CHAPTER: 1

INTRODUCTION

1.1 INTRODUCTION:

High-dimensional vector-based learning, such as face recognition and digit image classification, has benefitted from dimensionality reduction techniques for the low computational complexity and the informative representation of the data structure in the low-dimensional space . According to methodologies of different dimensionality reduction algorithms used, they can be mainly classified into three categories. First, many algorithms, e.g. Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), Maximum Margin Criterion (MMC) (Li et al., 2004, 2006) and Locality Preserving Projections (LPP) (He et al., 2005b), seek linear subspaces to preserve the desired structure of the data in the original Euclidean space. Second, some of the algorithms map the points in Euclidean space to a higher-dimensional Reproducing Kernel Hilbert Space (RKHS), then find a subspace in RKHS using the Representer Theorems (Baudat and Anouar, 2000; Mika et al., 1999; Müller et al., 2001; Schölkopf et al., 1998, 2001).

Third, much research effort has been done by direct non-linearly embedding the data in a global coordinate. These methods include ISOMAP (Tenenbaum et al., 2000), Locally Linear Embedding (LLE) (Roweis and Lawrance, 2000), Local Tangent Space Alignment (LTSA) (Zhang and Zha, 2004) and Laplacian Eigenmaps (Belkin and Niyogi, 2003). They usually make some assumptions on the data point cloud to preserve the local structures. However, the extension to new test data is limited. These three types of methods also have relationships. As we know, linear method can be kernelized if it only involves inner-product. Some direct non-linear embedding methods are also linearlized, then they can be kernelized again (He et al., 2005b,a). The main advantages of the linearlization are: (1) it can be directly used to do induction, which means it can handle the new test points, (2) the linearlized algorithm can keep the good properties of the original non-linear methods. For instance, the linear method LPP (He et al.,2005b) linearlizes the manifold-based method Laplacian Eigenmaps (Belkin and Niyogi, 2003). Thus, LPP also has the locality preserving property in the Euclidean space and achieves good results. In general, the above linearlization methods are all unsupervised, which means they cannot use any label information.1 Recently, some supervised dimensionality reduction algorithms brought the idea from the mentioned unsupervised methods. They start from the local structure of data and preserve the geometric information provided by both the data point cloud and the label information (Cai et al., 2007; Chen et al., 2005; Nie et al., 2007;

Sugiyama, 2006; Yan et al., 2007). Sometimes, we cannot expect that there are sufficient labeled data samples to realize dimensionality reduction for classification task, since labeling work is both time consuming and costly. Conversely, we can obtain many unlabeled data in the real world. Thus, semi-supervised dimensionality reduction, using some labeled data and lots of unlabeled data to find a low-dimensional representation, is of great practical importance (Zhu, 2005). In this paper, we propose a novel semi-supervised manifold discriminant analysis-based dimensionality reduction algorithm. We start from local and embed the data in global coordinate, which can separate each sub-manifold constructed by one class. Based on this idea, we define the within-manifold, between-manifold and totalmanifold scatter matrices. The former two are supervised and the last is unsupervised constructed by labeled and unlabeled data. Semi-supervised dimensionality reduction is realized by both maximizing the between-manifold scatter and minimizing the withinmanifold scatter, simultaneously preserving the local structure of a manifold and the intrinsic geometry of both labeled and unlabeled data.

It is interesting of our method from the following perspectives:

(1) The proposed between-manifold scatter matrix is robust to outlier.

(2) It can be easily kernelized, and both the linear and kernel versions can do induction.

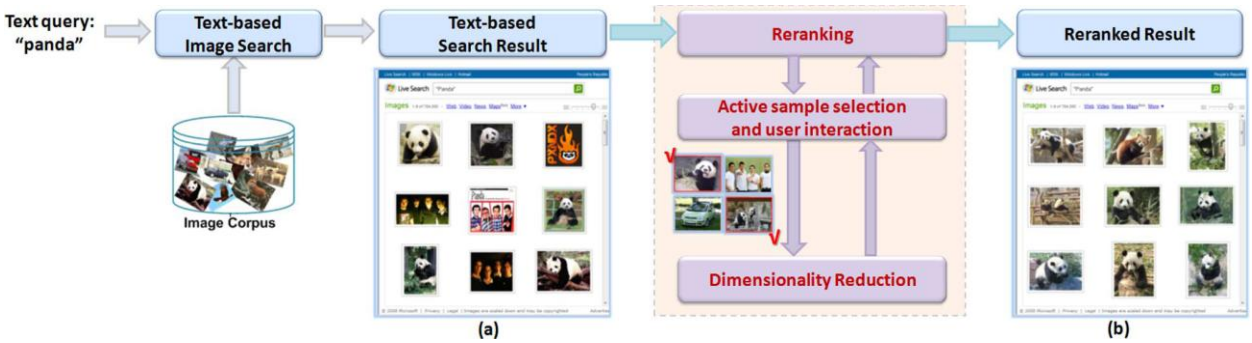
(3) The proposed algorithm framework can be modified to do direct non-linear embedding. It charts each sub-manifold for each class.

We have three types of images: labelled relevant, labeled irrelevant, and unlabelled. Therefore, we build 3 types of patches, which are:

1) local patches for labelled relevant images to represent the local geometry of them and the discriminative information to separate relevant images from irrelevant ones,

2) local patches for labelled irrelevant images to represent the discriminative information to separate irrelevant images from relevant ones, and

3) global patches for both labelled and unlabelled images for transferring both the local geometry and the discriminative information from all labelled images to the unlabelled ones.



1.2 SCOPE OF THE PROJECT:

This Project Uses a novel active reranking frameworkfor Web image search by using user interactions. To target the user's intention effectively and efficiently, we have proposed an active sample selection strategy and a dimension reduction algorithm, to reduce labeling efforts and to learn the visual characteristics of the intention respectively. To select the most informative query images, the structural information based active sample selection strategy takes both the ambiguity and the representativeness into consideration. To learn the visual characteristics, a new local-global discriminative dimension reduction algorithm transfers the local information in the domain of the labelled images domain to the whole image database. The experiments on both synthetic datasets and a real Web image search dataset have demonstrated the effectiveness of the proposed active reranking scheme, including both the sample selection strategy and the dimension reduction algorithm.

1.3 PROJECT FEATURES:

Reranking with user interactions,or active reranking ,is highly demanded to effectively improve the search performance.the essential problem is how to capture the user's intention.

To complete this goal ,this paper presents a structural information based sample selection strategy to reduce the user's labeling efforts.Furthermore ,to localize the user's intention in the visual feature space, a novel local-global discriminative dimension reduction algorithm is proposed.

CHAPTER-2

SYSTEM ANALYSIS

2.1 EXISTING SYSTEM

Although text based search techniques have shown their effectiveness in the document search,they are problematic when applied to the image search.There are two main problems.One is mismatching between images and their associated textual information.resulting into irrelevant images appearing in the search results.For example,an image which is irrelevant to “panda” will be mistaken as a relevant image if there is a word “panda”existing in its surrounding text.The other problem is that the textual information is insufficient to represent the semantic content of images.The same query words may refer to images that are semantically different.e.g,we can not differentiate an animal panda image from an image for a person whose name is panda,just with the text word “panda”.

2.2 PROPOSED SYSTEM

In active re ranking, the essential problem is how to capture the user’s intention, i.e., to distinguish query relevant images from irrelevant ones. Different from the conventional learning problems, in which each sample only has one fixed label, an image may be relevant for one user but irrelevant for another. In other words, the semantic space is user-driven, according to their different intentions but with identical query keywords. Therefore, we propose to target the user-driven intention from two aspects: collecting labeling information from users to obtain the specified semantic space, and localizing the visual characteristics of the user’s intention in this specific semantic space, respectively. Although Intent Search can be deemed as a simplified version of active re ranking, i.e., the user’s intention is defined by only one query image, it can not work well when the user’s intention is too complex to be represented by one image. As shown in Fig. 3, the query relevant images for “Animal” vary largely both in visual appearance and features, thus we cannot represent “Animal” only with one image. Instead, our proposed active re ranking method can learn the user’s intention more extensively and completely.

2.3 FEASIBILITY STUDY:

A feasibility study is a high-level capsule version of the entire System analysis and Design Process. The study begins by classifying the problem definition. Feasibility is to determine if it's worth doing. Once an acceptance problem definition has been generated, the analyst develops a logical model of the system. A search for alternatives is analyzed carefully. There are 3 parts in feasibility study.

Technical Feasibility:

Evaluating the technical feasibility is the trickiest part of a feasibility study. This is because, at this point in time, not too many detailed design of the system, making it difficult to access issues like performance, costs on (on account of the kind of technology to be deployed) etc. A number of issues have to be considered while doing a technical analysis. Understand the different technologies involved in the proposed system before commencing the project we have to be very clear about what are the technologies that are to be required for the development of the new system. Find out whether the organization currently possesses the required technologies. Is the required technology available with the organization?.

Operational Feasibility:

Proposed project is beneficial only if it can be turned into information systems that will meet the organizations operating requirements. Simply stated, this test of feasibility asks if the system will work when it is developed and installed. Are there major barriers to Implementation? Here are questions that will help test the operational feasibility of a project:

1. Is there sufficient support for the project from management from users? If the current system is well liked and used to the extent that persons will not be able to see reasons for change, there may be resistance.

2. Are the current business methods acceptable to the user? If they are not, Users may welcome a change that will bring about a more operational and useful systems.

3. Have the user been involved in the planning and development of the project?

4. Early involvement reduces the chances of resistance to the system and in general and increases the likelihood of successful project.

Since the proposed system was to help reduce the hardships encountered. In the existing manual system, the new system was considered to be operational feasible.

Economic Feasibility:

Economic feasibility attempts 2 weigh the costs of developing and implementing a new system, against the benefits that would accrue from having the new system in place. This feasibility study gives the top management the economic justification for the new system.A simple economic analysis which gives the actual comparison of costs and benefits are much more meaningful in this case. In addition, this proves to be a useful point of reference to compare actual costs as the project progresses. There could be various types of intangible benefits on account of automation. These could include increased customer satisfaction, improvement in product quality better decision making timeliness of information, expediting activities, improved accuracy of operations, better documentation and record keeping, faster retrieval of information, better employee morale.

Legal Feasibility:

Determines whether the proposed system conflicts with legal requirements, e.g. a data processing system must comply with the local Data Protection Acts.

Schedule feasibility

A project will fail if it takes too long to be completed before it is useful. Typically this means estimating how long the system will take to develop, and if it can be completed in a given time period using some methods like payback period. Schedule feasibility is a measure of how reasonable the project timetable is. Given our technical expertise, are the project deadlines reasonable? Some projects are initiated with specific deadlines. You need to determine whether the deadlines are mandatory or desirable.

Cultural Feasibility:

A cultural feasibility study is defined as one that investigates scientific as well as ethical, behavioral, and social issues in the design of clinical trials. The value of such a broadly defined assessment is illustrated through the presentation of two case studies conducted to prepare for clinical trials to reduce maternal-infant HIV transmission on Cité Soleil, Haiti. The first study addressed issues surrounding a trial of breast-feeding and exclusive bottle-feeding among HIV seropositive mothers. The second study focused on the implementation of a double-blind trial of HIV immune globulin and standard immune globulin to be administered to infants of seropositive mothers shortly after birth. Both cases used focus group interviews with mothers and in-depth interviews with key informants to investigate AIDS-related beliefs, acceptability of trial participation, risks to subjects, and community reactions and repercussions to the trial. Findings point to the difficulties posed by attempts to conduct trial involving complex research designs in socially disadvantaged populations. Recommendations highlight the need to consider the community-wide impact of a trial, and the need to undertake extensive educational preparation of participants to ensure informed consent and adherence to protocols.

When we have a plan to run a business or project, We should consider whether we can run the business well. We should identify business-related with the project. We must think capital first because we cannot run a business without capital. If we need capital from bank, saving union or other financial institution

We must provide feasibility study.

Why businessmen are bothered to make feasibility study? The feasibility study depicts the project or business well from various side. It helps the business owner, the investor, and the manager to appraise whether the project can be executed or not. The feasibility support you to :

- List all detail your business need. It help us to provide whatever we need for company operational.
- Check the logistical problem and solution. The operation may be stuck if there is a lack of logistical.
- Convince the bank your strategy so they attract to fund your project.

There are at least some feasibility like:

1. Technology feasibility. What will technology that the company uses? Sometimes technology can help the company more efficient. Identify whether the technology is effective to the company.

2. Economic feasibility. Calculate the benefit of the project and the cost of the project. List what the income

and expenses of the product or project.

3. Legal feasibility. Does the project break the law? Identify whether the project break the law or not.

4. Operation feasibility. How the operation of project run? What is the obstacle of the project operation? Sometimes it is difficult to operate the company.

5. Schedule of the project. Some times the project has not finished yet and it harm to the project. For example, a dam project should be finished before rain season. If the worker cannot finish at dry season, the project will fail and water will float the settlement.

6. Market or demand of the product:

Will the product sold out? Does the people need it and want buy the product. Research the market by surveying the demand. Find the effective cost to market your product. Make description about the industry, current market, anticipated future market potential, competition, and others.

7. Resource feasibility: Check the resources of the project. Is the resources easy to get or difficult to get it? The company have to guarantee the sufficient resources and cheap. Therefore, they can thrift some expenses.

8. Cultural feasibility :

Check whether the culture accept the project or they refuse the project. Some cultural does not like mining company.

CHAPTER-3

SYSTEM SPECIFICATION

HARDWARE & SOFTWARE REQUIREMENTS:

HARDWARE REQUIREMENTS:

- System : Pentium III and above.
- Hard Disk : 40 GB and above.
- Floppy Drive : 1.44 Mb and above.
- Monitor : 15 VGA Color and above.
- Mouse : Logitech and above.
- Ram : 512 MB and above.

SOFTWARE REQUIREMENTS:

- Operating system : Windows XP Professional (optional)
- Coding Language : java(jdk1.6.0)
- IDE : Integrated Development Environment
- Front End : Struts Framework
- Back End : Oracle 10g

Functional Requirements:

1) Searching text query
2) Show images based on text query
3) Identifying the relevant images and irrelevant images
4) Identifying text based results
5) Find the number of original images
6) Find the number of duplicate images
7) Delete the irrelevant iFind the reran ked results

NON FUNCTIONAL REQUIREMENTS

Non- Functional Requirements:

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

1. **Usability**

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

2. **Reliability**

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

3. **Performance**

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

4. **Supportability**

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

5. **Implementation**

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

6. Interface The user interface is based on HTML and XHTML.

MODULE ANALYSIS:

Modules:

1. Label information collection (Textual information search)
2. Visual characteristic localization
3. Re ranking implementation
4. Active sample selection

Label information Collection:

To collect the labeling information from users efficiently, a new structural information based strategy is proposed to actively select the most informative query images. It is boring and unacceptable to keep asking a user to label a lot of images in the interaction stage. Thus, it is essential to get the necessary information by labeling as few images as possible. Active learning is well-known for reducing the labeling efforts, by labeling most informative samples. Conventional active learning strategies can be divided into two categories: the error reduction strategy and the most uncertain (close-to-boundary) strategy. Both of them suffer from the small sample size problem, i.e., the unreliable estimation of the expected error risk and the uncertainty caused by the insufficient labelled samples.

Visual characteristic localization:

To localize the visual characteristics of the user's intention, we propose a novel local-global discriminative (LGD) dimension reduction algorithm. Basically, we assume that the query relevant images, which represent the user's intention, are lying on a low-dimensional sub manifold of the original ambient (visual feature) space. LGD learns this sub manifold by transferring both the local geometry and the discriminative information from labelled images to unlabelled ones. The learned sub manifold preserves both the local geometry

of labelled relevant images and the discriminative information to separate relevant from irrelevant images. As a consequence, we can eliminate the well-known semantic gap between low-level visual features and high-level semantics to further enhance the re ranking performance on this sub manifold.

