Java Technology

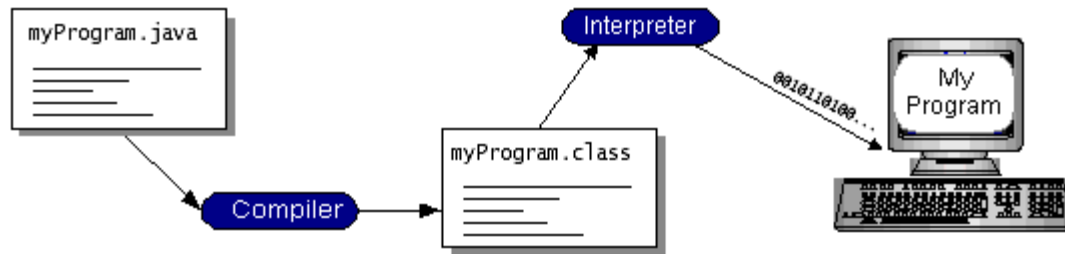
Java technology is both a programming language and a platform.

The Java Programming Language

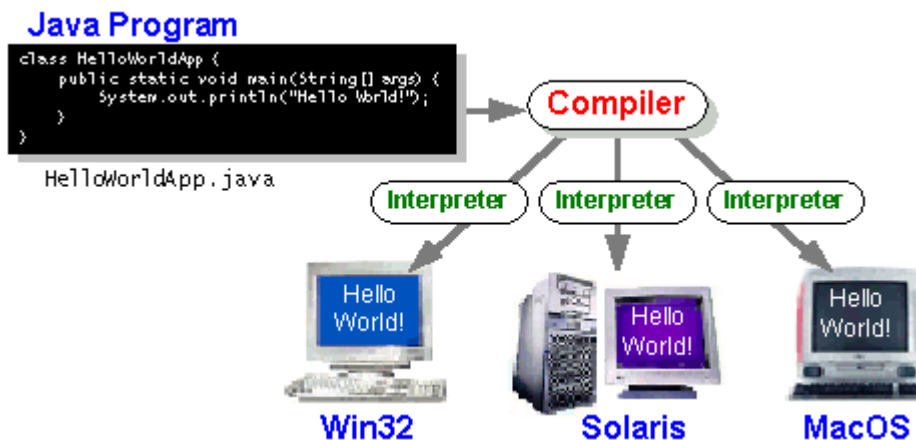
The Java programming language is a high-level language that can be characterized by all of the following buzzwords:

- Simple
- 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.



The 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:

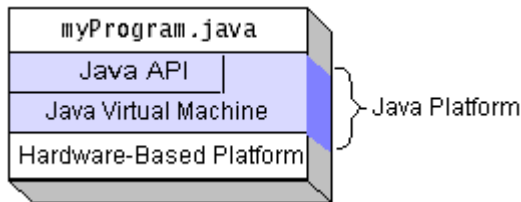
- The *Java Virtual Machine* (Java VM)
- The *Java Application Programming Interface* (Java API)

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.

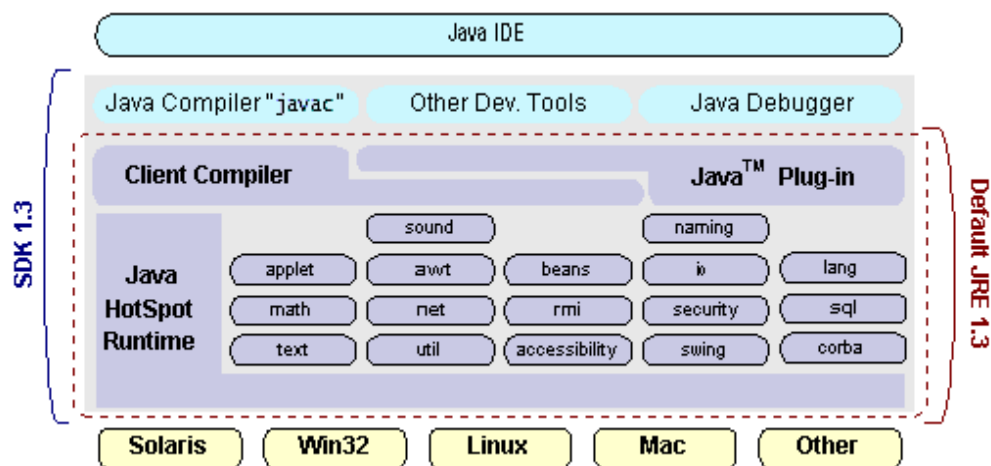
However, the Java programming language is not just for writing cute, entertaining applets for the Web. The general-purpose, high-level Java programming language is also a powerful software platform. Using the generous API, you can write many types of programs.

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 Datagram 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 JavaBeans™, can plug into existing component architectures.
- **Object serialization:** Allows lightweight persistence and communication via Remote Method Invocation (RMI).
- **Java Database Connectivity (JDBC™):** 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 Java™ 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.

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.

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.

2. *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.

3. *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.

4. *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.

5. ***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.

6. ***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.

7. ***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 [Networking](#).

And for dynamically updating the cache table we go for MS [Access](#) database.

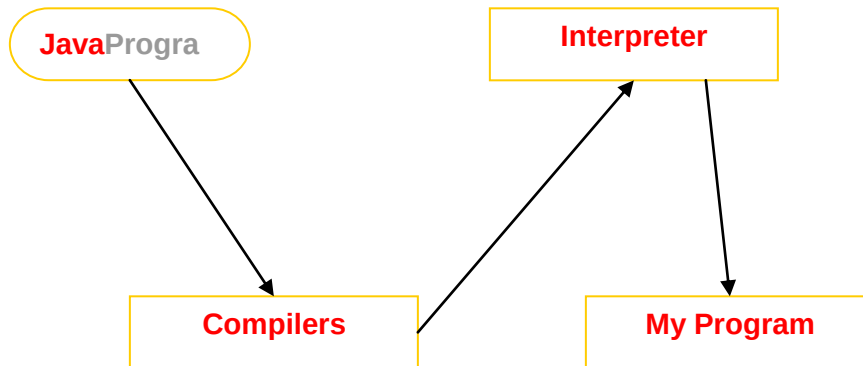
Java ha two things: a programming language and a platform.

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 compiler 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.



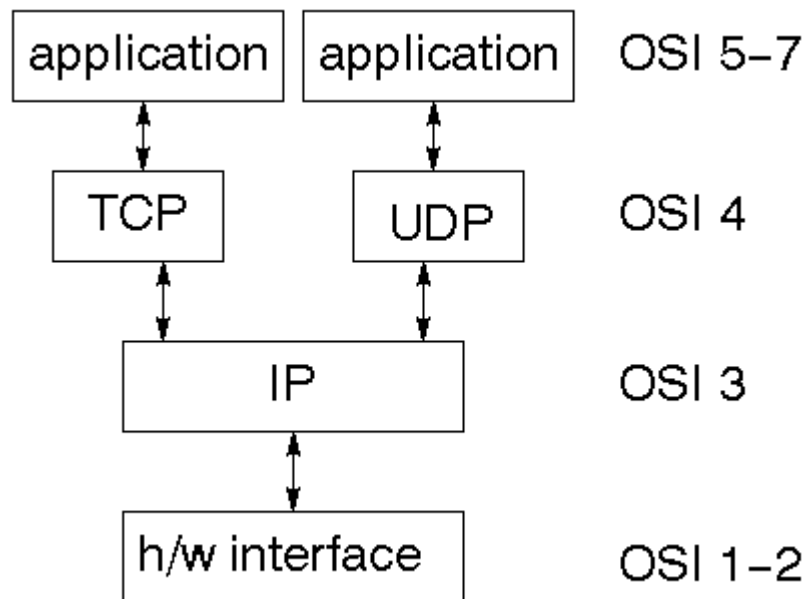
You can think of Java byte codes as the machine code instructions for the Java Virtual Machine (Java VM). Every Java interpreter, whether it's a Java development tool or a Web browser that can run Java applets, is an implementation of the Java VM. The Java VM can also be implemented in hardware.

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

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.

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.

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.

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.

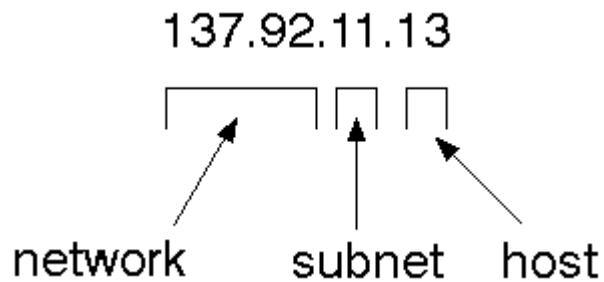
Subnet address

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".

