12-1-2001

E-commerce, its technology, and implementation

Bin Shu
University of Nebraska at Omaha

E-commerce, its technology, and implementation

A Project
Presented to the
Department of Computer Science
And the
Faculty of the Graduate College
University of Nebraska
In Partial Fulfillment
Of the Requirement for the Degree
Master of Science
University of Nebraska at Omaha

By
Bin Shu
December 2001

ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 - 1346

PROJECT ACCEPTANCE

Acceptance for the faculty of the Graduate College, University of Nebraska, in partial fulfillment of the Requirements for the degree (Master of Science), University of Nebraska at Omaha

Committee

Dr. Peter NG _____

Dr. Quiming Zhu _____

Dr. Bin Chen _____

Chairperson Dr. Peter NG _____

Date NOV 27, 2001

E-commerce, its technology and implementation

Bin Shu, MS
University of Nebraska, 2001

Advisor: Dr. Peter Ng

Electronic commerce refers to business activities involving consumers, manufacturers, service provider, and intermediaries using computer networks such as the Internet. The goals of e-commerce are to reduce product and service cost and improved customer response time and quality. Hence implementing initiatives in electronic commerce has emerged as a significant business strategy in the Information Age. Technological developments made possible by the convergence of the telecommunication and computing industries have opened the door to a world of new and exciting application that are changing the way business is conducted. A large number of businesses of all sizes have either begun to implement e-commerce strategies and initiatives, or have plans to do so in the immediate future.

Electronic commerce is a multidisciplinary field that includes technical areas such as networking security, and storage and retrieval of multimedia; business areas such as marketing, procurement and purchasing, billing and payment, and supply chain management; and legal aspects such as information privacy, intellectual property, taxation, contractual and legal settlements. This project contains a concise discussion of current research and future challenge of the electronic commerce, a comparison of current e-commerce technology, and an implementation of web business using Java programming.

Table of Contents

Chapter 1 Introduction

For some time now, large business enterprises have used electronic commerce to conduct their business-to-business transactions. Electronic data interchange (EDI) on private networks began in 1960s, and banks have been using dedicated networks for electronic funds transfer (EFT) almost as long. Recently, however, with the increased awareness and popularity of the Internet, electronic commerce has come to encompass individual consumers as well as businesses of all sizes.

The reasons for a success on the Internet are radically different to what we have seen in the past in business. It is not anymore the bigger fish swallowing the smaller fish, nor is it the faster runner beating the slower runner. In the information society the more knowledgeable is making more deals than the less informed. Knowledge is quality and this is where the business is heading for.

Through the Internet everything moves closer together, resulting in nearly zero response time and almost no distance. As the Internet is getting faster every day through new inventions and new programs, every company will be as fast as its competition. Distance, size and speed become irrelevant. In order to be successful the service needs to be better than the competition. Quality of service becomes the ultimate success factor.

The Internet reduces the three dimensions of the world and time to a single point. Instead of serving customers form nine to five, the customers are served around, whenever a customer feels like needing a product or service. Time zones and geographical boundaries have no importance anymore. Besides, all customers have their own universe, which needs to be addressed when offering goods, information or services online.

Through personalization, the universe of the Internet appears differently to everyone. The

Internet is constantly changing; making change the only constant one can count on. Products, ideas and prices are changing much faster than ever before.

The Internet is already changing the way that many companies conduct their business. As that influence grows, and more companies use the Internet, the possibilities for conducting business-to-business and business-to-consumer commerce on the Internet will expand greatly, and become more of a routine part of commerce than it is today. We've reached a critical mass that where everyone thinks of conducting business on the Internet everyday.

Chapter 2 Electronic Commerce

2.1 Electronic commerce-Definition and Market

Electronic commerce has already existed for over 20 years. In sectors such as retail and automotive, electronic data interchange (EDI) for application-to-application interaction is being used regularly. For defense and heavy manufacturing, electronic commerce lifecycle management concepts have been developed that aim to integrate information across larger parts of the value chain, from design to maintenance.

These forms of electronic commerce have been fairly limited in their diffusion and take-up until now. For example, no more than about 50000 enterprises in Europe and 44000 in the USA use EDI, although it was intended to have wide applicability. This represents not even 1% of the total number of companies.

Recently, however, we have seen explosive development in electronic commerce. The causes of that are, of course, the Internet and World Wide Web, which are making electronic commerce much more accessible. They promise easily usable and low-cost forms of electronic commerce. The Internet not only supports application-to-application electronic commerce similar to that already known from EDI, but also, and especially, person-to-person and person-to-application forms of electronic commerce.