Re ranking implementation process:

The proposed general framework for active re ranking in Web image search. Take the query term "panda" as an example. When "panda" is submitted to the Web image search engine, an initial text-based search result is returned to the user, as shown in Fig. 1(a) (only the top nine images are given for illustration). This result is unsatisfactory because both person and animal images are retrieved as top results. This is caused by the ambiguity of the query term. Without the user interactions, it is impossible to eliminate this ambiguity. In particular, which kind of images, animal panda or person whose name is Panda, are user's intention? Therefore, traditional re ranking methods, which improve the initial search results by only utilizing the visual property of images, cannot achieve good performances.

Active Sample Selection:

An SInfo active sample selection strategy is presented to learn the user's intention efficiently which selects images by considering not only the ambiguity but also the representativeness in the whole image database. Ambiguity and representativeness are two important aspects in active sample selection. Labeling a sample which is more ambiguous will bring more information. On the other side, the information provided by individual sample can be shared by its neighbors. Therefore, the more representative samples are preferred for labeling. In SInfo , the ambiguity of an image is measured by the entropy of the relevance probability distribution while the representativeness is measured by the density.

CHAPTER-4

LANGUAGE DESCRIPTION

HTML, an initialize of Hypertext Markup Language, is the predominant markup language for web pages. It provides a means to describe the structure of text-based information in a document — by denoting certain text as headings, paragraphs, lists, and so on — and to supplement that text with interactive forms, embedded images, and other objects. HTML is written in the form of labels (known as tags), surrounded by angle brackets. HTML can also describe, to some degree, the appearance and semantics of a document, and can include embedded scripting language code which can affect the behavior of web browsers and other HTML processors.

Hypertext Markup Language (HTML), the languages of the World Wide Web (WWW), allows users to produces Web pages that include text, graphics and pointer to other Web pages (Hyperlinks).

HTML is not a programming language but it is an application of ISO Standard 8879, SGML (Standard Generalized Markup Language), but specialized to hypertext and adapted to the Web. The idea behind Hypertext is that instead of reading text in rigid linear structure, we can easily jump from one point to another point. We can navigate through the information based on our interest and preference. A markup language is simply a series of elements, each delimited with special characters that define how text or other items enclosed within the elements should be displayed. Hyperlinks are underlined or emphasized works that load to other documents or some portions of the same document.

HTML provides tags (special codes) to make the document look attractive. HTML tags are not case-sensitive. Using graphics, fonts, different sizes, color, etc., can enhance the presentation of the document. Anything that is not a tag is part of the document itself.

Basic HTML Tags:

<! -- -->	Specifies comments
<A>……</A>	Creates hypertext links
<B>……</B>	Formats text as bold
<BODY>…</BODY>	Contains all tags and text in the HTML document
<FORM>...</FORM>	Encloses a fill-out form

<FRAME>...</FRAME>	Defines a particular frame in a set of frames
<H#>…</H#>	Creates headings of different levels (1 – 6)
<HEAD>...</HEAD>	Contains tags that specify information about a document
<HTML>…</HTML>	Contains all other HTML tags
<META>...</META>	Provides meta-information about a document
<SCRIPT>…</SCRIPT>	Contains client-side or server-side script
<TABLE>…</TABLE>	Creates a table
<TD>…</TD>	Indicates table data in a table
<TR>…</TR>	Designates a table row
<TH>…</TH>	Creates a heading in a table

Advantages:

- A HTML document is small and hence easy to send over the net. It is small because it does not include formatted information.
- HTML is platform independent.
- HTML tags are not case-sensitive.

JavaScript:

JavaScript is a script-based programming language that was developed by Netscape Communication Corporation. JavaScript was originally called Live Script and renamed as JavaScript to indicate its relationship with Java. JavaScript supports the development of both client and server components of Web-based applications. On the client side, it can be used to write programs that are executed by a Web browser within the context of a Web page. On the server side, it can be used to write Web server programs that can process information submitted by a Web browser and then update the browser's display accordingly

Even though JavaScript supports both client and server Web programming, we prefer JavaScript at Client side programming since most of the browsers supports it. JavaScript is almost as easy to learn as HTML, and JavaScript statements can be included in HTML documents by enclosing the statements between a pair of scripting tags

<SCRIPTS> . . . </SCRIPT>.

<SCRIPT LANGUAGE = "JavaScript">

JavaScript statements

</SCRIPT>

Here are a few things we can do with JavaScript:

- Validate the contents of a form and make calculations.
- Add scrolling or changing messages to the Browser's status line.
- Animate images or rotate images that change when we move the mouse over them.
- Detect the browser in use and display different content for different browsers.
- Detect installed plug-ins and notify the user if a plug-in is required.

We can do much more with JavaScript, including creating entire application.

Advantages:

- JavaScript can be used for Sever-side and Client-side scripting.
- It is more flexible than VBScript.
- JavaScript is the default scripting languages at Client-side since all the browsers supports it.

Java Technology:

Initially the language was called as "oak" but it was renamed as "Java" in 1995. The primary motivation of this language was the need for a platform-independent (i.e., architecture neutral) language that could be used to create software to be embedded in various consumer electronic devices.

- Java is a programmer's language.
- Java is cohesive and consistent.
- Except for those constraints imposed by the Internet environment, Java gives the programmer, full control.
- Finally, Java is to Internet programming where C was to system programming.

Importance of Java to the Internet:

Java has had a profound effect on the Internet. This is because; Java expands the Universe of objects that can move about freely in Cyberspace. In a network, two categories of objects are transmitted between the Server and the Personal computer. They are: Passive information and Dynamic active programs. The Dynamic, Self-executing programs cause serious problems in the areas of Security and probability. But, Java addresses those concerns and by doing so, has opened the door to an exciting new form of program called the Applet.

Java can be used to create two types of programs:

Applications and Applets: An application is a program that runs on our Computer under the operating system of that computer. It is more or less like one creating using C or C++. Java's ability to create Applets makes it important. An Applet is an application designed to be transmitted over the Internet and executed by a Java –compatible web browser. An applet is actually a tiny Java program, dynamically downloaded across the network, just like an image. But the difference is, it is an intelligent program, not just a media file. It can react to the user input and dynamically change.

Features of Java Security:

Every time you that you download a "normal" program, you are risking a viral infection. Prior to Java, most users did not download executable programs frequently, and those who did scan them for viruses prior to execution. Most users still worried about the possibility of infecting their systems with a virus. In addition, another type of malicious program exists that must be guarded against. This type of program can gather private information, such as credit card numbers, bank account balances, and passwords. Java answers both these concerns by providing a "firewall" between a network application and your computer.

When you use a Java-compatible Web browser, you can safely download Java applets without fear of virus infection or malicious intent.

Portability:

For programs to be dynamically downloaded to all the various types of platforms connected to the Internet, some means of generating portable executable code is needed .As you will see, the same mechanism that helps ensure security also helps create portability. Indeed, Java's solution to these two problems is both elegant and efficient.

The Byte code:

The key that allows the Java to solve the security and portability problems is that the output of Java compiler is Byte code. Byte code is a highly optimized set of instructions designed to be executed by the Java run-time system, which is called the Java Virtual Machine (JVM). That is, in its standard form, the JVM is an interpreter for byte code.

Translating a Java program into byte code helps makes it much easier to run a program in a wide variety of environments. The reason is, once the run-time package exists for a given system, any Java program can run on it.

Although Java was designed for interpretation, there is technically nothing about Java that prevents on-the-fly compilation of byte code into native code. Sun has just completed its Just in Time (JIT) compiler for byte code. When the JIT compiler is a part of JVM, it compiles byte code into executable code in real time, on a piece-by-piece, demand basis. It is not possible to compile an entire Java program into executable code all at once, because Java performs various run-time checks that can be done only at run time. The JIT compiles code, as it is needed, during execution.

Java Virtual Machine (JVM):

Beyond the language, there is the Java virtual machine. The Java virtual machine is an important element of the Java technology. The virtual machine can be embedded within a web browser or an operating system. Once a piece of Java code is loaded onto a machine, it is verified. As part of the loading process, a class loader is invoked and does byte code verification makes sure that the code that's has been generated by the compiler will not corrupt the machine that it's loaded on. Byte code verification takes place at the end of the compilation process to make sure that is all accurate and correct. So byte code verification is integral to the compiling and executing of Java code.

Overall Description:

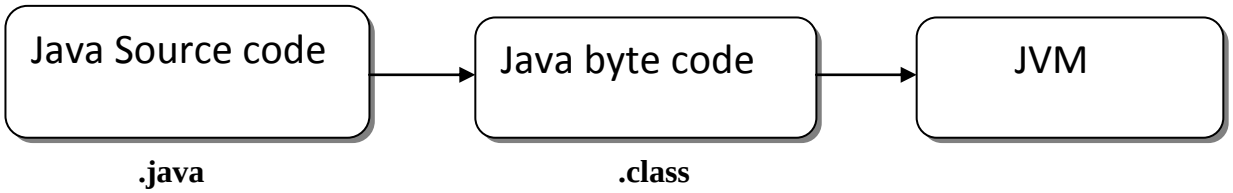


Fig-28 Development process of Java program

Java programming uses to produce byte codes and executes them. The first box indicates that the Java source code is located in a. Java file that is processed with a Java compiler called javac. The Java compiler produces a file called a. class file, which contains the byte code. The .Class file is then loaded across the network or loaded locally on your machine into the execution environment is the Java virtual machine, which interprets and executes the byte code.

Java Architecture :

Java architecture provides a portable, robust, high performing environment for development. Java provides portability by compiling the byte codes for the Java Virtual Machine, which is then interpreted on each platform by the run-time environment. Java is a dynamic system, able to load code when needed from a machine in the same room or across the planet.

Compilation of code:

When you compile the code, the Java compiler creates machine code (called byte code) for a hypothetical machine called Java Virtual Machine (JVM). The JVM is supposed to execute the byte code. The JVM is created for overcoming the issue of portability. The code is written and compiled for one machine and interpreted on all machines. This machine is called Java Virtual Machine.

Compiling and interpreting Java Source Code:

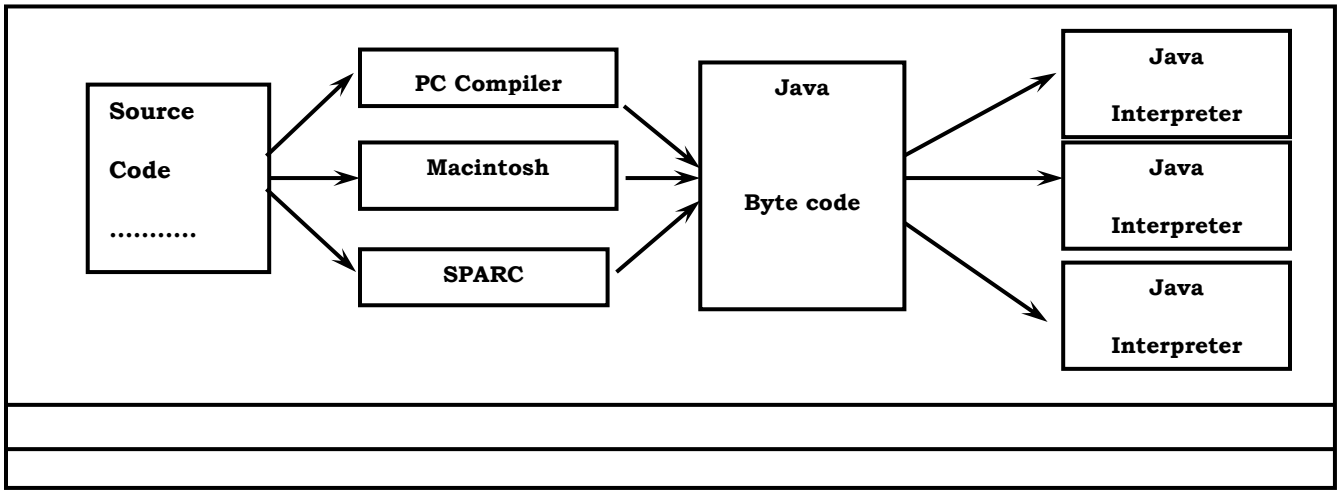


Fig-29 Compiling and interpreting Java Source code

During run-time the Java interpreter tricks the byte code file into thinking that it is running on a Java Virtual Machine. In reality this could be a Intel Pentium Windows 95 or Sun

SARC station running Solaris or Apple Macintosh running system and all could receive code from any computer through Internet and run the Applets.

Simple Java was designed to be easy for the Professional programmer to learn and to use effectively. If you are an experienced C++ programmer, learning Java will be even easier. Because Java inherits the C/C++ syntax and many of the objects oriented features of C++. Most of the confusing concepts from C++ are either left out of Java or implemented in a cleaner, more approachable manner. In Java there are a small number of clearly defined ways to accomplish a given task.

Object-Oriented:

Java was not designed to be source-code compatible with any other language. This allowed the Java team the freedom to design with a blank slate. One outcome of this was a clean usable, pragmatic approach to objects. The object model in Java is simple and easy to extend, while simple types, such as integers, are kept as high-performance non-objects.

Robust:

The multi-platform environment of the Web places extraordinary demands on a program, because the program must execute reliably in a variety of systems. The ability to create robust programs was given a high priority in the design of Java. Java is strictly typed language; it checks your code at compile time and run time.

Java virtually eliminates the problems of memory management and de-allocation, which is completely automatic. In a well-written Java program, all run time errors can –and should –be managed by your program.

Java Database Connectivity:

What Is JDBC?

JDBC is a Java API for executing SQL statements. (As a point of interest, JDBC is a trademarked name and is not an acronym; nevertheless, JDBC is often thought of as standing for Java Database Connectivity. It consists of a set of classes and interfaces written in the Java

programming language. JDBC provides a standard API for tool/database developers and makes it possible to write database applications using a pure Java API.

Using JDBC, it is easy to send SQL statements to virtually any relational database. One can write a single program using the JDBC API, and the program will be able to send SQL statements to the appropriate database. The combinations of Java and JDBC lets a programmer write it once and run it anywhere.

What Does JDBC Do?

Simply put, JDBC makes it possible to do three things:

- Establish a connection with a database
- Send SQL statements
- Process the results.

JDBC versus ODBC and other APIs:

At this point, Microsoft's ODBC (Open Database Connectivity) API is that probably the most widely used programming interface for accessing relational databases. It offers the ability to connect to almost all databases on almost all platforms.

So why not just use ODBC from Java? The answer is that you can use ODBC from Java, but this is best done with the help of JDBC in the form of the JDBC-ODBC Bridge, which we will cover shortly. The question now becomes "Why do you need JDBC?" There are several answers to this question:

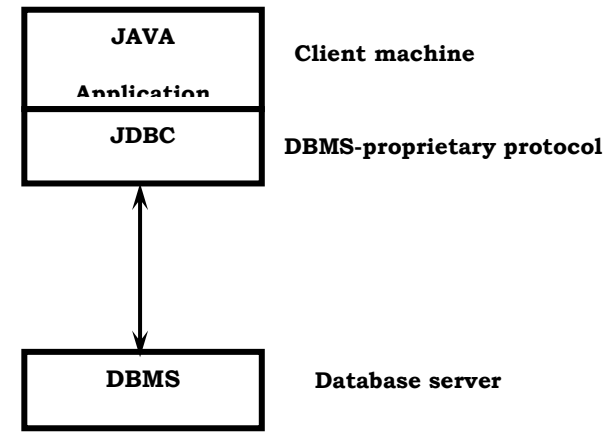
1. ODBC is not appropriate for direct use from Java because it uses a C interface. Calls from Java to native C code have a number of drawbacks in the security, implementation, robustness, and automatic portability of applications.
2. A literal translation of the ODBC C API into a Java API would not be desirable. For example, Java has no pointers, and ODBC makes copious use of them, including the notoriously error-prone generic pointer "void *". You can think of JDBC as ODBC translated into an object-oriented interface that is natural for Java programmers.
3. ODBC is hard to learn. It mixes simple and advanced features together, and it has complex options even for simple queries. JDBC, on the other hand, was designed to keep simple things simple while allowing more advanced capabilities where required.