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](#). It is distributed under the terms of the [GNU Lesser General Public Licence](#) (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.

Tomcat 6.0 web server

Tomcat is an open source web server developed by Apache Group. Apache Tomcat is the servlet container that is used in the official Reference Implementation for the Java Servlet and JavaServer Pages technologies. The Java Servlet and JavaServer Pages specifications are developed by Sun under the Java Community Process. Web Servers like Apache Tomcat support only web components while an application server supports web components as well as business components (BEAs Weblogic, is one of the popular application server). To develop a web application with jsp/servlet install any web server like JRun, Tomcat etc to run your application.

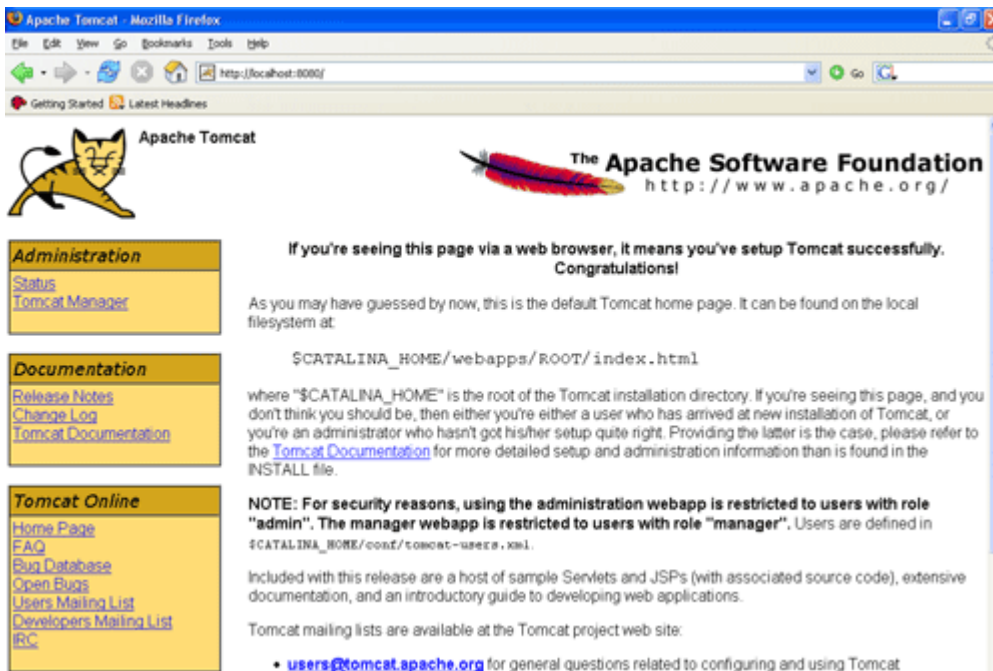
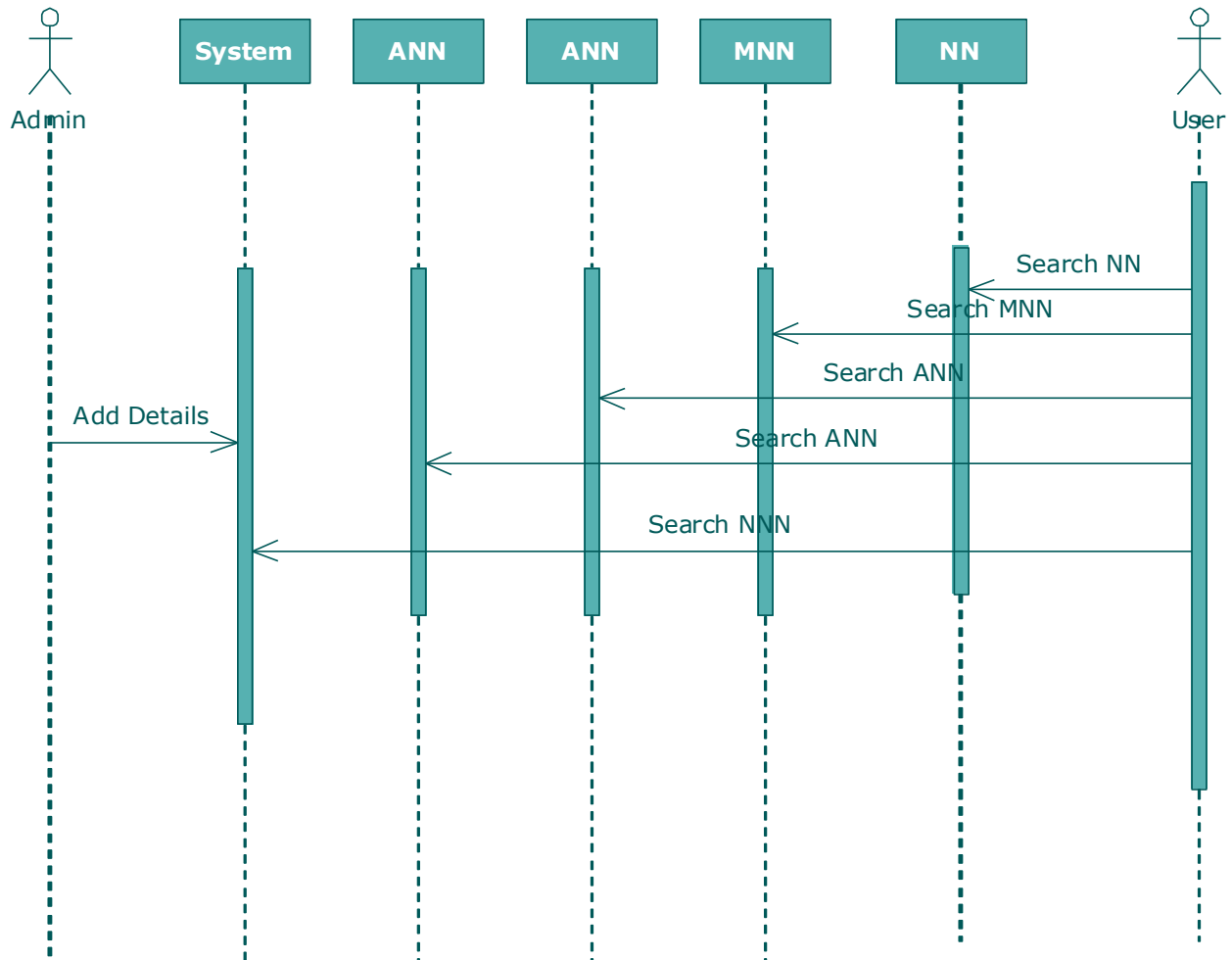