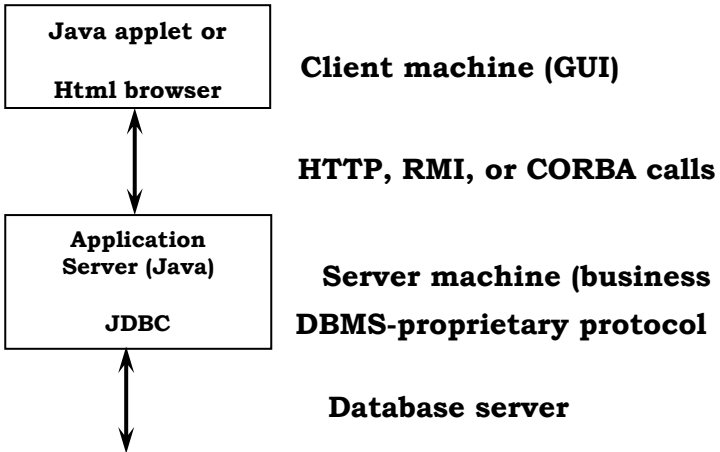
4. A Java API like JDBC is needed in order to enable a "pure Java" solution. When ODBC is used, the ODBC driver manager and drivers must be manually installed on every client machine. When the JDBC driver is written completely in Java, however, JDBC code is automatically installable, portable, and secure on all Java platforms from network computers to mainframes.

Two-tier and three-tier Models:

The JDBC API supports both two-tier and three-tier models for database access.

In the two-tier model, a Java applet or application talks directly to the database. This requires a JDBC driver that can communicate with the particular database management system being accessed. A user's SQL statements are delivered to the database, and the results of those statements are sent back to the user. The database may be located on another machine to which the user is connected via a network. This is referred to as a client/server configuration, with the user's machine as the client, and the machine housing the database as the server. The network can be an Intranet, which, for example, connects employees within a corporation, or it can be the Internet.





In the three-tier model, commands are sent to a "middle tier" of services, which then send SQL statements to the database. The database processes the SQL statements and sends the results back to the middle tier, which then sends them to the user. MIS directors find the three-tier model very attractive because the middle tier makes it possible to maintain control over access and the kinds of updates that can be made to corporate data. Another advantage is that when there is a middle tier, the user can employ an easy-to-use higher-level API which is translated by the middle tier into the appropriate low-level calls. Finally, in many cases the three-tier architecture can provide performance advantages.

Until now the middle tier has typically been written in languages such as C or C++, which offer fast performance.

However, with the introduction of optimizing compilers that translate Java byte code into efficient machine-specific code, it is becoming practical to implement the middle tier in Java. This is a big plus, making it possible to take advantage of Java's robustness, multithreading, and security features. JDBC is important to allow database access from a Java middle tier.

JDBC Driver Types:

The JDBC drivers that we are aware of at this time fit into one of four categories:

- JDBC-ODBC bridge plus ODBC driver
- Native-API partly-Java driver
- JDBC-Net pure Java driver
- Native-protocol pure Java driver

JDBC-ODBC Bridge:

If possible, use a Pure Java JDBC driver instead of the Bridge and an ODBC driver. This completely eliminates the client configuration required by ODBC. It also eliminates the potential that the Java VM could be corrupted by an error in the native code brought in by the Bridge (that is, the Bridge native library, the ODBC driver manager library, the ODBC driver library, and the database client library).

What Is the JDBC- ODBC Bridge?

The JDBC-ODBC Bridge is a JDBC driver, which implements JDBC operations by translating them into ODBC operations. To ODBC it appears as a normal application program. The Bridge implements JDBC for any database for which an ODBC driver is available. The Bridge is implemented as the

Sun.jdbc.odbc Java package and contains a native library used to access ODBC. The Bridge is a joint development of Innersole and Java Soft.

JDBC connectivity:

The JDBC provides database-independent connectivity between the J2EE platform and a wide range of tabular data sources. JDBC technology allows an Application Component Provider to:

- Perform connection and authentication to a database server
- Manager transactions
- Move SQL statements to a database engine for preprocessing and execution
- Execute stored procedures
- Inspect and modify the results from Select statements

Database:

A database management system (DBMS) is computer software designed for the purpose of managing databases, a large set of structured data, and run operations on the data requested by numerous users. Typical examples of DBMSs include Oracle, DB2, Microsoft Access, Microsoft SQL Server, Firebird, PostgreSQL, MySQL, SQLite, FileMaker and Sybase Adaptive Server Enterprise. DBMSs are typically used by Database administrators in the creation of Database systems. Typical examples of DBMS use include accounting, human resources and customer support systems.

Originally found only in large companies with the computer hardware needed to support large data sets, DBMSs have more recently emerged as a fairly standard part of any company back office.

Description:

A DBMS is a complex set of software programs that controls the organization, storage, management, and retrieval of data in a database. A DBMS includes:

✓ A modeling language to define the schema of each database hosted in the DBMS, according to the DBMS data model.

- The four most common types of organizations are the hierarchical, network, relational and object models. Inverted lists and other methods are also used. A given database management system may provide one or more of the four models. The optimal structure depends on the natural organization of the application's data, and on the application's requirements (which include transaction rate (speed), reliability, maintainability, scalability, and cost).
- The dominant model in use today is the ad hoc one embedded in SQL, despite the objections of purists who believe this model is a corruption of the relational model, since it violates several of its fundamental principles for the sake of practicality and performance. Many DBMSs also support the Open Database Connectivity API that supports a standard way for programmers to access the DBMS.

✓ Data structures (fields, records, files and objects) optimized to deal with very large amounts of data stored on a permanent data storage device (which implies relatively slow access compared to volatile main memory).

✓ A database query language and report writer to allow users to interactively interrogate the database, analyze its data and update it according to the users privileges on data.

- It also controls the security of the database.
- Data security prevents unauthorized users from viewing or updating the database. Using passwords, users are allowed access to the entire database or subsets of it called subschemas. For example, an employee database can contain all the data about an individual employee, but one group of users may be authorized to view only payroll data, while others are allowed access to only work history and medical data.
- If the DBMS provides a way to interactively enter and update the database, as well as interrogate it, this capability allows for managing personal databases. However, it may not leave an audit trail of actions or provide the kinds of controls necessary in a multi-user organization. These controls are only available when a set of application programs are customized for each data entry and updating function.

✓ A transaction mechanism, that ideally would guarantee the ACID properties, in order to ensure data integrity, despite concurrent user accesses (concurrency control), and faults (fault tolerance).

- It also maintains the integrity of the data in the database.
- The DBMS can maintain the integrity of the database by not allowing more than one user to update the same record at the same time. The DBMS can help prevent duplicate records via unique index constraints; for example, no two customers with the same customer numbers (key fields) can be entered into the database. See ACID properties for more information (Redundancy avoidance).

The DBMS accepts requests for data from the application program and instructs the operating system to transfer the appropriate data.

When a DBMS is used, information systems can be changed much more easily as the organization's information requirements change. New categories of data can be added to the database without disruption to the existing system.

Organizations may use one kind of DBMS for daily transaction processing and then move the detail onto another computer that uses another DBMS better suited for random inquiries and analysis. Overall systems design decisions are performed by data administrators and systems analysts. Detailed database design is performed by database administrators.

Database servers are specially designed computers that hold the actual databases and run only the DBMS and related software. Database servers are usually multiprocessor computers, with RAID disk arrays used for stable storage. Connected to one or more servers via a high-speed channel, hardware database accelerators are also used in large volume transaction processing environments.

DBMSs are found at the heart of most database applications. Sometimes DBMSs are built around a private multitasking kernel with built-in networking support although nowadays these functions are left to the operating system.

DATABASE ARCHITECTURE:

SQL:

Structured Query Language (SQL) is the language used to manipulate relational databases. SQL is tied very closely with the relational model.

In the relational model, data is stored in structures called relations or tables.

SQL statements are issued for the purpose of:

Data definition: Defining tables and structures in the database (DDL used to create, alter and drop schema objects such as tables and indexes).

Data manipulation: Used to manipulate the data within those schema objects (DML Inserting, Updating, Deleting the data, and Querying the Database).

A schema is a collection of database objects that can include: tables, views, indexes and sequences

List of SQL statements that can be issued against an Oracle database schema are:

- **ALTER** - Change an existing table, view or index definition (DDL)
- **AUDIT** - Track the changes made to a table (DDL)
- **COMMENT** - Add a comment to a table or column in a table (DDL)
- **COMMIT** - Make all recent changes permanent (DML - transactional)
- **CREATE** - Create new database objects such as tables or views (DDL)
- **DELETE** - Delete rows from a database table (DML)
- **DROP** - Drop a database object such as a table, view or index (DDL)
- **GRANT** - Allow another user to access database objects such as tables or views (DDL)
- **INSERT** - Insert new data into a database table (DML)
- **No AUDIT** - Turn off the auditing function (DDL)
- **REVOKE** - Disallow a user access to database objects such as tables and views (DDL)
- **ROLLBACK** - Undo any recent changes to the database (DML - Transactional)
- **SELECT** - Retrieve data from a database table (DML)
- **TRUNCATE** - Delete all rows from a database table (can not be rolled back) (DML)
- **UPDATE** - Change the values of some data items in a database table (DML)

SERVLETS:

Introduction

The Java web server is Java Soft's own web Server. The Java web server is just a part of a larger framework, intended to provide you not just with a web server, but also with tools. To build customized network servers for any Internet or Intranet client/server system. Servlets are to a web server, how applets are to the browser.

There are many features of Servlets that make them easy and attractive to use. These include:

- Easily configured using the GUI-based Admin tool
- Can be loaded and invoked from a local disk or remotely across the network.
- Can be linked together, or chained, so that one Servlets can call another Servlets or several Servlets in sequence.
- Can be called dynamically from within HTML pages, using server-side include tags.
- Are secure - even when downloading across the network, the Servlets security model and Servlets sandbox protect your system from unfriendly behavior.

Advantages of the Servlet API:

One of the great advantages of the Servlet API is protocol independence. It assumes nothing about:

- The protocol being used to transmit on the net
- How it is loaded
- The server environment it will be running in

These qualities are important, because it allows the Servlet API to be embedded in many different kinds of servers. There are other advantages to the Servlet API as well. These include:

- It's extensible - you can inherit all your functionality from the base classes made available to you.
- It's simple, small, and easy to use.

Features of Servlets:

- Servlets are persistent. Servlet are loaded only by the web server and can maintain services between requests.
- Servlets are fast. Since Servlets only need to be loaded once, they offer much better performance over their CGI counterparts.
- Servlets are platform independent.
- Servlets are extensible. Java is a robust, object-oriented programming language, which easily can be extended to suit your needs
- Servlets are secure.
- Servlets can be used with a variety of clients.

Java Server Pages (JSP):

Java server Pages is a simple, yet powerful technology for creating and maintaining dynamic-content web pages. Based on the Java programming language, Java Server Pages offers proven portability, open standards, and mature re-usable component model .The Java Server Pages architecture enables the separation of content generation from content presentation. This separation not eases maintenance headaches; it also allows web team members to focus on their areas of expertise. Now, web page designer can concentrate on layout, and web application designers on programming, with minimal concern about impacting each other's work.

CHAPTER-5:

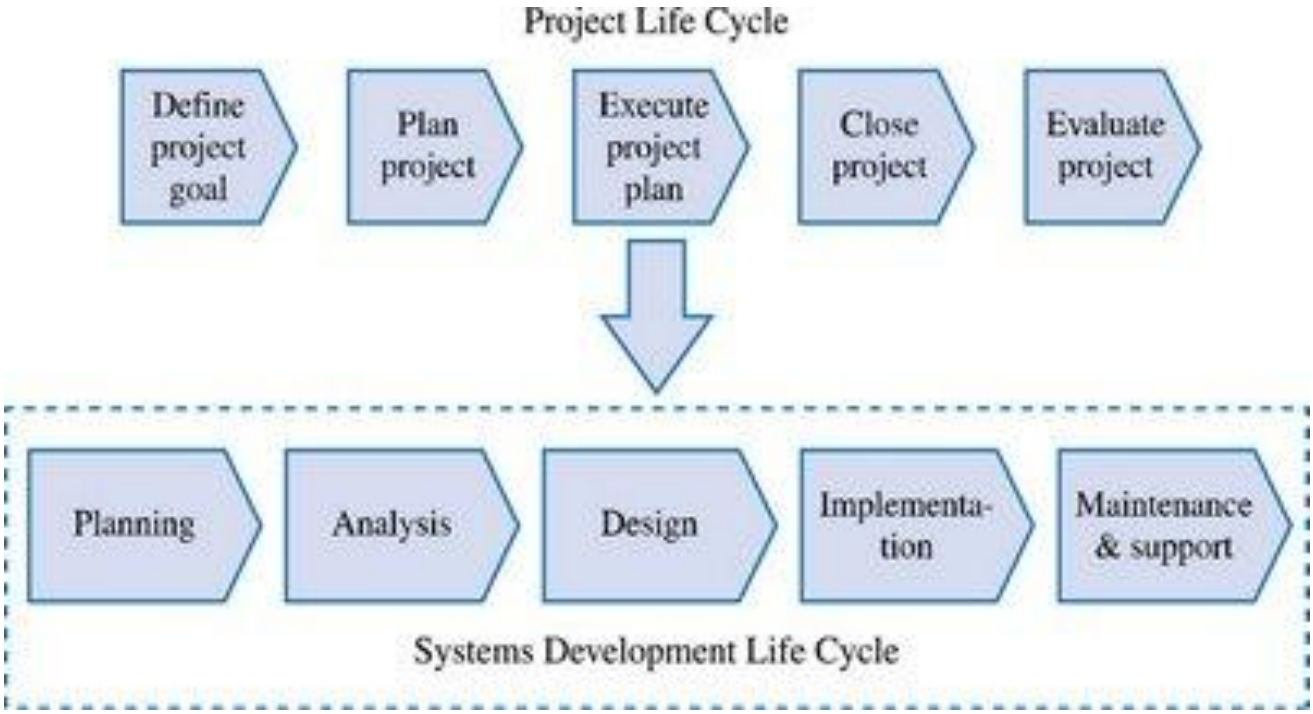
SYSTEM DESIGN AND DEVELOPMENT

SOFTWARE MODEL OR ARCHITECTURE ANALYSIS:

System Development Life Cycle:

The **Systems Development Life Cycle (SDLC)**, or *Software Development Life Cycle* in systems engineering, information systems and software engineering, is the process of creating or altering systems, and the models and methodologies that people use to develop these systems.

In software engineering the SDLC concept underpins many kinds of software development methodologies. These methodologies form the framework for planning and controlling the creation of an information system the software development process.