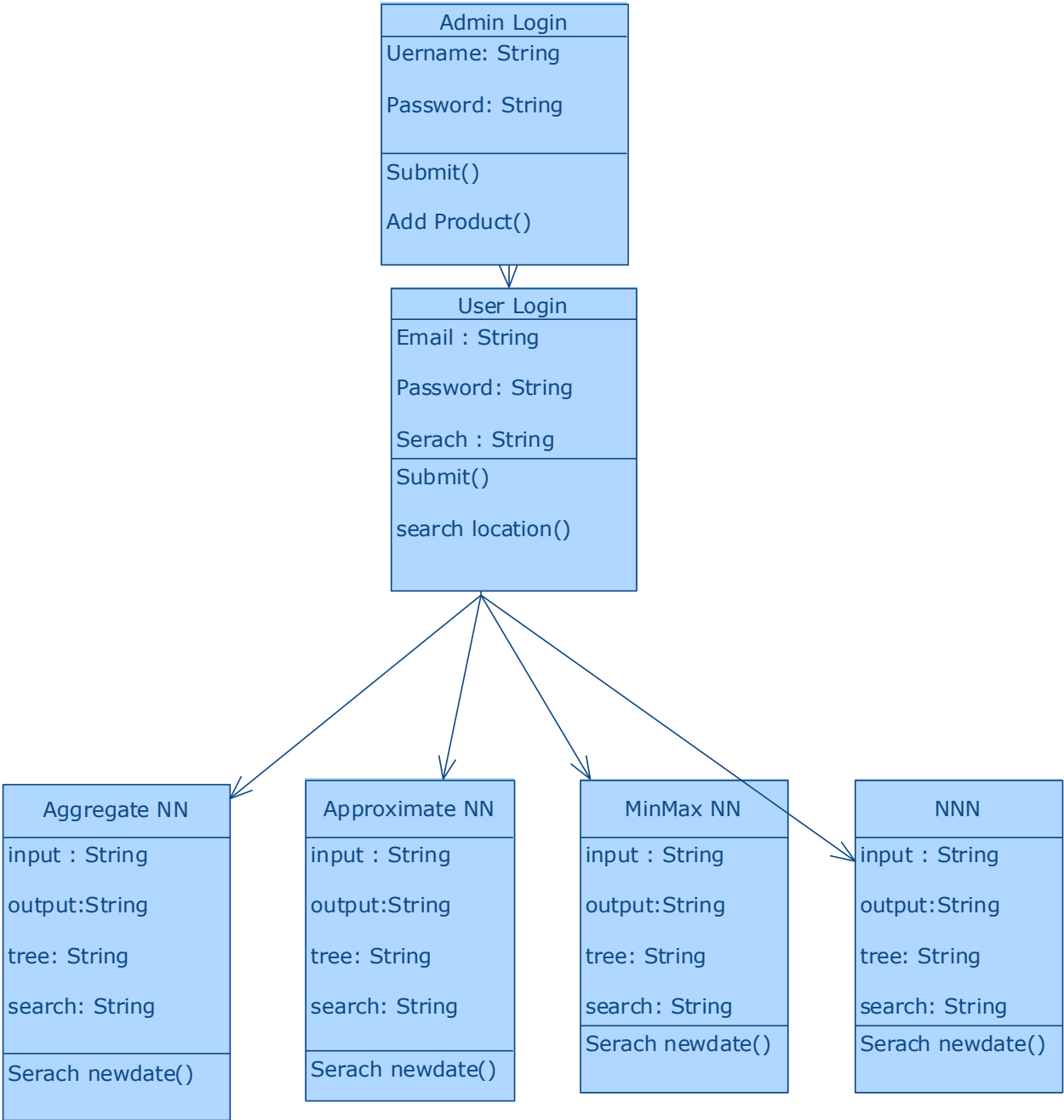
Fig Tomcat Webserver

CHAPTER 5

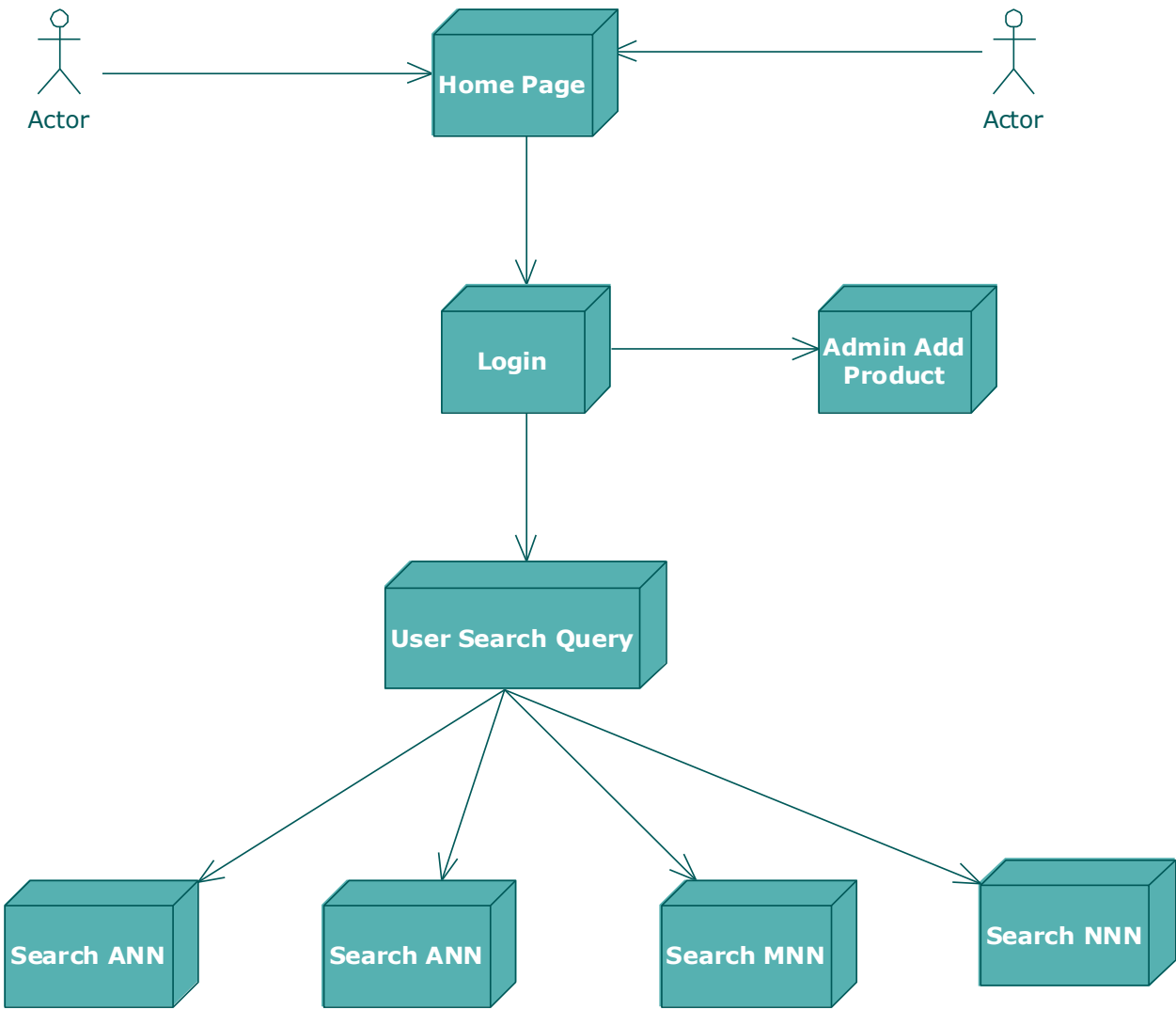
5. SYSTEMDESIGN Sequence Diagram:



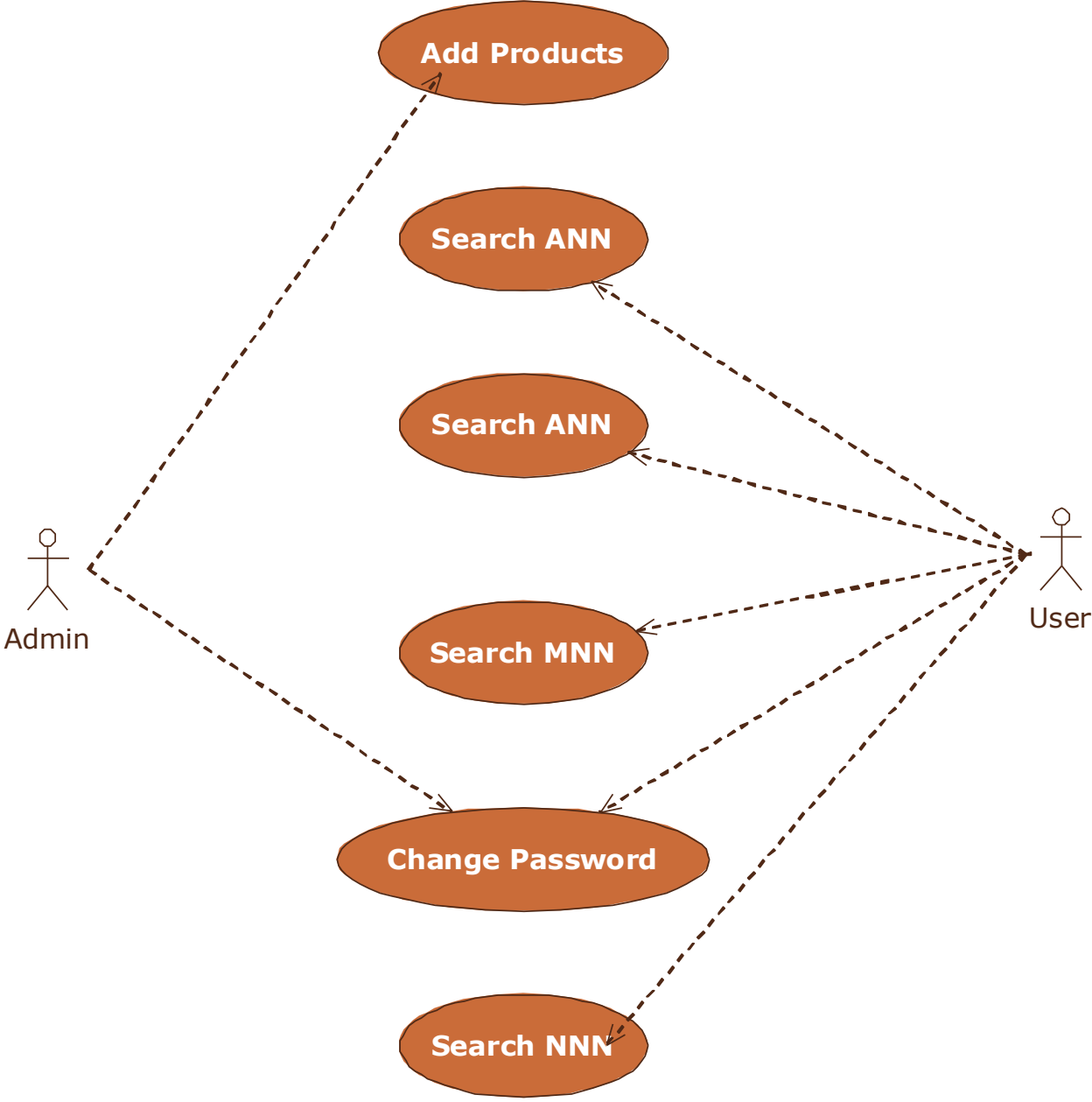
Class Diagram:



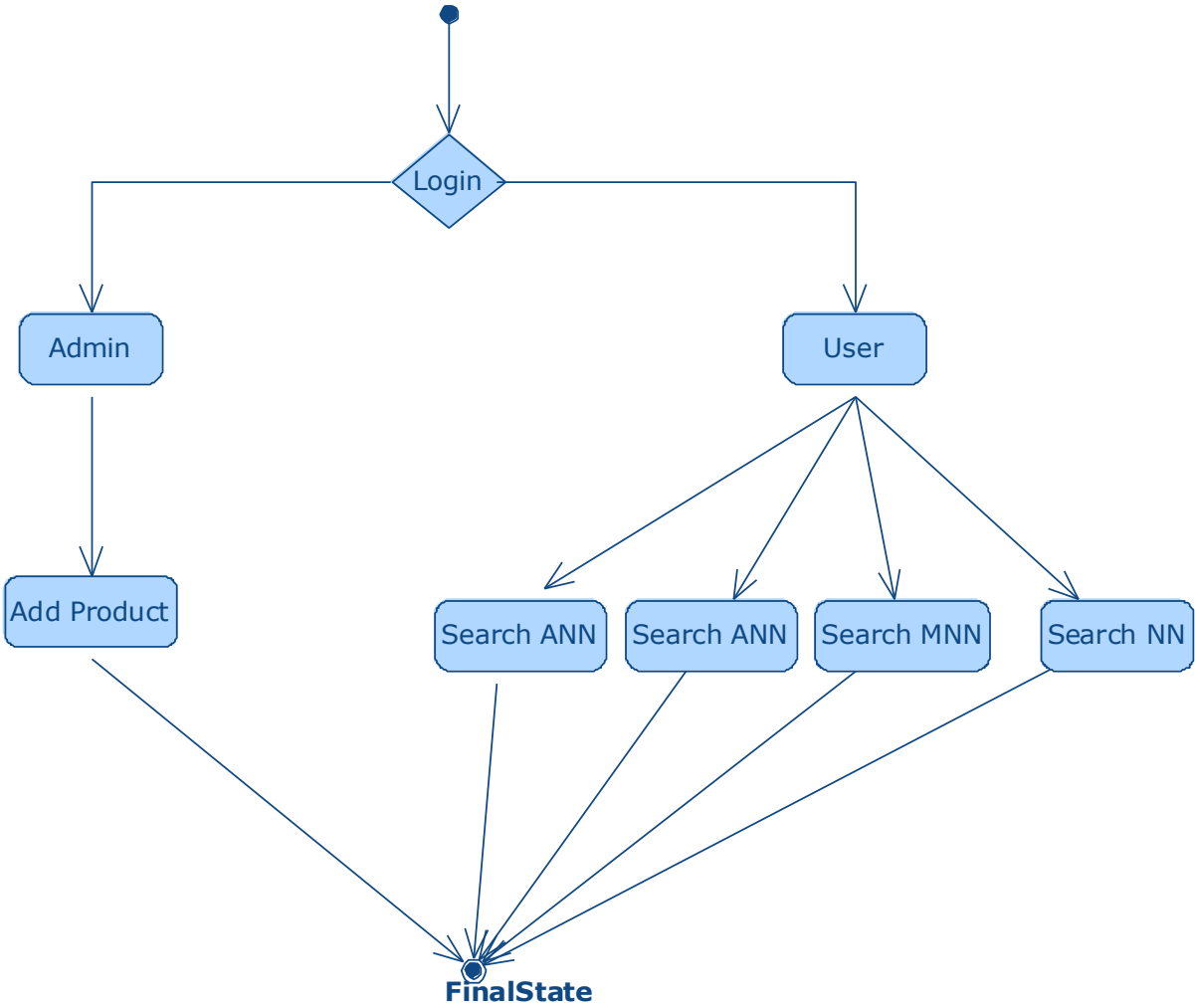
Component Diagram:



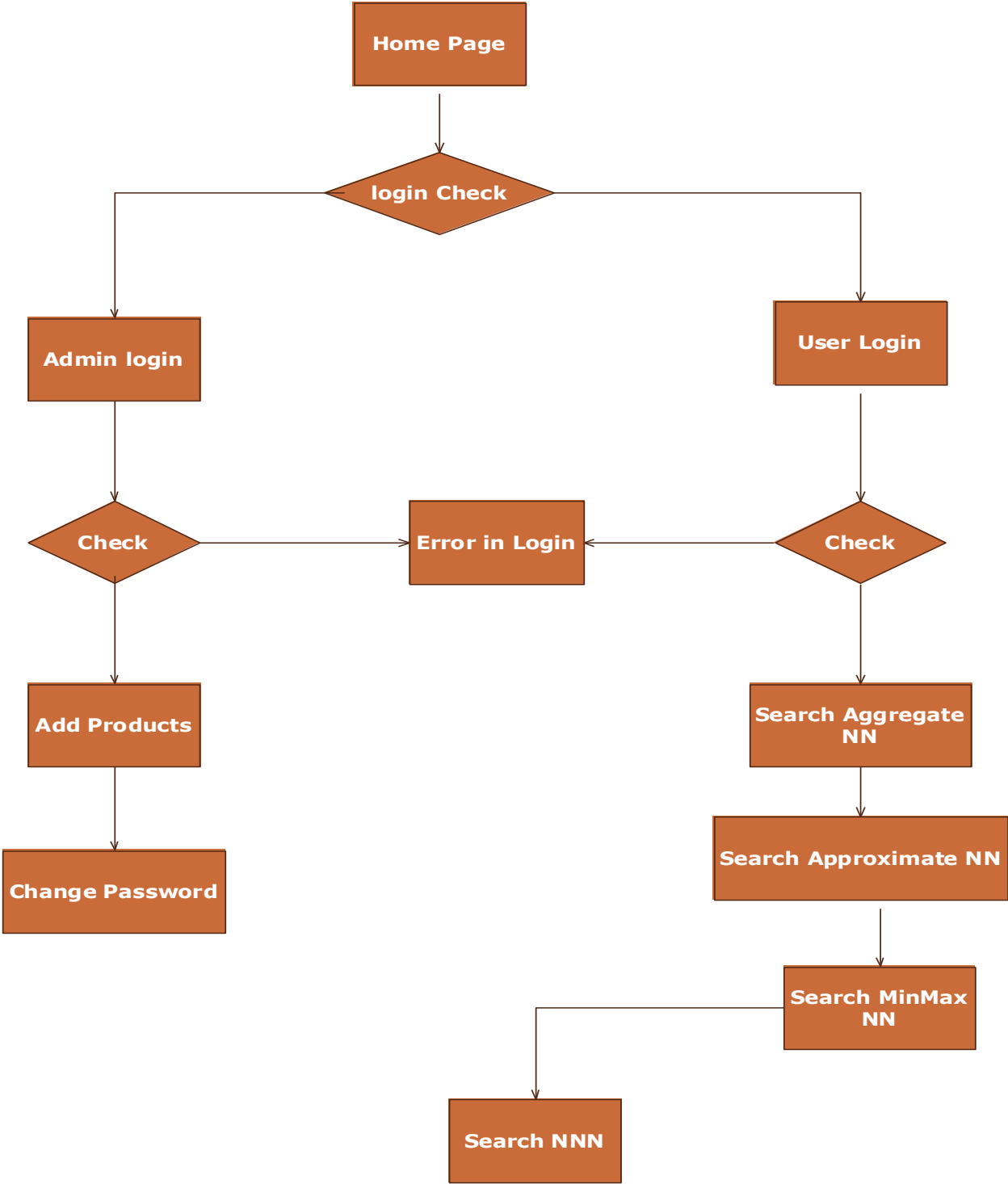
Use case Diagram:



Activity Diagram:



ER Diagram:



CHAPTER 6

Implementation Modules

This work presents new, efficient algorithms, including both exact and approximate versions, for the GEQ problem that significantly outperform the state of the art, the MBM algorithm. Specifically, • We present a new exact search method for the GEQ problem in Section 4 that instantiates several new geometric insights, such as the minimum enclosing ball, the convex hull and the furthest voronoi diagram of the query group, to achieve higher pruning power than the MBM approach.

- 1. Aggregate Nearest Neighbor**
- 2. Approximate Nearest Neighbor**
- 3. MinMax Nearest Neighbor**
- 4. Nearest Neighbor**

Aggregate Nearest Neighbor:

The state of the art method for the GEQ problem is the Minimum Bounding Method (MBM) from. The principal methodology adopted by the MBM is the triangle inequality. It is a heuristic method that has $O(N + M)$ query cost. In practice, its query cost has only been studied in the two-dimensional space. Our experiments over a large number of data sets in Section 7 suggests that the MBM algorithm may lead to high query cost for large data sets and more importantly, its performance degrades significantly with the increase of the dimensionality.

Approximate Nearest Neighbor:

In fact, it is easy to see that any exact search method for the GEQ problem will inevitably suffer from the curse of the dimensionality, since the classical nearest neighbor search is a special instance of the GEQ problem (when Q has only one point). Hence, for data sets in high dimensions, similar to the motivation of doing approximate nearest neighbor search instead of retrieving the exact nearest neighbor in high dimensions (where almost all exact methods degrade to the expensive linear scan of the entire data set), finding efficient and effective Approximation algorithms are the best alternative.

MinMax Nearest Neighbor:

R-tree and its variants have been the most widely deployed indexing structure for the spatial database, or data in multi-dimensions in general. Intuitively, R-tree is an extension of the B-tree to higher dimensions. Points are grouped into minimum bounding rectangles (MBRs) which are recursively grouped into MBRs in higher levels of the tree. The grouping is based on data locality and bounded by the page size. An example of the R-tree is illustrated in Figure 2. Two important query types that we leverage on R-tree are nearest neighbor (NN) queries and range queries.

NN search has been extensively studied, and many related works therein. In particular R-tree demonstrates efficient algorithms using either the depth-first or the best-first approach. The main idea behind these algorithms is to utilize branch and bound pruning techniques based on the relative distances between a query point q to a given MBR N (e.g., minmaxDist , minDist ,).

Nearest Neighbor:

Unfortunately, the worst case query costs are not logarithmic when R-tree is used for NN or range search (even for approximate versions of these queries). To design theoretically sound algorithms with logarithmic costs for our problem, we need a space partition tree with the following properties : (1) The tree has $O(N)$ nodes, $O(\log N)$ depth and each node of the tree is associated with at least one data point. (2) The cells have bounded aspect ratio. (3) The distance of a point to a cell of the tree can be computed in $O(1)$. Arya et.al designed a modification of the standard kd-tree called the Balanced Box Decomposition (BBD) tree which satisfies all these

properties and hence can answer $(1 + \phi)$ -approximate nearest neighbor queries in $O((1/\phi^d) \log N)$ and $(1 + \phi)$ - approximate range search queries in $O((1/\phi^d) + \log N)$. BBD-tree takes $O(N \log N)$ time to build. We use BBD trees in the design of the optimal $(1 + \phi)$ -approximation algorithm with the logarithmic query complexity for

solving the GEQ problem. For nearest neighbor search in high dimensions, all exact methods will eventually degrade to the expensive linear scan of the entire data set and one has to adopt efficient and effective approximate algorithms.

The BBD-tree also becomes impractical for large data sets in high dimensions. In this case, we could use the iDistance index for exact nearest neighbor retrieval (in still relatively low dimensions), or Medrank and

LSH-based methods (locality sensitive hashing) (e.g., the latest development represented by the LSB-tree) for the approximate versions in very high dimensions. Since our idea in designing the approximate algorithms for solving the GEQ problem is to reduce it to the basic nearest neighbor search problem, our approach could leverage on all these techniques for the nearest neighbor search and benefit by *any advancement* in this topic. This is a very appealing feature of our approximation algorithm and makes it extremely flexible and simple for adaptation.

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.

TYPES OF TESTS

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.

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 satisfactory, 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.

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.

System Test

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 the software tester has knowledge of the inner workings, structure and language of the software, or at least its 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. 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 Unit Testing:

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

7.2 Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

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

7.3 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.

Chapter 8

INPUT DESIGN

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.
-

OBJECTIVES

1. Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

3. When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maze of instant. Thus the objective of input design is to create an input layout that is easy to follow

OUTPUT DESIGN

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard

copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.

2. Select methods for presenting information.

3. Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives.

- ❖ Convey information about past activities, current status or projections of the
- ❖ Future.
- ❖ Signal important events, opportunities, problems, or warnings.
- ❖ Trigger an action.
- ❖ Confirm an action.

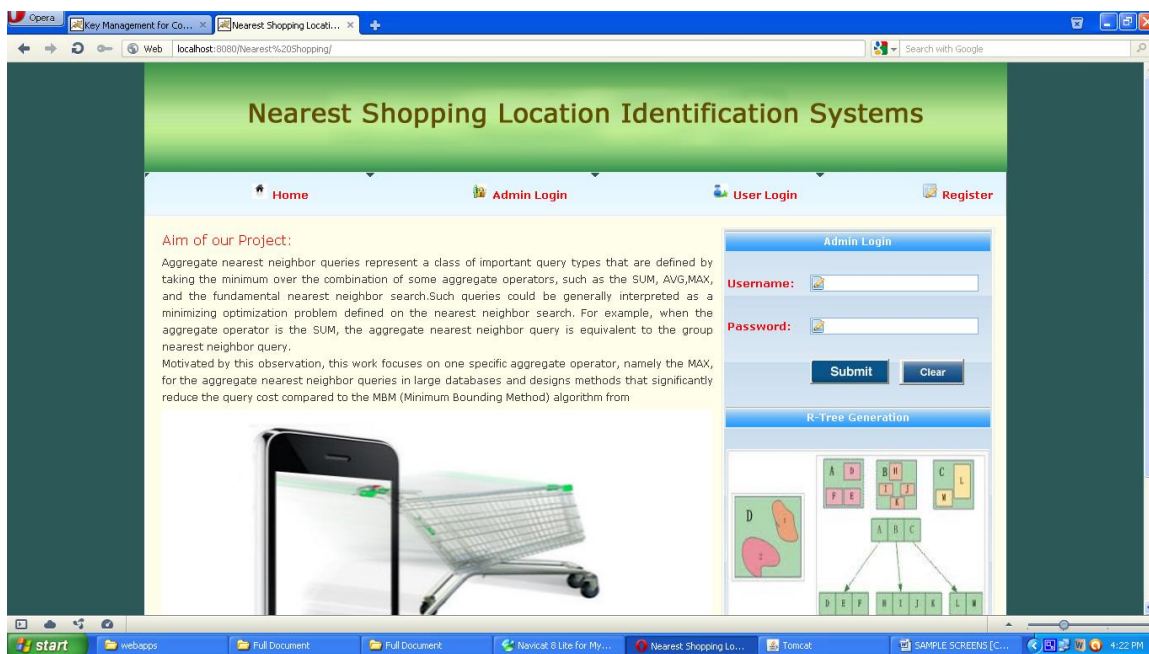
APPENDIX:

A.SCREENSHOTS

Help File

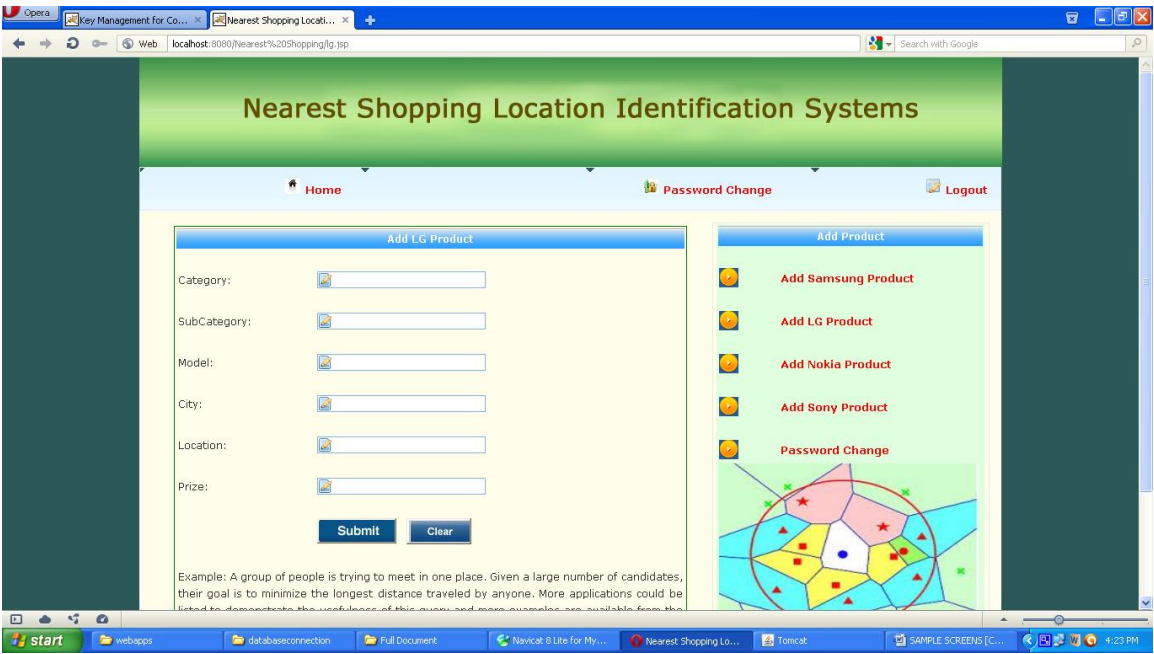
Home Page:

Login Admin:

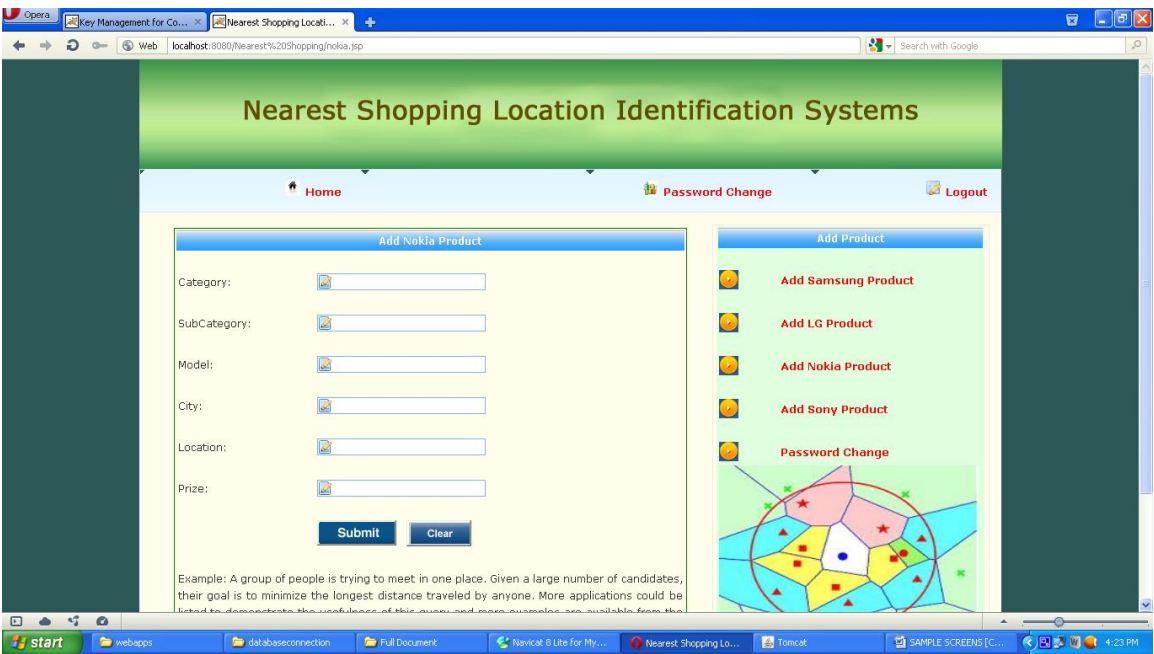


[illegible]48

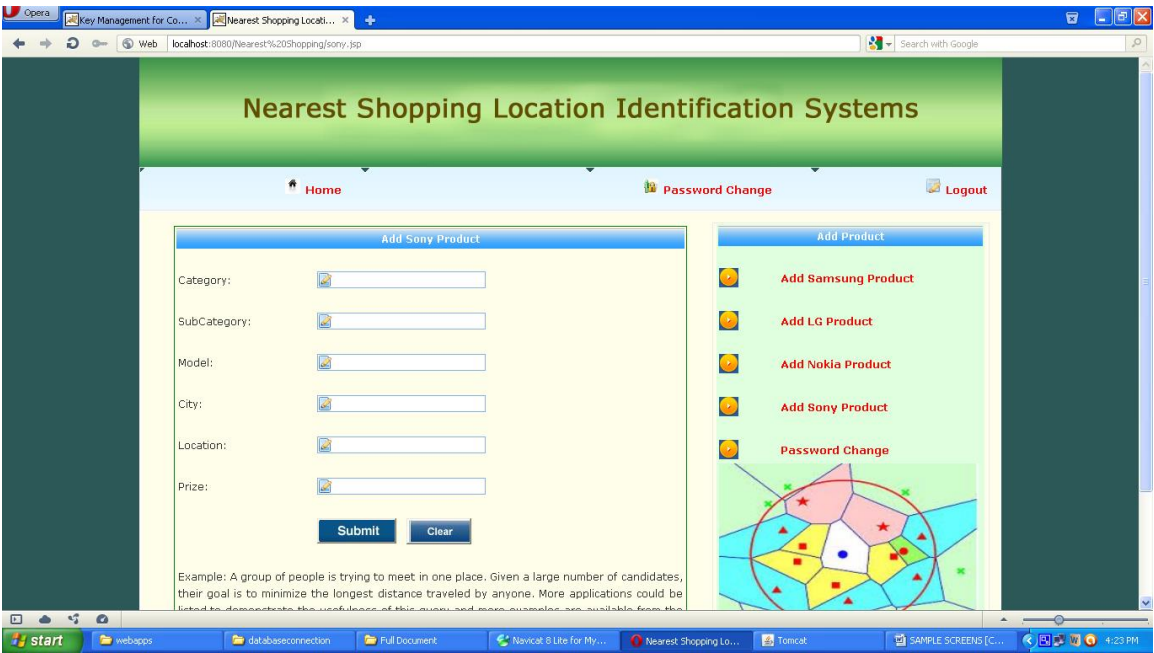
Add Product 2:



Add Product 3:



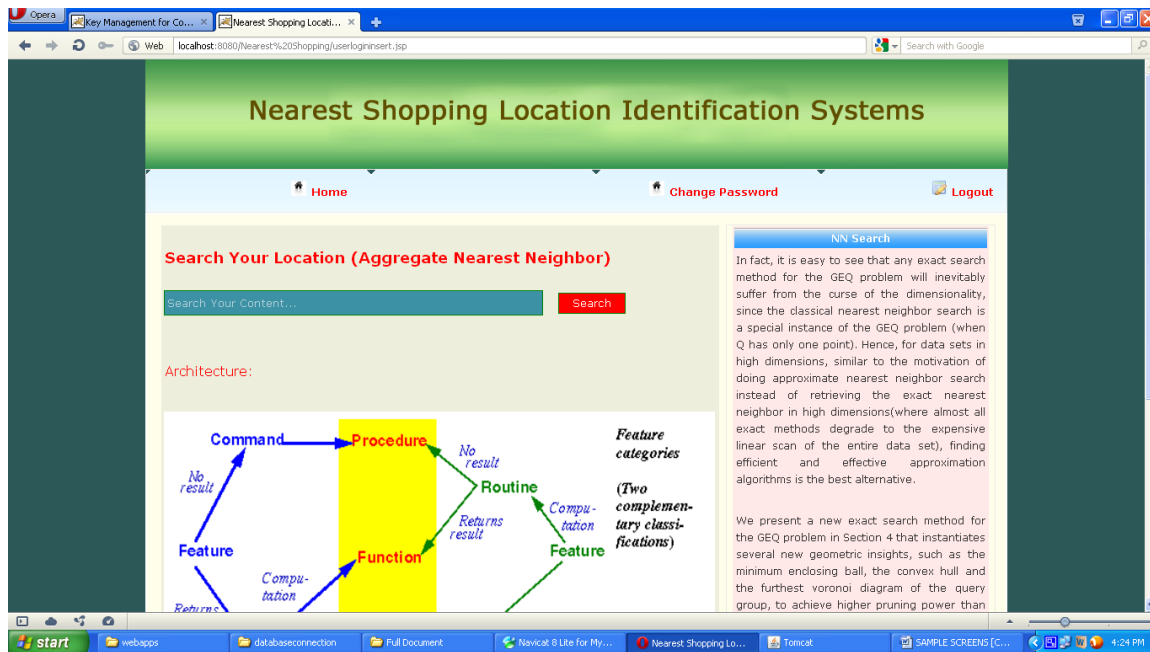
Add Product 4:



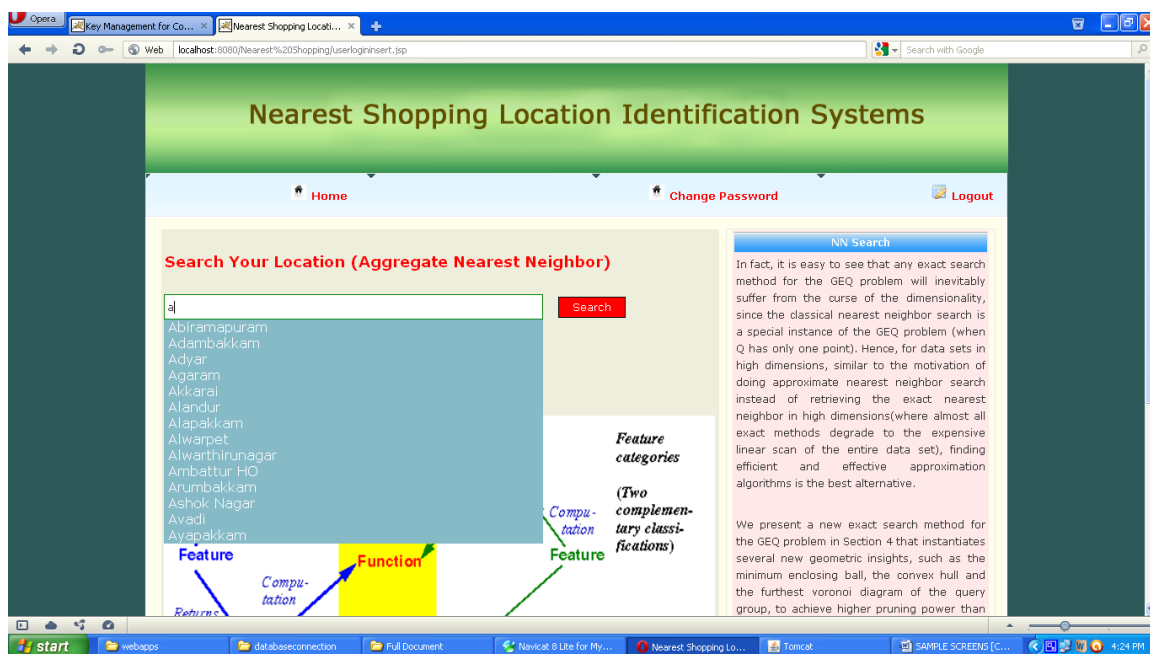
Change Password Admin:



User Login:



Search Location



Search Aggregate NN:

The screenshot shows a web application interface for 'Nearest Shopping Location Identification Systems'. The page has a green header with the title. Below the header, there are navigation links: Home, Change Password, and Logout. The main content area is divided into two sections. On the left, there is a search bar labeled 'Search Your Location' with a 'Search' button. Below the search bar, there is a table with the following data:

Category:	Iphone
Model:	1234
Location:	Kodambakkam
Price:	10000

On the right side of the page, there is a section titled 'Approximate Nearest Neighbor' with a green checkmark icon. Below this, there is a text block explaining the motivation for approximate nearest neighbor search in high dimensions. Further down, there is a section titled 'MinMax Nearest Neighbor' with a green checkmark icon.

The screenshot shows a web application interface for 'Group Enclosing Queries'. The page has a green header with the title. Below the header, there are navigation links: Home, Change Password, and Logout. The main content area is divided into two sections. On the left, there is a search bar labeled 'Search Your Location' with a 'Search' button. Below the search bar, there is a table with the following data:

Category:	LG
Model:	1235

On the right side of the page, there is a section titled 'Approximate Nearest Neighbor' with a green checkmark icon. Below this, there is a text block explaining the motivation for approximate nearest neighbor search in high dimensions. Further down, there is a section titled 'MinMax Nearest Neighbor' with a green checkmark icon. Below this, there is a diagram illustrating the R-tree structure and its use in nearest neighbor search. The diagram shows a tree with nodes labeled $P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_8, P_9, P_{10}$ and $N_1, N_2, N_3, N_4, N_5, N_6, N_7, N_8, N_9, N_{10}$. The diagram also shows the calculation of the minimum distance between a query point q and a node N_i using the formula $\min_{p \in N_i} \text{dist}(q, p)$.

Opera Group Enclosing Queries

Web localhost:8080/Group/usercontent.jsp

Location: **Kodambakkam**

Price: **26999**

R-Tree Generation: **R-Tree Generation**

S1 - Samsung

N1 - Nokia

L1 - LG

S2 - Sony

F1 - Field Name

DB - Database

Kodambakkam

S1 N1 L1 S2

Iphone 1234 10000 LG 1235 20000 Sony 4564 45555 Nokia 1456 26999

Group(DB)

Group Enclosing Queries

recursively grouped into MBRs in higher levels of the tree. The grouping is based on data locality and bounded by the page size. An example of the R-tree is illustrated. Two important query types that we leverage on R-tree are nearest neighbor (NN) queries and range queries.

start Full Document Abstract webapps Abstract - Micro... SAMPLE SCREE... Group Enclosing... Tomcat Group Enclosing... 10:43 AM

Approximate NN:

Opera Key Management for Co... Nearest Shopping Locali...

Web localhost:8080/Nearest%20Shopping/app.jsp

Nearest Shopping Location Identification Systems

Home Change Password Logout

Search Your Location

Search Your Content... Search

Approximate Nearest Neighbor

Category: **Iphone**

Model: **1234**

Location: **Kodambakkam**

Price: **10000**

Approximate Nearest Neighbor

Hence, for data sets in high dimensions, similar to the motivation of doing approximate nearest neighbor search instead of retrieving the exact nearest neighbor in high dimensions (where almost all exact methods degrade to the expensive linear scan of the entire data set), finding efficient and effective approximation algorithms is the best alternative.

MinMax Nearest Neighbor

MinMax Nearest Neighbor

The R-tree

start webapps databaseconnection Full Document Navicat 8 Lite for My... Nearest Shopping Lo... Tomcat SAMPLE SCREENS [C... 4:24 PM

Approximate

Approximate Nearest Neighbor

Category: **Phone**

Model: **1215**

Location: **Vadapalani**

Price: **12990**

R-Tree Generation: **R-Tree Generation**

S1 - Samsung

N1 - Nokia

L1 - LG

S2 - Sony

F1 - Field Name

DB - Database

Vadapalani

S1 N1 L1 S2

Phone 1215 12990

Nearest Neighbor

$GEQ(P = \{p_1, \dots, p_8\}, Q = \{q_1, q_2, q_3\}) = p_3$, the enclosing distance is $\|q_1 - p_3\|$.