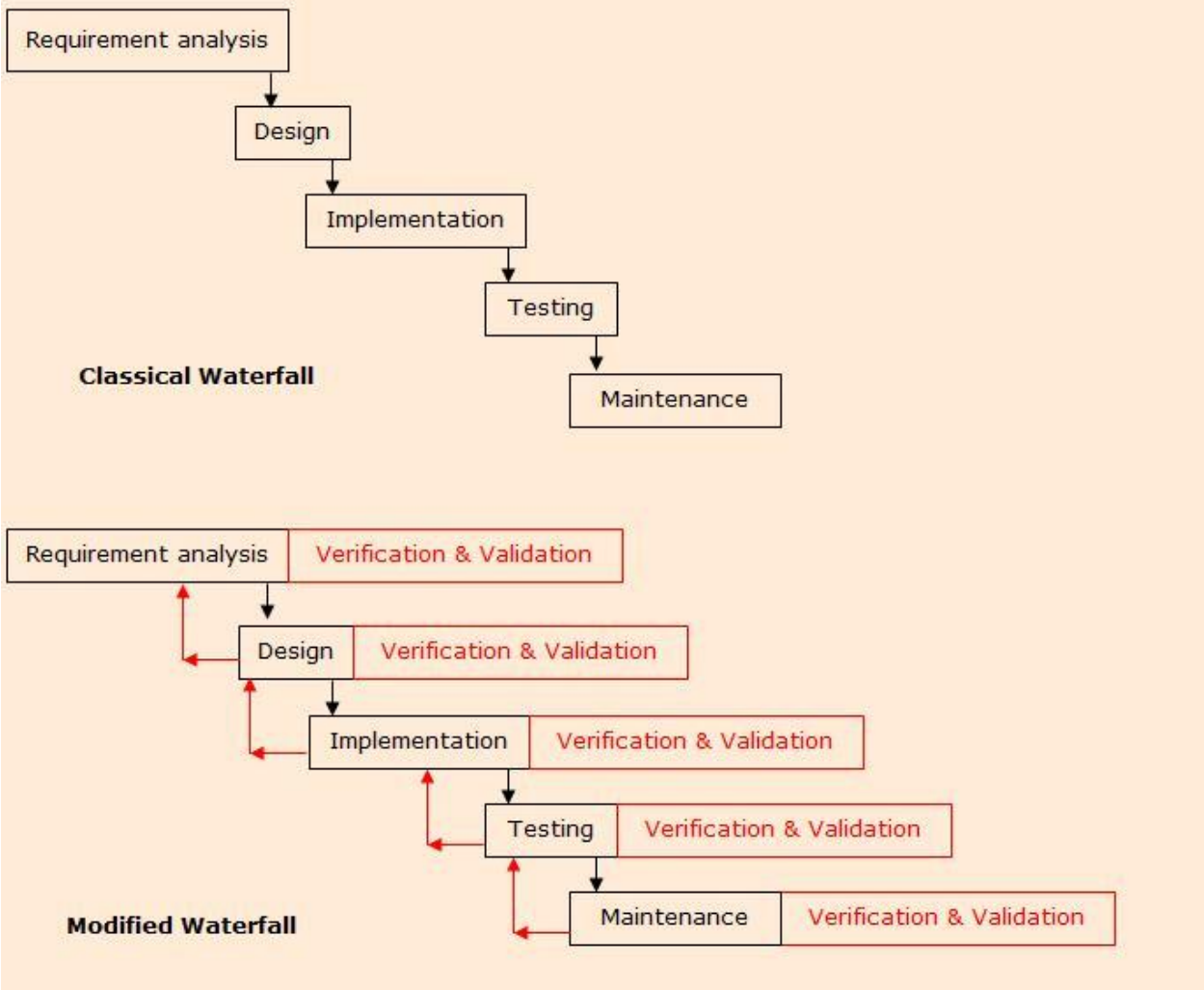


Systems development methodology

A Systems Development Life Cycle (SDLC) adheres to important phase that are essential for developers, such as planning, analysis, design, and implementation.

Waterfall model

The waterfall model is a sequential software development process, in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of Conception, Initiation, Design(validation), Construction , Testing and Maintenance.



- **Project planning, feasibility study**: Establishes a high-level view of the intended project and determines its goals.

- **Systems analysis, requirements definition**: Refines project goals into defined functions and operation of the intended application. Analyzes end-user information needs.

- **Systems design**: Describes desired features and operations in detail, including screen layouts, business rules, process diagrams, pseudocode and other documentation.

- **Implementation**: The real code is written here.

- **Integration and testing**: Brings all the pieces together into a special testing environment, then checks for errors, bugs and interoperability.

- **Acceptance, installation, deployment**: The final stage of initial development, where the software is put into production and runs actual business.

- **Maintenance**: What happens during the rest of the software's life: changes, correction, additions, moves to a different computing platform and more. This, the least glamorous and perhaps most important step of all, goes on seemingly forever.

In the following example (see picture) these stage of the Systems Development Life Cycle are divided in ten steps from definition to creation and modification of IT work products:

The tenth phase occurs when the system is disposed of and the task performed is either eliminated or transferred to other systems. The tasks and work products for each phase are described in subsequent chapters. [6]

System analysis:

The goal of system analysis is to determine where the problem is in an attempt to fix the system. This step involves breaking down the system in different pieces to analyze the situation, analyzing project goals, breaking down what needs to be created and attempting to engage users so that definite requirements can be defined. Requirements analysis sometimes requires individuals/teams from client as well as service provider sides to get detailed and accurate requirements....often there has to be a lot of communication to and from to understand these

requirements. Requirement gathering is the most crucial aspect as many times communication gaps arise in this phase and this leads to validation errors and bugs in the software program.

Design:

In systems design the design functions and operations are described in detail, including screen layouts, business rules, process diagrams and other documentation. The output of this stage will describe the new system as a collection of modules or subsystems.

The design stage takes as its initial input the requirements identified in the approved requirements document. For each requirement, a set of one or more design elements will be produced as a result of interviews, workshops, and/or prototype efforts.

Design elements describe the desired software features in detail, and generally include functional hierarchy diagrams, screen layout diagrams, tables of business rules, business process diagrams, pseudocode, and a complete entity-relationship diagram with a full data dictionary. These design elements are intended to describe the software in sufficient detail that skilled programmers may develop the software with minimal additional input design.

Implementation

Modular and subsystem programming code will be accomplished during this stage. Unit testing and module testing are done in this stage by the developers. This stage is intermingled with the next in that individual modules will need testing before integration to the main project.

Testing

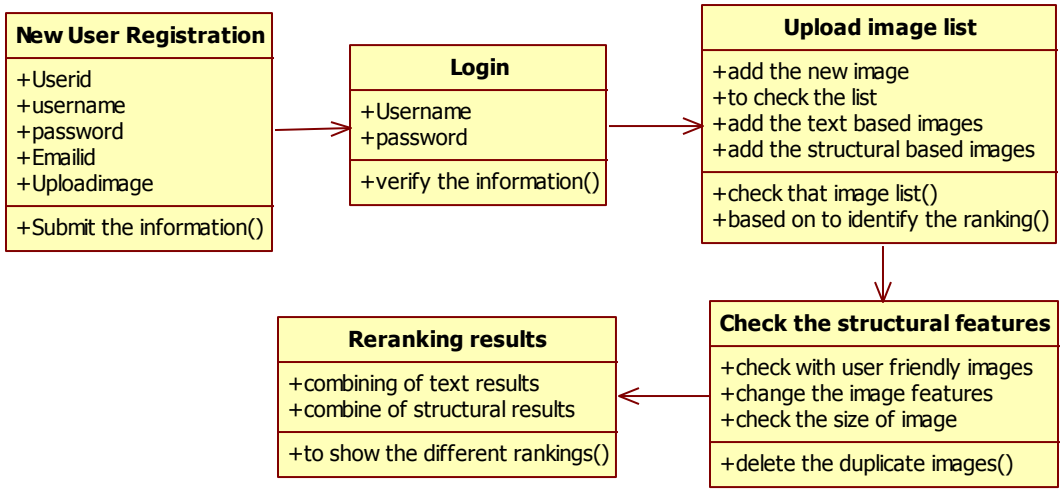
The code is tested at various levels in software testing. Unit, system and user acceptance testings are often performed. This is a grey area as many different opinions exist as to what the stages of testing are and how much if any iteration occurs.

Operations and maintenance: The deployment of the system includes changes and enhancements before the decommissioning or sunset of the system. Maintaining the system is an important aspect of SDLC. As key personnel change positions in the organization, new changes will be implemented, which will require system .

UML DIAGRAMS

Class Diagram:

Class diagrams are widely used to describe the types of objects in a system and their relationships. Class diagrams model class structure and contents using design elements such as classes, packages and objects. Class diagrams describe three different perspectives when designing a system, conceptual, specification, and implementation. These perspectives become evident as the diagram is created and help solidify the design.



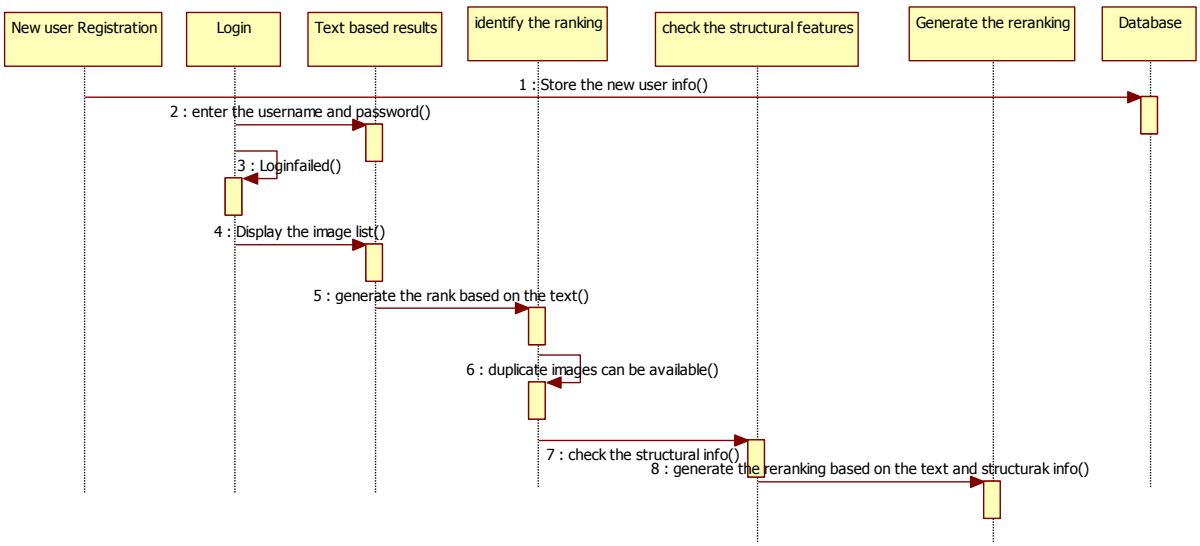
Sequence Diagram:

Sequence diagrams belong to a group of UML diagrams called Interaction Diagrams. Sequence diagrams describe how objects interact over the course of time through an exchange of messages. A single sequence diagram often represents the flow of events for a single use case.

Instance: An instance of a class shows a sample configuration of an object. On the sequence diagram, each instance has a lifeline box underneath it showing it's existence over a period of time.

Actor: An actor is anything outside the system that interacts with the system. It could be a user or another system.

Message: The message indicates communication between objects. The order of messages from top to bottom on your diagram should be the order in which the messages occur.



Collaboration Diagram:

Collaboration diagrams belong to a group of UML diagrams called Interaction Diagrams . Collaboration diagrams, like Sequence Diagrams, show how objects interact over the course of time. However, instead of showing the sequence of events by the layout on the diagram, collaboration diagrams show the sequence by numbering the messages on the diagram. This makes it easier to show how the objects are linked together, but harder to see the sequence at a glance.

Instance:

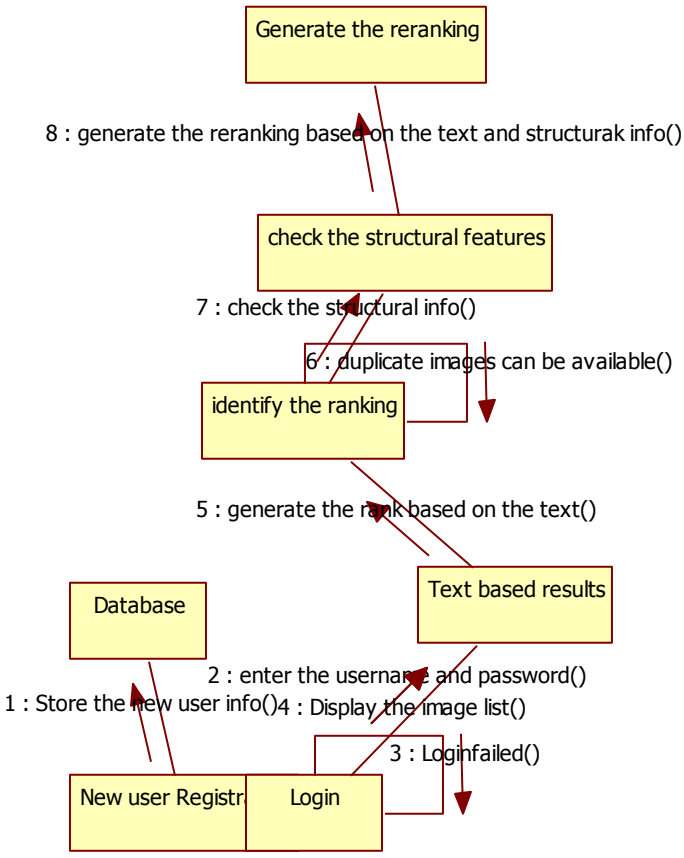
An **instance** of a class shows a sample configuration of an object. On the sequence diagram, each instance has a **lifeline** box underneath it showing it's existence over a period of time.

Lollipop Interface:

A **lollipop interface** is a shorthand syntax for a interface. It shows the interface name without displaying the operations.

Message:

The message indicates communication between objects. The order of messages from top to bottom on your diagram should be the order in which the messages occur.



Use Case Diagram:

A **use case diagram** in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals , and any dependencies between those use cases.

Actor

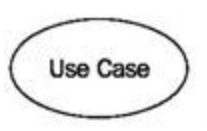
You can picture an actor as a user of the IT system, for example Mr. Steel or Mrs. Smith from check-in. Because individual persons are irrelevant for the model, they are abstracted. So the actors are called “check-in employee” or “passenger”:

Actors represent roles that users take on when they use the IT system, e.g., the role of a check-in employee. One person can act in more than one role toward the IT system. It is important for the IT system in which role a person is acting. Therefore, it is necessary to log on to many IT systems in a certain role, for instance, as a normal user or as an administrator. In each case access to the appropriate functionalities (use cases) is granted.

Actors themselves are not part of the IT system. However, as employees they can be part of the business system (see Figure 4.5).

Use Case

Use cases describe the interactions that take place between actors and IT systems during the execution of business processes:



A use case represents a part of the functionality of the IT system and enables the user (modeled as an actor) to access this functionality.

Anything that users would like to do with the IT system has to be made available as a use case (or part of a use case). Functionalities that exist in the IT system, but that are not accessed by means of use cases, are *not* available to users.

Even though the idea behind use cases is to describe interactions, flows of batch processing, which generally do not include interactions, can also be described as use cases. The actor of such a batch use case is then the one who initiates batch processing. For instance, *generating check-in statistics* would be a batch use case.

Relationships:

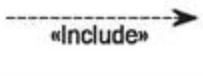
Association

An association is a connection between an actor and a use case. An association indicates that an actor can carry out a use case. Several actors at one use case mean that each actor can carry out the use case on his or her own and not that the actors carry out the use case together:

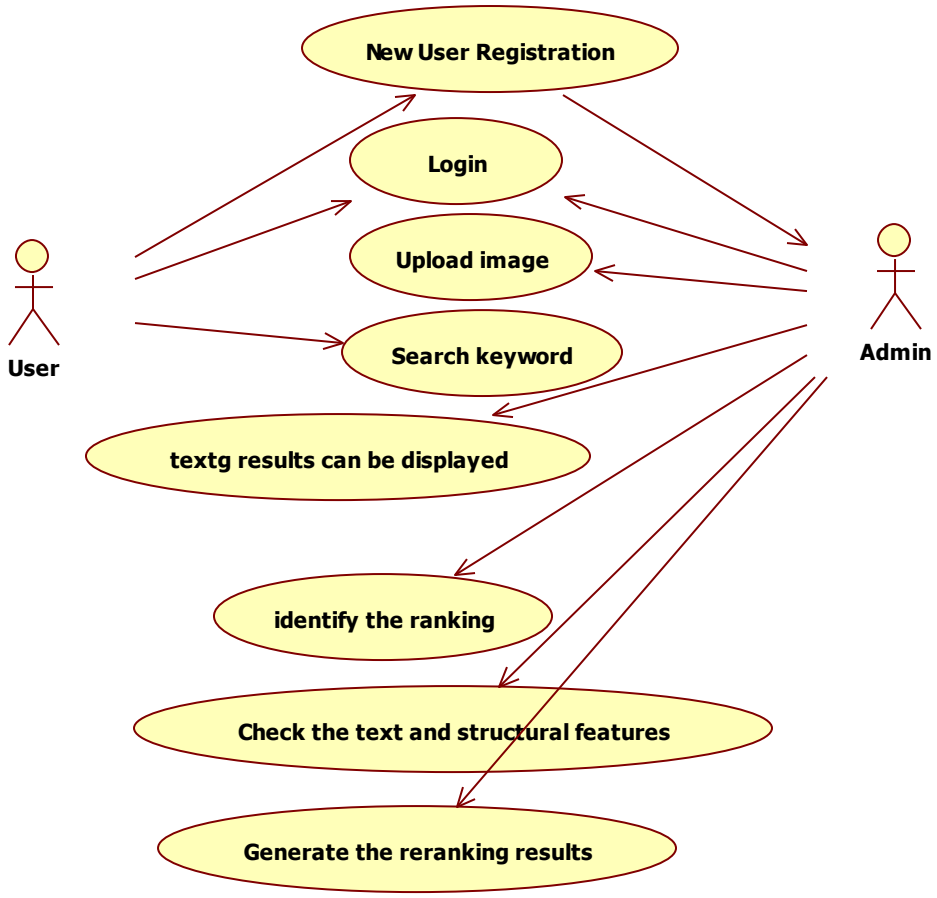
According to UML, association only means that an actor is involved in a use case. We use associations in a restricted manner.