R-tree and its variants (e.g., R-tree) have been the most widely deployed indexing structure for the spatial database, or data in multi-dimensions in general. Intuitively, R-tree is an extension of the B-tree to higher dimensions. Points are grouped into minimum bounding rectangles (MBRs) which are recursively grouped into MBRs in higher levels of the tree. The grouping is based on data locality and bounded by the page size. An example of the R-tree is illustrated. Two important query types that we leverage on R-tree are nearest neighbor (NN) queries and range queries.

MinMax NN:

Nearest Shopping Location Identification Systems

Home Change Password Logout

Search Your Location

Search Your Content... Search

MinMax Nearest Neighbor

Category: **Nokia**

Model: **1456**

Location: **Kodambakkam**

Price: **26999**

R-Tree Generation: **R-Tree Generation**

Approximate Nearest Neighbor

Hence, for data sets in high dimensions, similar to the motivation of doing approximate nearest neighbor search instead of retrieving the exact nearest neighbor in high dimensions (where almost all exact methods degrade to the expensive linear scan of the entire data set), finding efficient and effective approximation algorithms is the best alternative.

MinMax Nearest Neighbor

The R-tree

Minmax

The screenshot shows a web application titled "MinMax Nearest Neighbor" running in a browser. The application displays search results for a phone in the Mambalam location. The results are organized into a table with columns for Category, Model, Location, Price, and R-Tree Generation. The search results show a phone with model 1456, located in Mambalam, with a price of 14879. The R-Tree Generation section shows a diagram of an R-tree structure with nodes S1, N1, L1, S2, and F1, and a leaf node containing the phone details: Phone 1456 14879 LG 1235 20000.

MinMax Nearest Neighbor

Category: **Phone**

Model: **1456**

Location: **Mambalam**

Price: **14879**

R-Tree Generation: **R-Tree Generation**

S1 - Samsung

N1 - Nokia

L1 - LG

S2 - Sony

F1 - Field Name

DB - Database

Mambalam

S1 N1 L1 S2

Phone 1456 14879 LG 1235 20000

Nearest Neighbor

$GEQ(P = \{p_1, \dots, p_8\}, Q = \{q_1, q_2, q_3\}) = p_3$, the enclosing distance is $\|q_1 - p_3\|$.

R-tree and its variants (e.g., R-tree) have been the most widely deployed indexing structure for the spatial database, or data in multi-dimensions in general. Intuitively, R-tree is an extension of the B-tree to higher dimensions. Points are grouped into minimum bounding rectangles (MBRs) which are recursively grouped into MBRs in higher levels of the tree. The grouping is based on data locality and bounded by the page size. An example of the R-tree is illustrated. Two important query types that we leverage on R-tree are nearest neighbor (NN) queries and range queries.

Nearest Neighbor:

The screenshot shows a web application titled "Nearest Neighbor" running in a browser. The application displays search results for a phone in the Choolaimedu location. The results are organized into a table with columns for Category, Model, Location, Price, and R-Tree Generation. The search results show a phone with model 6754, located in Choolaimedu, with a price of 59000. The R-Tree Generation section shows a diagram of an R-tree structure with nodes S1, N1, L1, S2, and F1, and a leaf node containing the phone details: Phone 6754 59000.

Nearest Neighbor

Category: **Phone**

Model: **6754**

Location: **Choolaimedu**

Price: **59000**

R-Tree Generation: **R-Tree Generation**

S1 - Samsung

N1 - Nokia

L1 - LG

S2 - Sony

F1 - Field Name

DB - Database

Choolaimedu

S1 N1 L1 S2

Phone 6754 59000

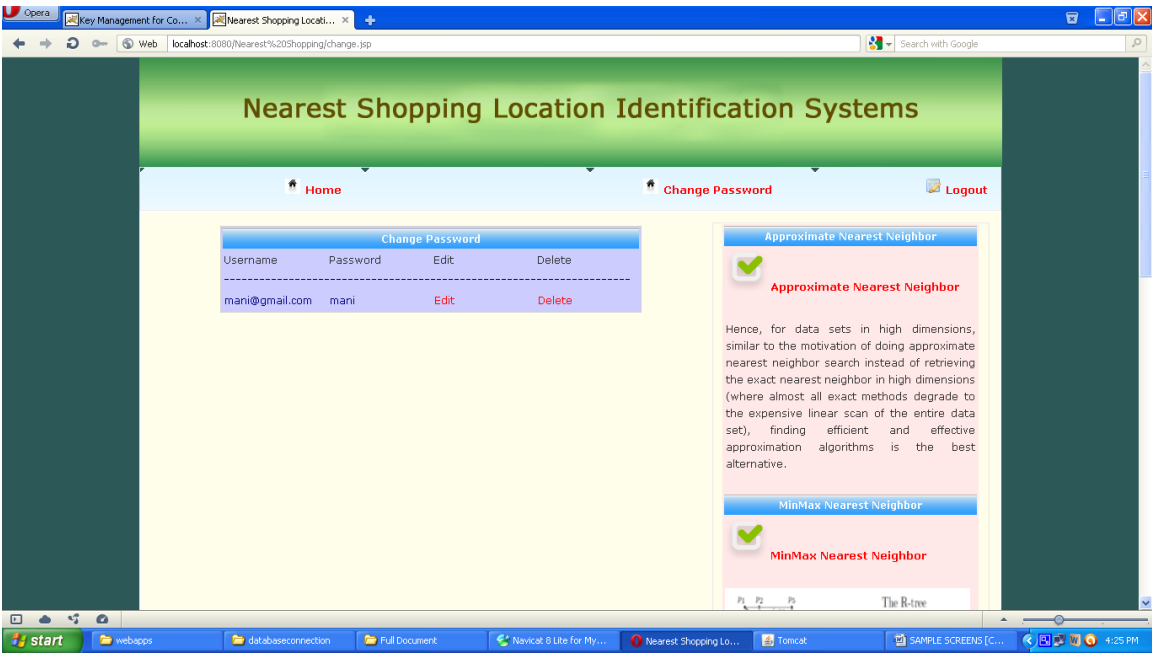
MinMax Nearest Neighbor

MinMax Nearest Neighbor

The R-tree

NN search has been extensively studied, and many related works therein. In particular R-tree demonstrates efficient algorithms using either the depth-first and the best-first approach. The main idea behind these algorithms is to utilize branch and bound pruning techniques based on the relative distances between a query point q to a given MBR N (e.g., minmaxDist, minDist).

Change Password User:



Chapter 9

CONCLUSION

This paper studies an important aggregate nearest neighbor query type (the GEQ problem) that has many real life applications. Our work presents novel exact and approximate methods that have significantly outperformed the existing method for this problem. Our approximate method is simple, efficient and effective. It has a worst-case $\sqrt{2}$ -approximation ratio when exact nearest neighbor of a point can be found and it is close to 1 in practice. In high dimensions, this approximate algorithm could be adapted to work with approximate nearest neighbors and approximate minimum enclosing balls. It gives a (close to) constant approximations and retains it sin practice. Future work includes extending these algorithms to cope with moving objects and data in the road network databases.

Chapter 10

BIBLIOGRAPHY

1. References Made From:

1. Accoria. Rock web server and load balancer. <http://www.accoria.com>.
2. Amazon Web Services. Amazon Web Services (AWS). <http://aws.amazon.com>.
3. V. Cardellini, M. Colajanni, and P. S. Yu. Dynamic load balancing on web-server systems. IEEE Internet Computing, 3(3):28{39, 1999.
4. L. Cherkasova. FLEX: Load Balancing and Management Strategy for Scalable Web Hosting Service. IEEE Symposium on Computers and Communications, 0:8, 2000.
5. F5 Networks. F5 Networks. <http://www.f5.com>.
6. R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. Hypertext transfer protocol { http/1.1. In IETF RFC 2616, 1999.
7. Google Inc. Google App Engine. <http://code.google.com/appengine/>.
8. HaProxy. HaProxy load balancer. <http://haproxy.1wt.eu/>.
9. G. Hunt, E. Nahum, and J. Tracey. Enabling content-based load distribution for scalable services. Technical report, 1997.

10. E. Katz, M. Butler, and R. McGrath. A scalable HTTP server: The NCSA prototype. In Proc. First International Conference on the World Wide Web, Apr. 1994.

2. Sites Referred:

<http://java.sun.com>

<http://www.sourcefordgde.com>

<http://www.networkcomputing.com/>

<http://www.roseindia.com/>

<http://www.java2s.com/>