Include Relationships

An include relationship is a relationship between two use cases:

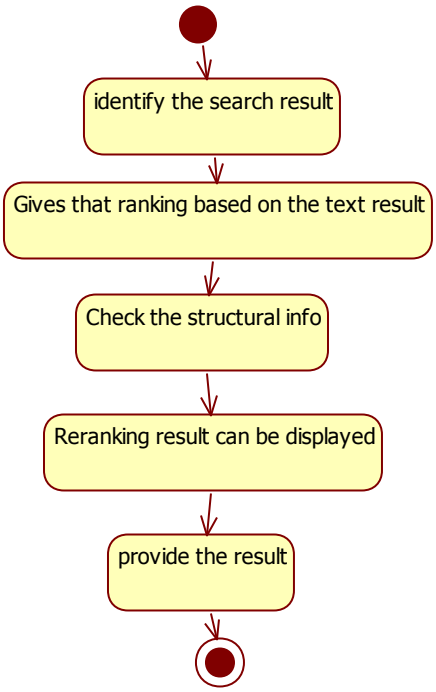


It indicates that the use case to which the arrow points is included in the use case on the other side of the arrow. This makes it possible to reuse a use case in another use case. Figure 4.9 shows an example of this relationship. In the flow of the use case, express check-in is a point at which the use case generating boarding pass is included. This means that at this point the entire process generating boarding pass is carried out.



STATE CHART DIAGRAM:

1. Member of the Behavioral group
2. Graph of states and transitions
3. Showing the response of an object to external stimuli
4. Attached to a class or a method

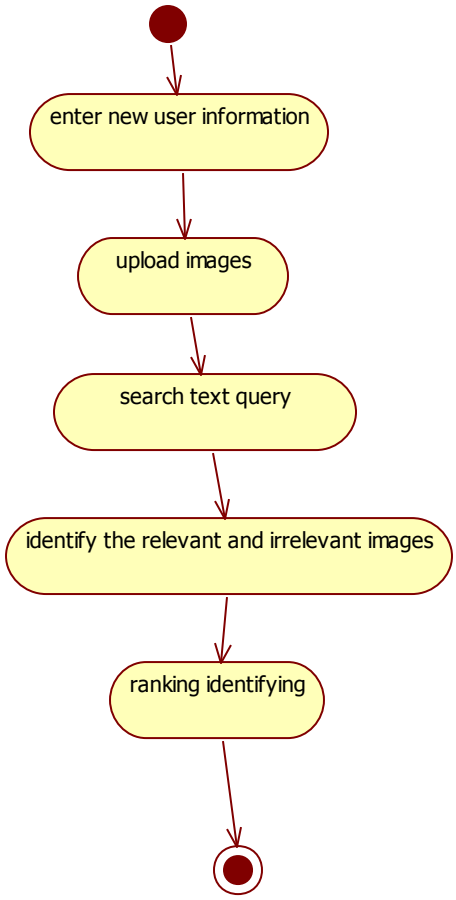


ACTIVITY DIAGRAM:

Activity diagrams describe the workflow behavior of a system. Activity diagrams are similar to state diagrams because activities are the state of doing something. The diagrams describe the state of activities by showing the sequence of activities performed. Activity diagrams can show activities that are conditional or parallel.

Activity diagrams should be used in conjunction with other modeling techniques such as interaction diagrams and state diagrams. The main reason to use activity diagrams is to model the workflow behind the system being designed. Activity Diagrams are also useful for: analyzing a use case by describing what actions need to take place and when they should

occur; describing a complicated sequential algorithm; and modeling applications with parallel processes.

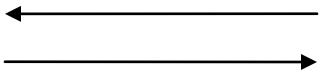


Data Flow Diagrams:

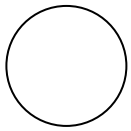
A graphical tool used to describe and analyze the moment of data through a system manual or automated including the process, stores of data, and delays in the system. Data Flow Diagrams are the central tool and the basis from which other components are developed. The transformation of data from input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

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

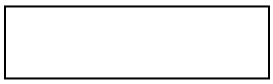
1. Dataflow: Data move in a specific direction from an origin to a destination.



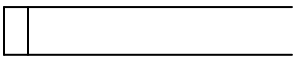
2. Process: People, procedures, or devices that use or produce (Transform) Data. The physical component is not identified.



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

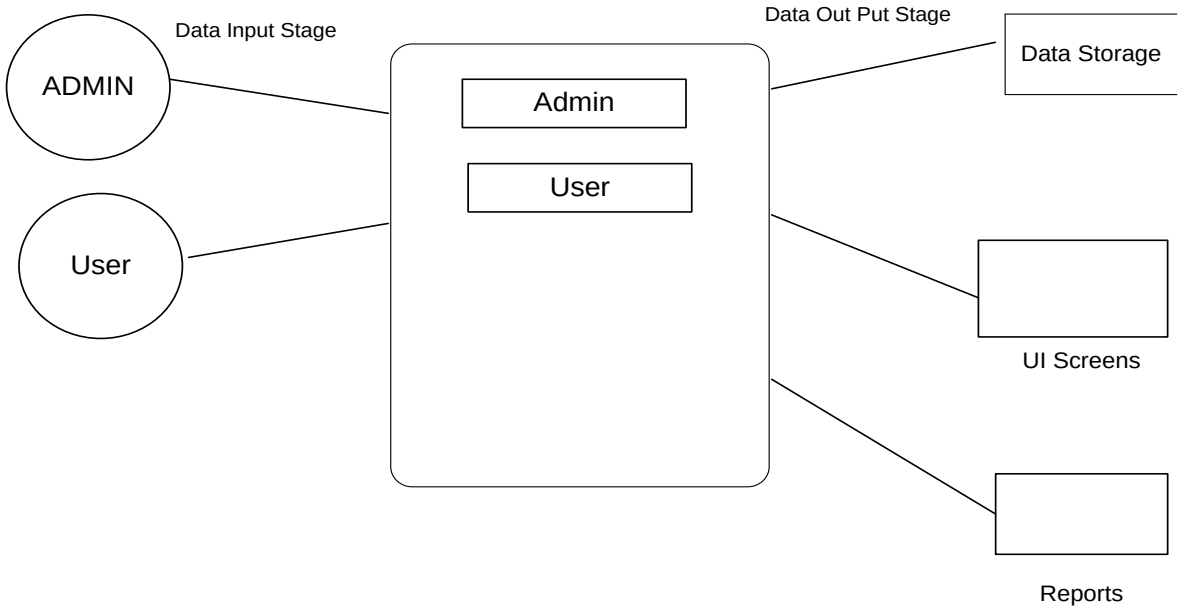


4. Data Store: Here data are stored or referenced by a process in the System.

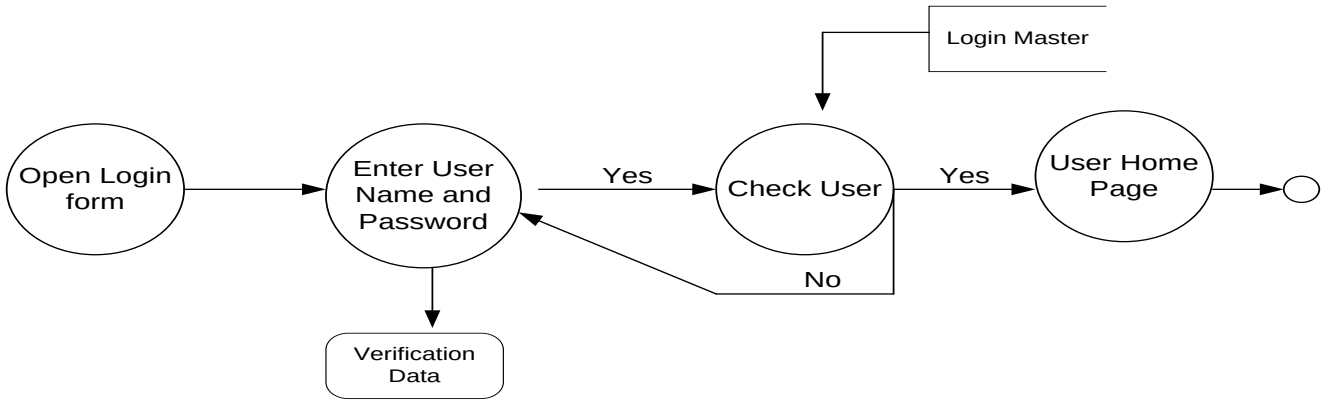


CONTEXT LEVEL DIAGRAM

Context Level DFD



Login DFD



E-R DIAGRAMS:

In software engineering, an **entity-relationship model** (**ERM**) is an abstract and conceptual representation of data. Entity-relationship modeling is a database modeling method, used to produce a type of conceptual schema or semantic data model of a system, often a relational database, and its requirements in a top-down fashion. Diagrams created by this process are called **entity-relationship diagrams**, **ER diagrams**, or **ERDs**. The definitive reference for entity-relationship modeling is Peter Chen's 1976 paper. However, variants of the idea existed previously, and have been devised subsequently.

An entity may be defined as a thing which is recognized as being capable of an independent existence and which can be uniquely identified. An entity is an abstraction from the complexities of some domain. When we speak of an entity we normally speak of some aspect of the real world which can be distinguished from other aspects of the real world.[3]

An entity may be a physical object such as a house or a car, an event such as a house sale or a car service, or a concept such as a customer transaction or order. Although the term entity is the one most commonly used, following Chen we should really distinguish between an entity and an entity-type. An entity-type is a category. An entity, strictly speaking, is an instance of a given entity-type. There are usually many instances of an entity-type. Because the term entity-type is somewhat cumbersome, most people tend to use the term entity as a synonym for this term.

Entities can be thought of as nouns. Examples: a computer, an employee, a song, a mathematical theorem.

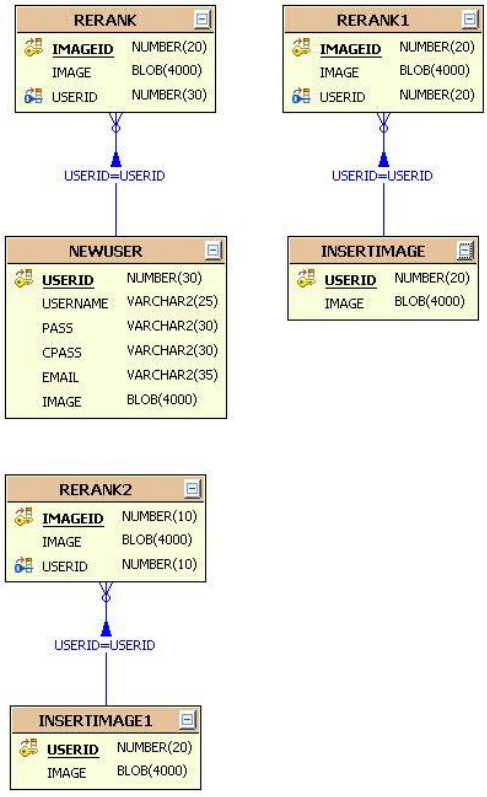
A relationship captures how two or more entities are related to one another. Relationships can be thought of as verbs, linking two or more nouns. Examples: an *owns* relationship between a company and a computer, a *supervises* relationship between an employee and a department, a *performs* relationship between an artist and a song, a *proved* relationship between a mathematician and a theorem.

The model's linguistic aspect described above is utilized in the declarative database query language ERROL, which mimics natural language constructs.

Entities and relationships can both have attributes. Examples: an *employee* entity might have a *Social Security Number* (SSN) attribute; the *proved* relationship may have a *date* attribute.

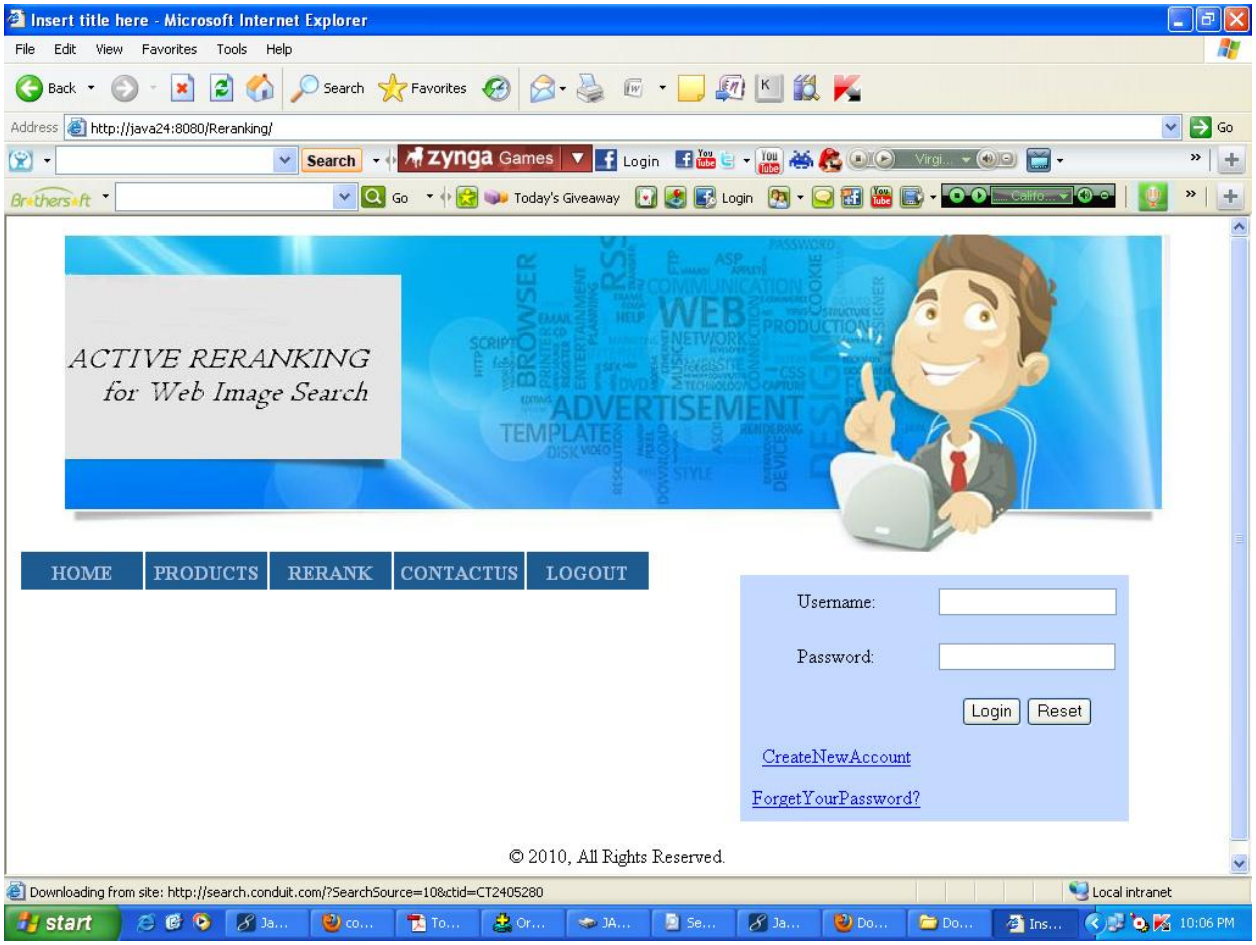
Every entity (unless it is a weak entity) must have a minimal set of uniquely identifying attributes, which is called the entity's primary key.

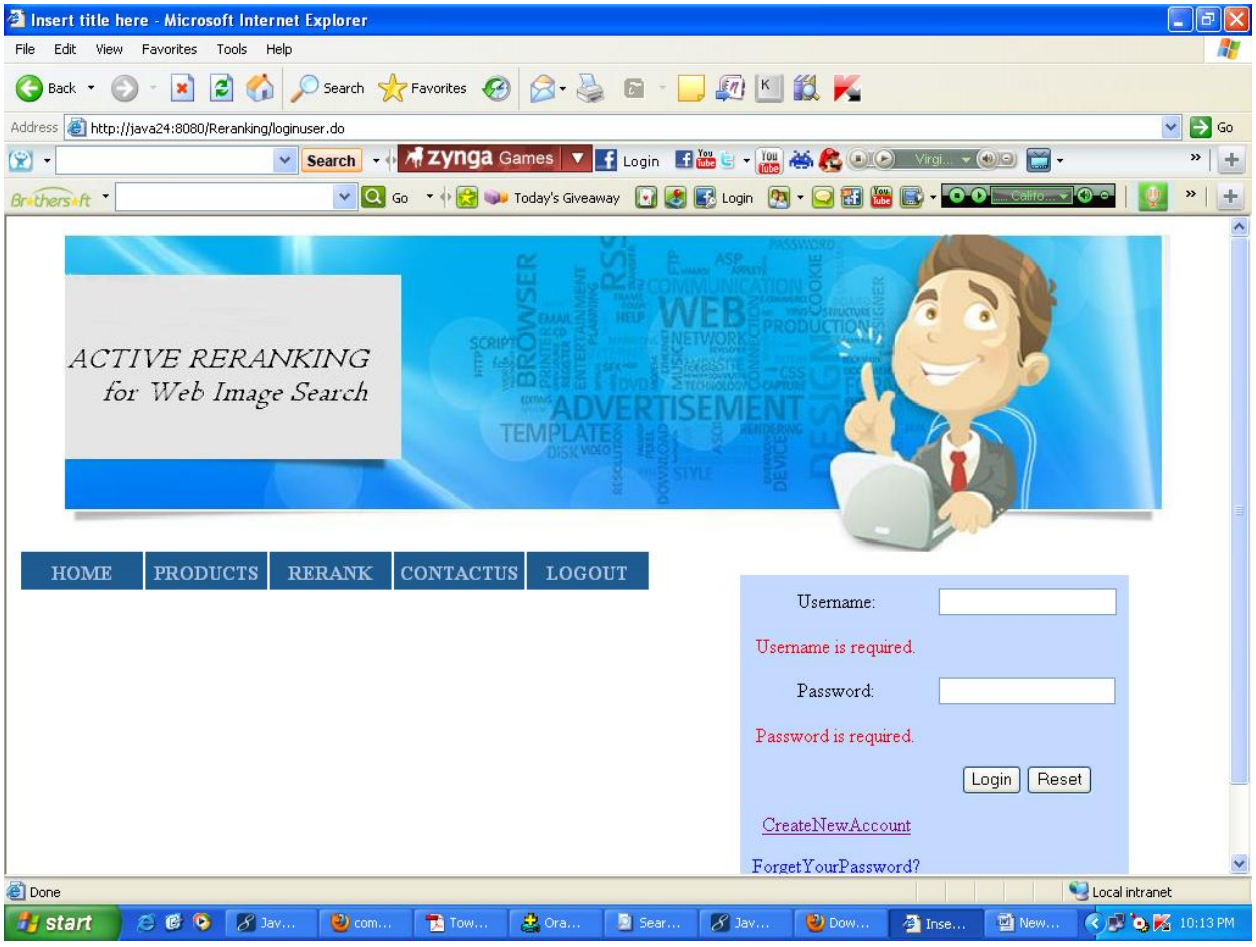
Entity-relationship diagrams don't show single entities or single instances of relations. Rather, they show entity sets and relationship sets. Example: a particular *song* is an entity. The collection of all songs in a database is an entity set. The *eaten* relationship between a child and her lunch is a single relationship. The set of all such child-lunch relationships in a database is a relationship set. In other words, a relationship set corresponds to a relation in mathematics, while a relationship corresponds to a member of the relation. Certain cardinality constraints on relationship sets may be indicated as well.

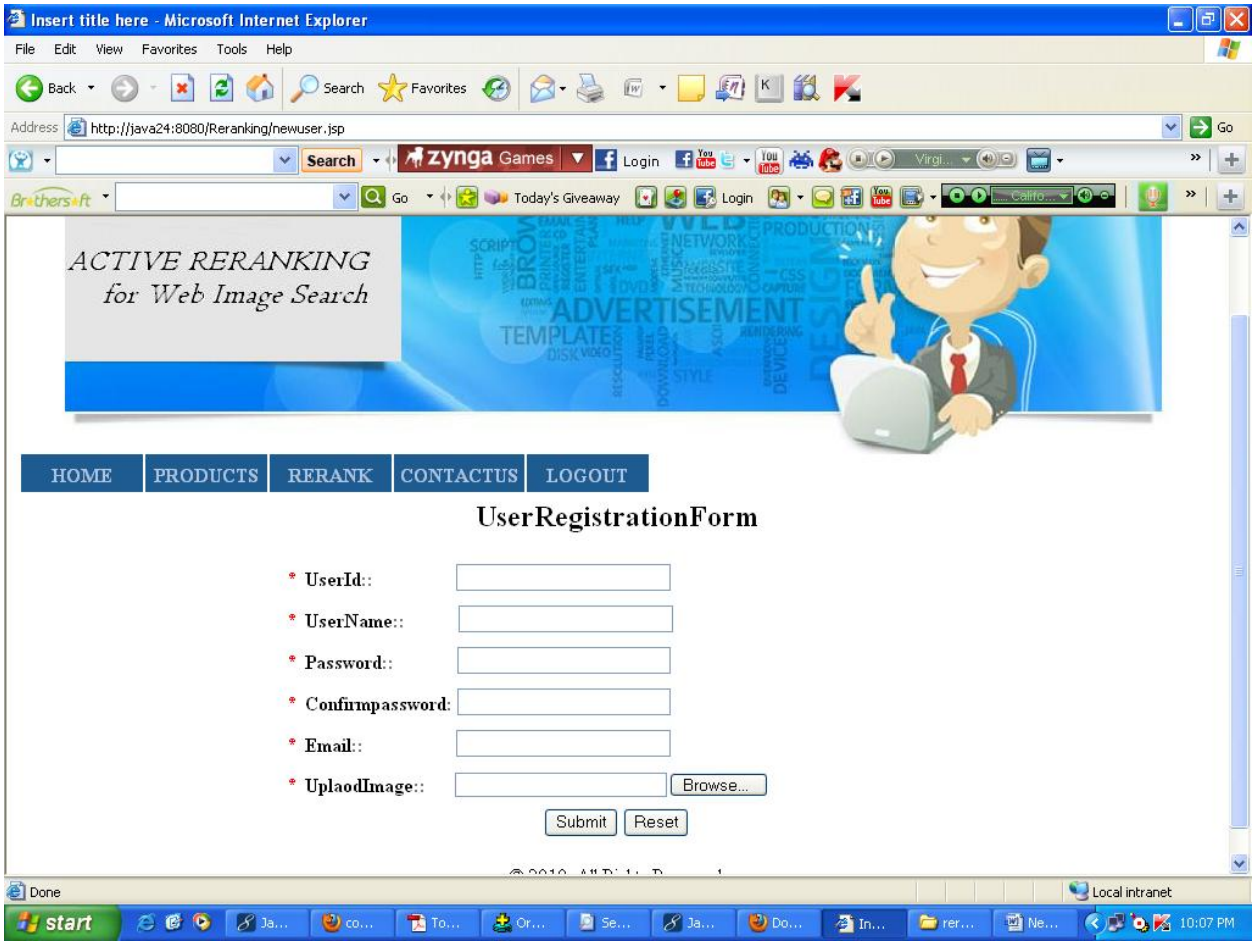


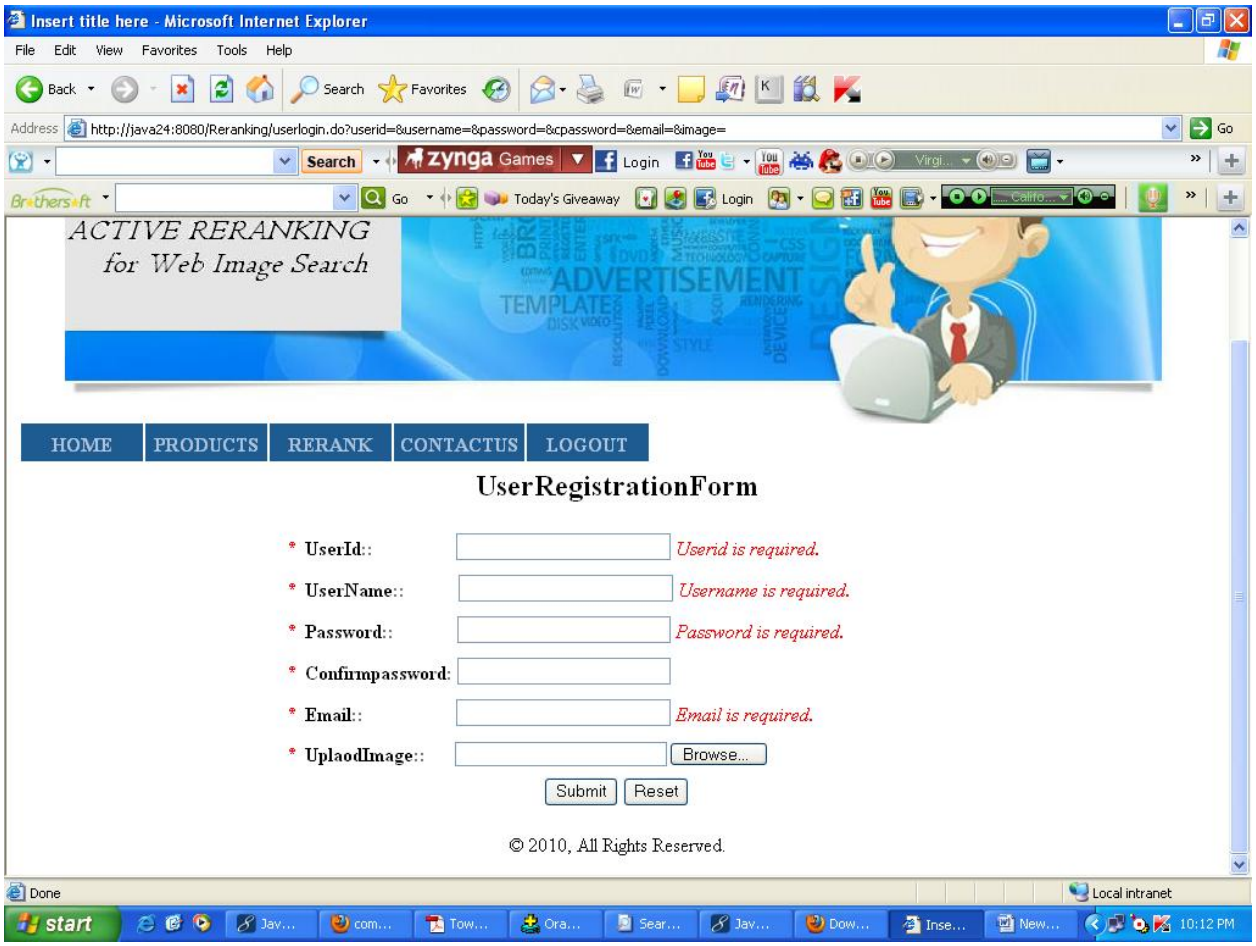
SCREEN SHOTS:

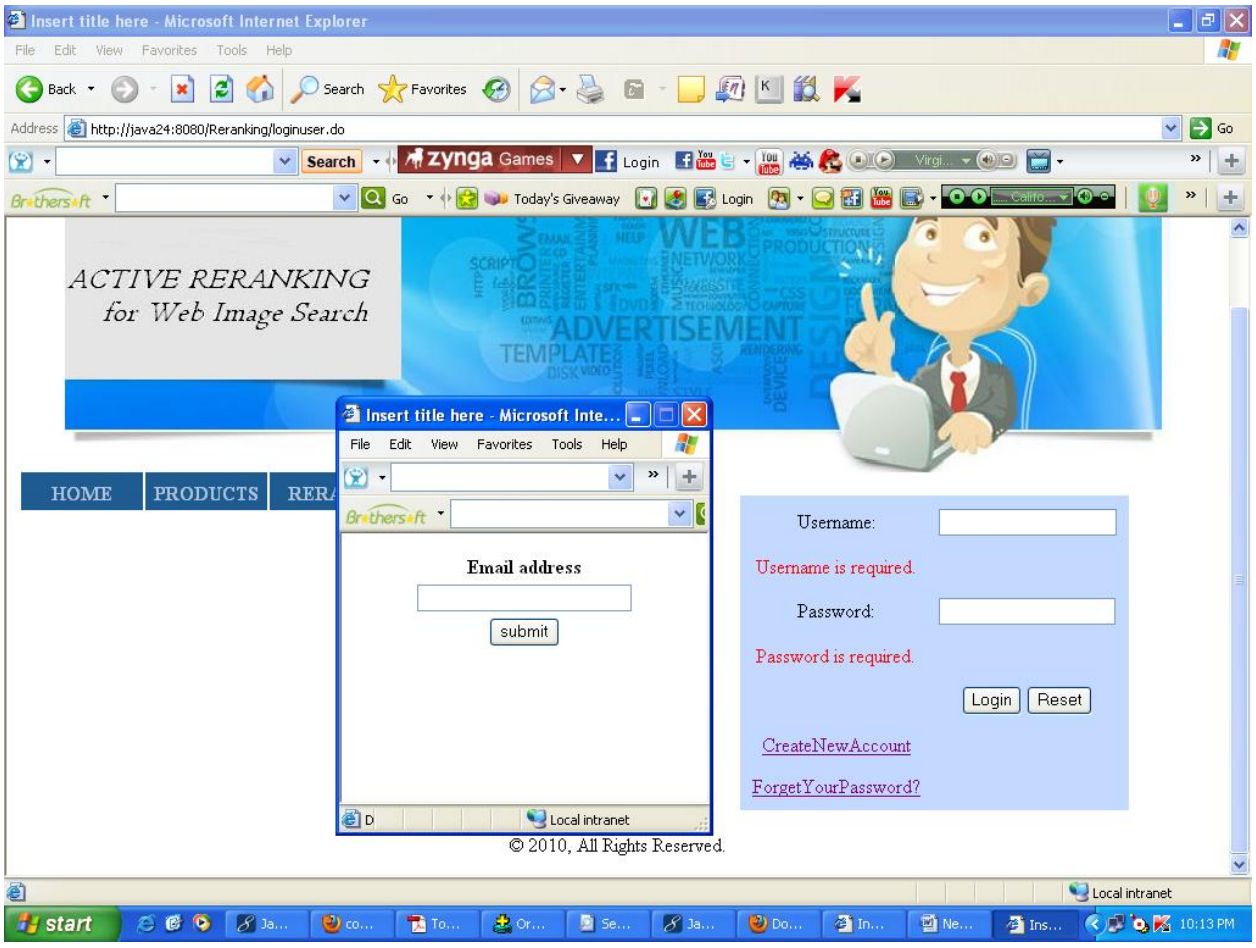
`

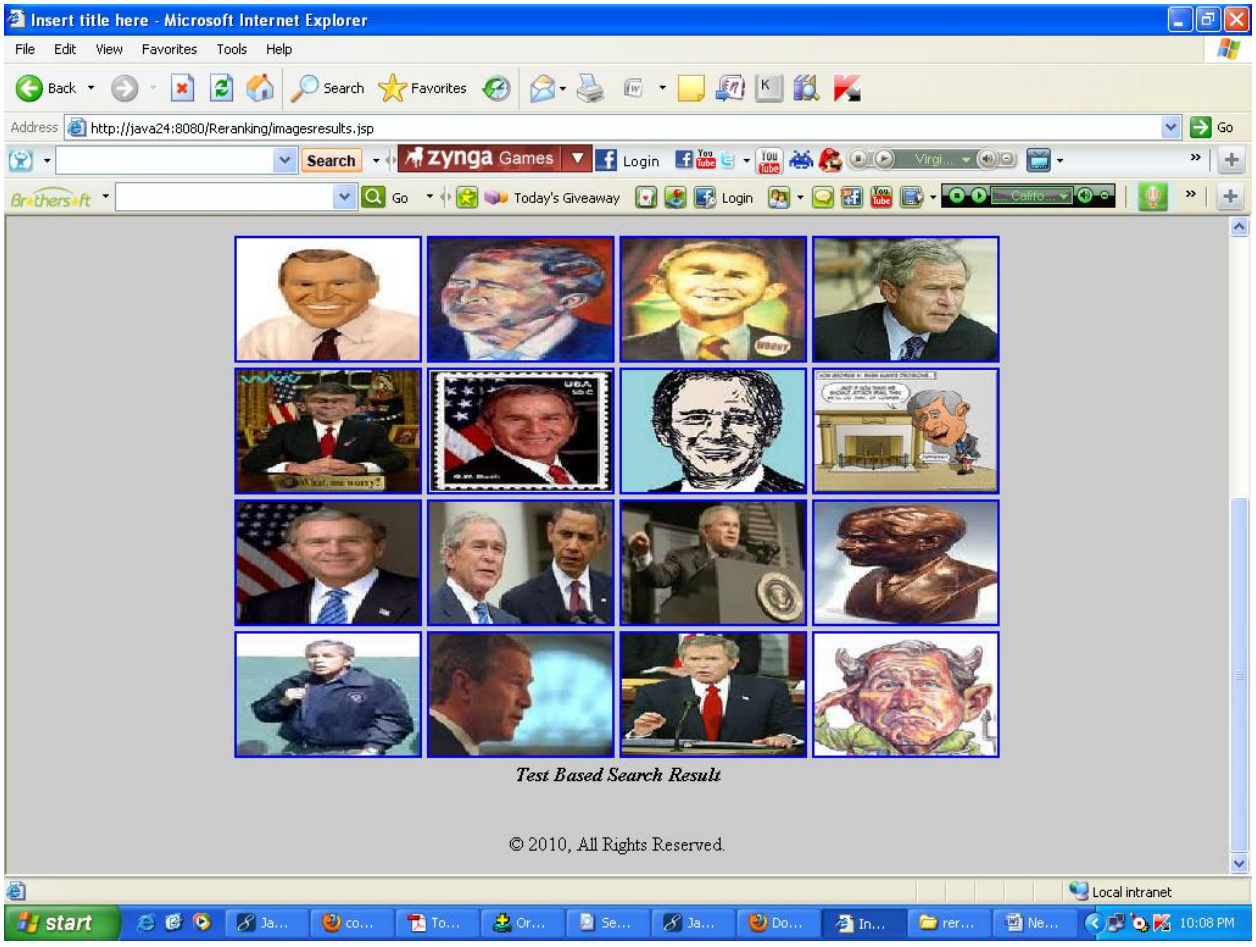


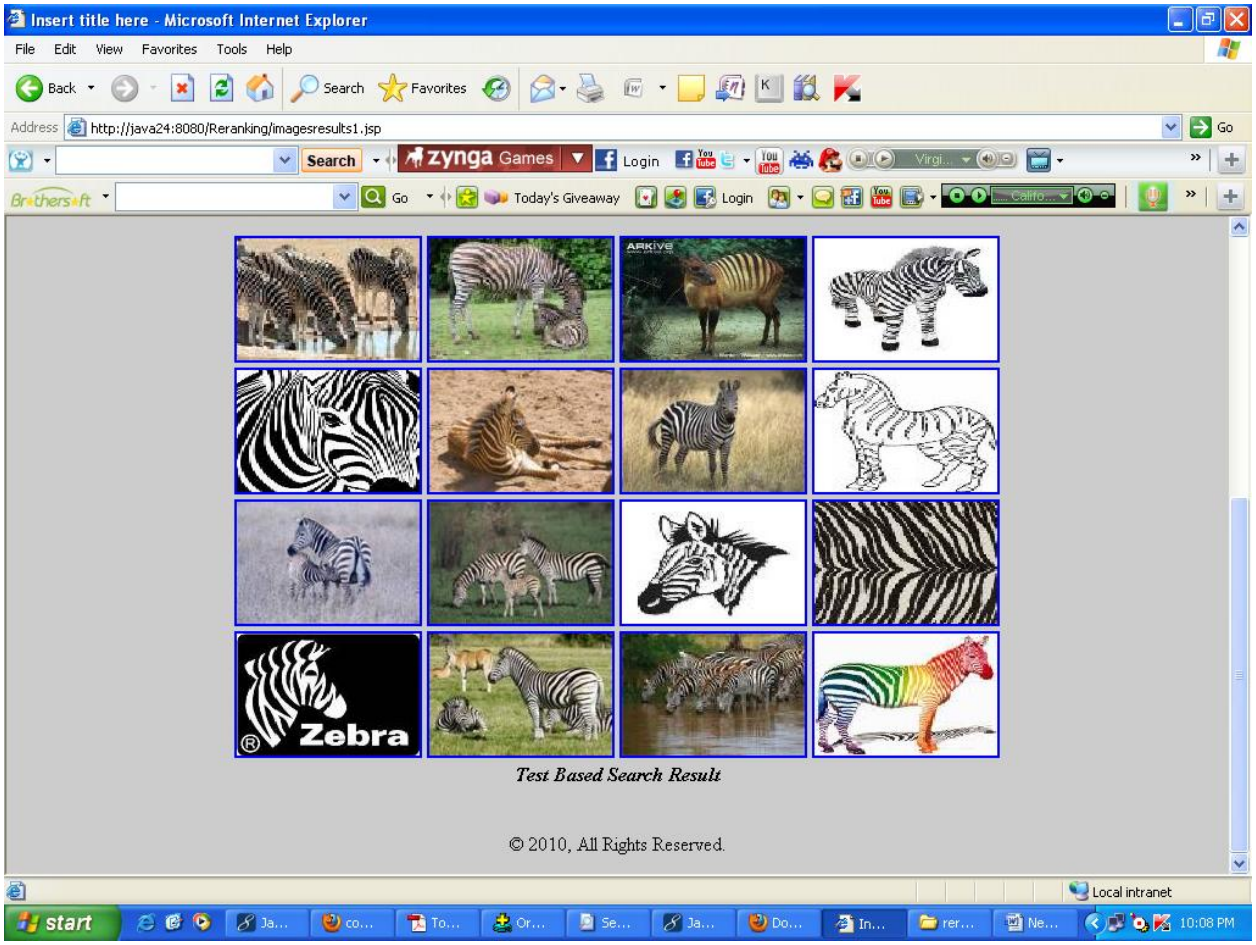

Insert title here - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://java24:8080/Reranking/loginuser.do
Go
Search
zynga Games
Login
Today's Giveaway
ACTIVE RERANKING
for Web Image Search
HOME
PRODUCTS
RERANK
CONTACTUS
LOGOUT
Username:
Username is required.
Password:
Password is required.
Login
Reset
CreateNewAccount
ForgetYourPassword?
Done
Local intranet
start
10:13 PM

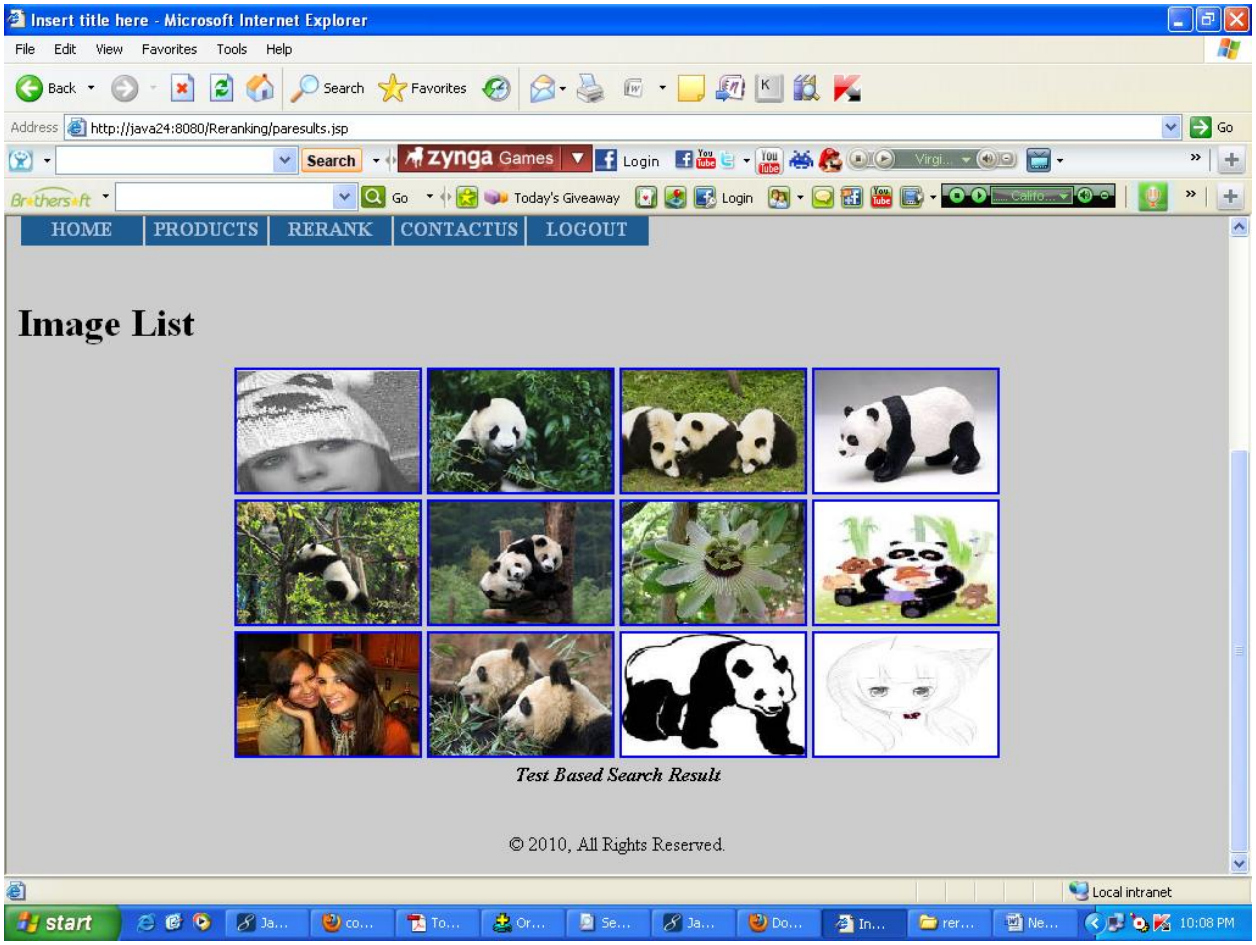

Insert title here - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://java24:8080/Reranking/newuser.jsp
Go
Search
zynga Games
Login
Today's Giveaway
ACTIVE RERANKING
for Web Image Search
HOME
PRODUCTS
RERANK
CONTACTUS
LOGOUT
UserRegistrationForm
* UserId::
* UserName::
* Password::
* Confirmpassword:
* Email::
* UplaodImage::
Browse...
Submit
Reset
Done
Local intranet
start
10:07 PM

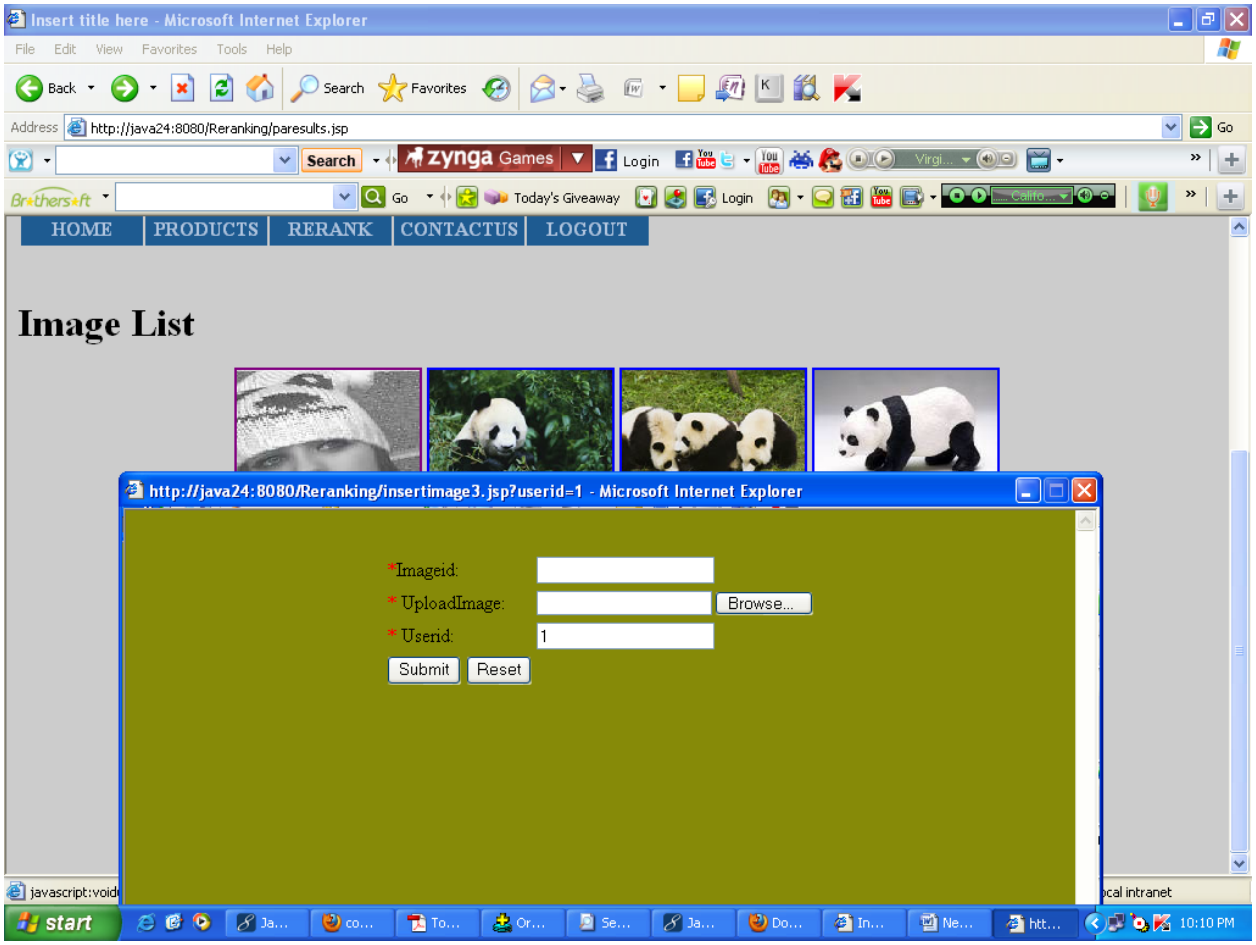

Insert title here - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://java24:8080/Reranking/userlogin.do?userid=&username=&password=&cpassword=&email=&image=
Go
Search
zynga Games
Login
Today's Giveaway
ACTIVE RERANKING
for Web Image Search
HOME
PRODUCTS
RERANK
CONTACTUS
LOGOUT
UserRegistrationForm
* UserId::
Userid is required.
* UserName::
Username is required.
* Password::
Password is required.
* Confirmpassword:
* Email::
Email is required.
* UplaodImage::
Browse...
Submit
Reset
© 2010, All Rights Reserved.
Done
Local intranet
start
10:12 PM

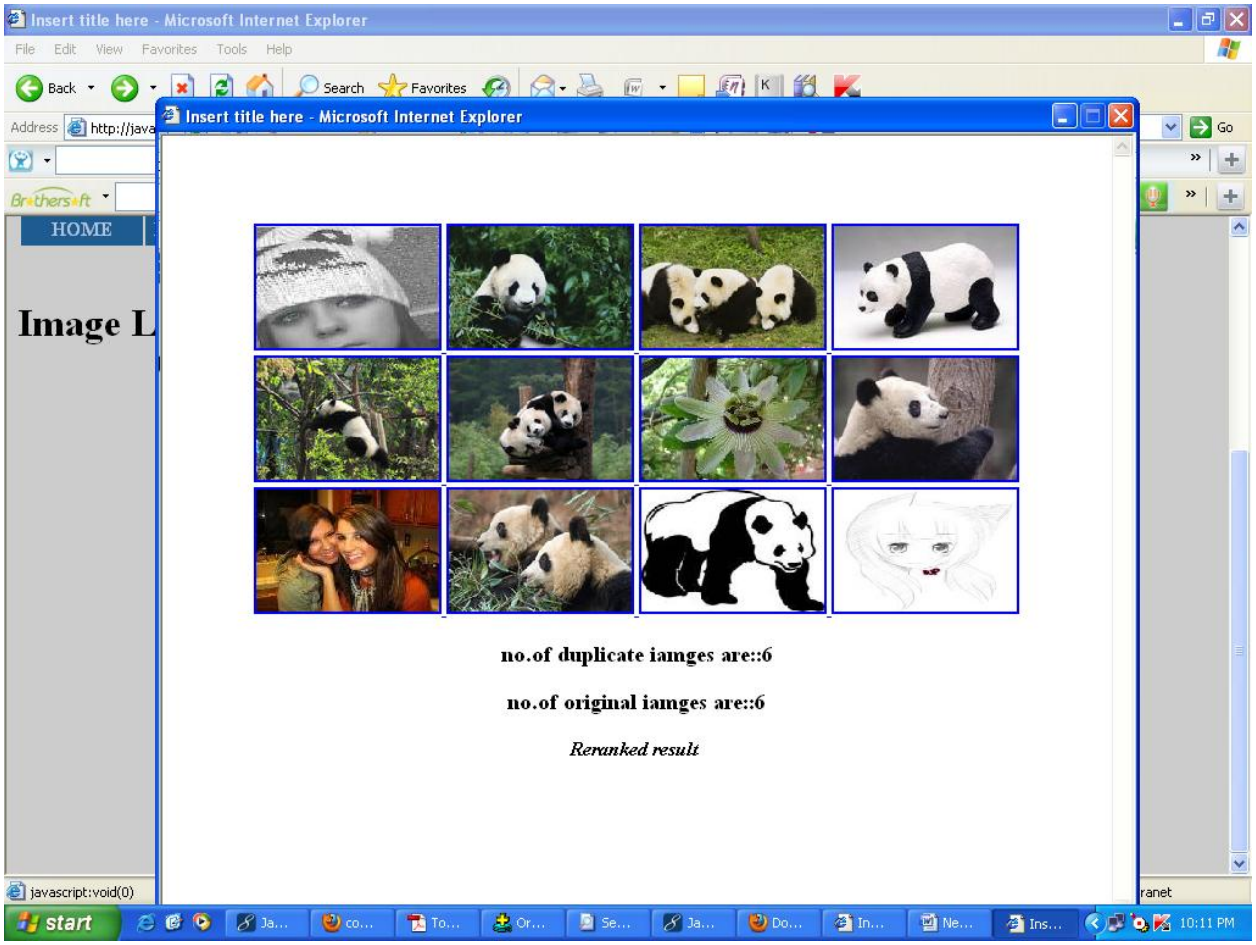

Insert title here - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://java24:8080/Reranking/loginuser.do
Go
Search
zynga Games
Login
Today's Giveaway
ACTIVE RERANKING
for Web Image Search
HOME
PRODUCTS
Insert title here - Microsoft Inte...
File Edit View Favorites Tools Help
Email address
submit
Local intranet
Username:
Username is required.
Password:
Password is required.
Login
Reset
CreateNewAccount
ForgetYourPassword?
© 2010, All Rights Reserved.
Local intranet
start
10:13 PM


Insert title here - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://java24:8080/Reranking/imagesresults.jsp
Go
Search
zynga Games
Login
Today's Giveaway
Test Based Search Result
© 2010, All Rights Reserved.
Local intranet
start
10:08 PM


Insert title here - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://java24:8080/Reranking/imagesresults1.jsp
Go
Search
zynga Games
Login
Today's Giveaway
Zebra
Test Based Search Result
© 2010, All Rights Reserved.
Local intranet
start
10:08 PM


Insert title here - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://java24:8080/Reranking/paresults.jsp
Go
Search
zynga Games
Login
Today's Giveaway
HOME
PRODUCTS
RERANK
CONTACTUS
LOGOUT
Image List
Test Based Search Result
© 2010, All Rights Reserved.
Local intranet
start
10:08 PM


Insert title here - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://java24:8080/Reranking/paresults.jsp
Go
Search
zynga Games
Login
Today's Giveaway
HOME
PRODUCTS
RERANK
CONTACTUS
LOGOUT
Image List
http://java24:8080/Reranking/insertimage3.jsp?userid=1 - Microsoft Internet Explorer
*Imageid:
* UploadImage:
Browse...
* Userid:
1
Submit
Reset
start
10:10 PM


Insert title here - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://java
Go
HOME
Image L
Insert title here - Microsoft Internet Explorer
no.of duplicate iamges are::6
no.of original iamges are::6
Reranked result
javascript:void(0)
start
10:11 PM

CHAPTER-6:

TESTING AND IMPLEMENTATION

TESTING STRATAGIES:

Testing concepts:

a. Testing

b. Testing Methodologies

- Black box Testing:
- White box Testing.
- Gray Box Testing.

c. Level of Testing

- Unit Testing.
- Module Testing.
- Integration Testing.
- System Testing.
- User Acceptance Testing.

d. Types of Testing

- Smoke Testing.
- Sanitary Testing.
- Regression Testing.
- Re-Testing.
- Static Testing.
- Dynamic Testing.
- Alpha-Testing.
- Beta-Testing.
- Monkey Testing.
- Compatibility Testing.

- Installation Testing.
- Adhco Testing.
- Etc….

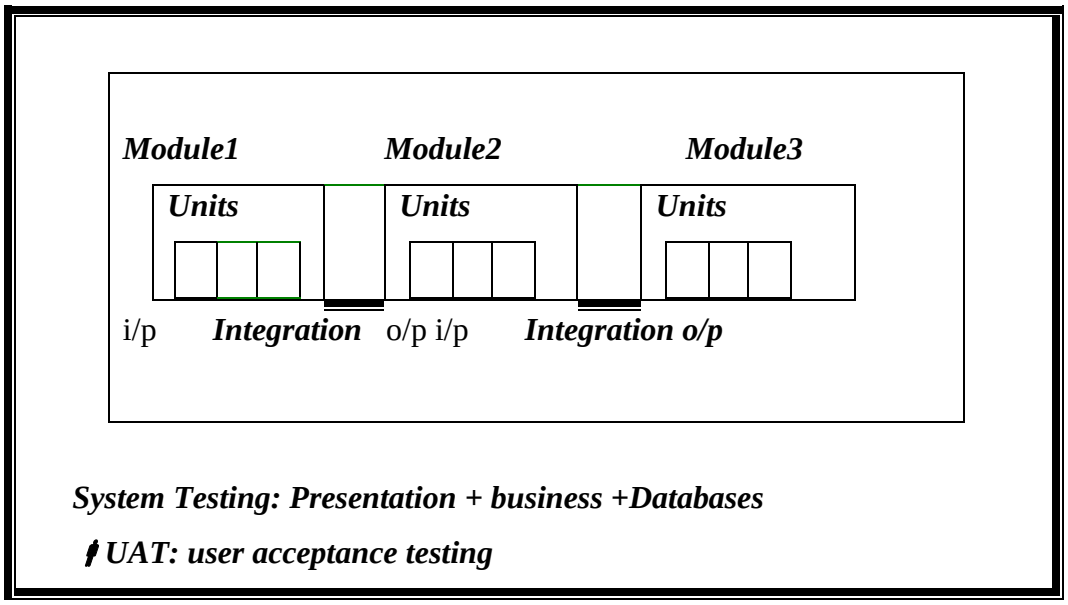
Testing:

- The process of executing a system with the intent of finding an error.
- Testing is defined as the process in which defects are identified, isolated, subjected for rectification and ensured that product is defect free in order to produce the quality product and hence customer satisfaction.
- Quality is defined as justification of the requirements
- Defect is nothing but deviation from the requirements
- Defect is nothing but bug.
- Testing --- The presence of bugs
- Testing can demonstrate the presence of bugs, but not their absence
- Debugging and Testing are not the same thing!
- Testing is a systematic attempt to break a program or the AUT
- Debugging is the art or method of uncovering why the script /program did not execute properly.

Testing Methodologies:

- **Black box Testing**: is the testing process in which tester can perform testing on an application without having any internal structural knowledge of application.
  Usually Test Engineers are involved in the black box testing.
- **White box Testing**: is the testing process in which tester can perform testing on an application with having internal structural knowledge.
  Usually The Developers are involved in white box testing.
- **Gray Box Testing**: is the process in which the combination of black box and white box tonics' are used.

Levels of Testing:



STLC (SOFTWARE TESTING LIFE CYCLE)

Test Planning:

1. Test Planis defined as a strategic document which describes the procedure how to perform various testing on the total application in the most efficient way.

2. This document involves the scope of testing,

3. Objective of testing,

4. Areas that need to be tested,

5. Areas that should not be tested,

6. Scheduling Resource Planning,

7. Areas to be automated, various testing tools used….

Test Development:

1. Test case Development (check list)

2. Test Procedure preparation (Description of the Test cases).

1. Implementation of test cases. Observing the result.

Result Analysis:

1. Expected value: is nothing but expected behavior Of application.

2. Actual value: is nothing but actual behavior of application

Bug Tracing: Collect all the failed cases, prepare documents.

Reporting: Prepare document (status of the application)

Types of Testing:

Smoke Testing: is the process of initial testing in which tester looks for the availability of all the functionality of the application in order to perform detailed testing on them. (Main check is for available forms)

Sanity Testing: is a type of testing that is conducted on an application initially to check for the proper behavior of an application that is to check all the functionality are available before the detailed testing is conducted by on them.

Regression Testing: is one of the best and important testing. Regression testing is the process in which the functionality, which is already tested before, is once again tested whenever some new change is added in order to check whether the existing functionality remains same.

Re-Testing: is the process in which testing is performed on some functionality which is already tested before to make sure that the defects are reproducible and to rule out the environments issues if at all any defects are there.

Static Testing: is the testing, which is performed on an application when it is not been executed. Ex: GUI, Document Testing

Dynamic Testing: is the testing which is performed on an application when it is being executed. Ex: Functional testing.

Alpha Testing: it is a type of user acceptance testing, which is conducted on an application when it is just before released to the customer.

Beta-Testing:

it is a type of UAT that is conducted on an application when it is released to the customer, when deployed in to the real time environment and being accessed by the real time users.

Monkey Testing:

It is the process in which abnormal operations, beyond capacity operations are done on the application to check the stability of it in spite of the users abnormal behavior.

Compatibility testing:

it is the testing process in which usually the products are tested on the environments with different combinations of databases (application servers, browsers…etc) In order to check how far the product is compatible with all these environments platform combination.

Installation Testing:

it is the process of testing in which the tester try to install or try to deploy the module into the corresponding environment by following the guidelines produced in the deployment document and check whether the installation is successful or not.

Adhoc Testing:

Adhoc Testing is the process of testing in which unlike the formal testing where in test case document is used, with out that test case document testing can be done of an application, to cover that testing of the future which are not covered in that test case document. Also it is intended to perform GUI testing which may involve the cosmetic issues.

CHAPTER-7:

CONCLUSION

CONCLUSION:

This paper has presented a novel active reranking frameworkfor Web image search by using user interactions. To target the user's intention effectively and efficiently, we have proposed an active sample selection strategy and a dimension reduction algorithm, to reduce labeling efforts and to learn the visual characteristics of the intention respectively. To select the most informative query images, the structural information based active sample selection strategy takes both the ambiguity and the representativeness into consideration. To learn the visual characteristics, a new local-global discriminative dimension reduction algorithm transfers the local information in the domain of the labelled images domain to the whole image database. The experiments on both synthetic datasets and a real Web image search dataset have demonstrated the effectiveness of the proposed active reranking scheme, including both the sample selection strategy and the dimension reduction algorithm.

Future Enhancements:

1. As we know that our website support based on different urls we are getting the same domains, books etc.
2. To reduce the duplicates our present system is very useful.
3. So, mainly based on url we have to reduce the duplicates.
4. This is very useful because we don't gather the same information from different urls.Actually this is time wasting process.
5. To solve this problem, we are using this present system.

CHAPTER-8:

BIBLIOGRAPHY

REFERENCES:

[1] R. Ananthakrishna, S. Chaudhuri, and V. Ganti, "Eliminating Fuzzy Duplicates in Data Warehouses," Proc. 28th Int'l Conf. Very Large Data Bases, pp. 586-597, 2002.

[2] R. Baeza-Yates and B. Ribeiro-Neto, Modern Information Retrieval. ACM Press, 1999.

[3] R. Baxter, P. Christen, and T. Churches, "A Comparison of Fast Blocking Methods for Record Linkage," Proc. KDD Workshop Data Cleaning, Record Linkage, and Object Consolidation, pp. 25-27, 2003.

[4] O. Bennjelloun, H. Garcia-Molina, D. Menestrina, Q. Su, S.E.Whang, and J. Widom, "Swoosh: A Generic Approach to Entity Resolution," The VLDB J., vol. 18, no. 1, pp. 255-276, 2009.

[5] M. Bilenko and R.J. Mooney, "Adaptive Duplicate Detection Using Learnable String Similarity Measures," Proc. ACM SIGKDD, pp. 39-48, 2003.

[6] S. Chaudhuri, K. Ganjam, V. Ganti, and R. Mitanni, "Robust and Efficient Fuzzy Match for Online Data Cleaning," Proc. ACM SIGMOD, pp. 313-324, 2003.

[7] S. Chaudhuri, V. Ganti, and R. Motwani, "Robust Identification of Fuzzy Duplicates," Proc. 21st IEEE Int'l Conf. Data Eng., pp. 865-876, 2005.

[8] P. Christen, "Automatic Record Linkage Using Seeded Nearest Neighbour and Support Vector Machine Classification," Proc. ACM SIGKDD, pp. 151-159, 2008.

[9] P. Christen, T. Churches, and M. Hegland, "Febrl—A Parallel Open Source Data Linkage System," Advances in Knowledge Discovery and Data Mining, pp. 638-647, Springer, 2004.

[10] P. Christen and K. Goiser, "Quality and Complexity Measures for Data Linkage and Deduplication," Quality Measures in Data Mining, F. Guillet and H. Hamilton, eds., vol. 43, pp. 127-151, Springer, 2007.