Through the combination of interactivity, networking, multimedia and data processing, Internet electronic commerce offers a tremendously wide variety of electronic business opportunities, limited only by imagination. Electronic commerce on the basis of the Internet is set to become a very important way of doing business.

Internet electronic commerce includes electronic trading of physical goods and of intangibles such as information. This encompasses all the trading steps such as online

marketing, ordering, payment, and support for delivery. Electronic commerce includes the electronic provision of services, such as after-sales support or online legal advice. It also includes electronic support for collaboration between companies, such as collaborative online design and engineering, or virtual business consultancy teams. This wide range of business activities illustrates that it makes little sense to come up with a restrictive definition of electronic commerce, but also that there is likely not to be a single unique definition.

Forrester Research forecast was that business-to-business electronic commerce would grow to $327 billion in the year 2002-that is, the value of goods and services traded via the Internet. This excludes that value of the hardware, software and services that are needed to perform electronic commerce, whose values is likewise estimated at several hundred billions of dollars. It also excludes the value of other forms of electronic commerce mentioned before, such as collaborative design and engineering or electronic trading in financial markets. While this is only 1% of the forecast total world economy of $30 trillion in 2002, the growth is predicted to be exponential. Forrester expected B-to-B electronic commerce to jump to $1.3 trillion in US by 2003, which already amounts to 4% of the world economy. Others predict that the Figure 5.will grow further to 30% of the world economy in 2010. While such high growth rates will not be sustained, it is clear that electronic commerce will become pervasive: Datamonitor expected 630000 US companies and 245000 European companies to be involved in fully fledged integrated B-to-B electronic commerce in five years' time. The EITO's 99 survey expected 47% of European companies to be using some form of electronic commerce in 2001. In March 2000 IBM claimed to be selling more than $1 billion online. According to a survey by

Grainger, a US-based MRO distributor, in 1998 only 8% of companies interviewed were using the Internet to order maintenance, repair and operations (MRO) supplies. This can be contrasted with data from The Report on Electronic Commerce (1998), which expects that Internet connectivity in business will grow from 10% in 1997 to an impressive 90% in 2001.

The number of individual connected to the Internet is expected to continue to grow explosively. Likewise, the tremendous growth of the number of Internet host computers will not abate for the next few years. However, business-to-consumer electronic commerce, although also growing fast, is not expected to reach such high levels within the next few years.

With the new medium of the Internet, new ways of doing business are also developing. Most of those that capture the public attention are consumer oriented (such as Amazon.com). Less publicity is given to the way that the Internet can be used for business-to-business electronic commerce, although such commerce is a reality today. The state of the market differs greatly around the world, with the USA being in the lead in most areas of electronic commerce. The European market is estimated to be between one and three years behind, although it is catching up, with growth rates in business-to-business electronic commerce in 1999 that were higher than in the USA. Europe is more advanced in some electronic commerce-related services such as in electronic payments and in smart-card usage. Some to be developing even more rapidly to overtake the European market in a few years' time predicts the Asia-Pacific market.

2.2 Strengths and Advantage of E-business

The strengths of e-business depend on the strengths of the Internet, which is the preferred infrastructure today and in the future. The Internet is available all over the world, twenty-four hours a day, seven days a week. It is simple to use and the transaction costs for the end user are low. The costs are also extremely low for the vendors on the Internet, compared to traditional distribution channels. The Internet allows two-way communications and is built around open standards. The two-way communication allows for direct feedback of the customers and the open standards mean interoperability between companies, web sites and services. It is fairly easy to integrate processes, services and products, once they have been digitized.

Using the latest software, it is possible to customize our entire web site for every single user, without any additional cost. The mass-customization allows us to create web pages, products and services that suit the requirements of the user. A customized web page does not only include the preferred layout of the customer, but also a pre-selection of goods the customer may be interested in. Internet pricing becomes irrelevant, as all prices drop to the lowest possible level. The only chance to distinguish the products of your company from the ones of our competitor is to add services that increase the value of the product without increasing its price.

Although many people are afraid of security breaches on the Internet, it can be made very secure through encryption, digital signatures and firewall software and secure procedures. This will allow companies to offer private information to their customers and business partners without having to fear that an unauthorized person is able to see that particular information. Banks, for example, are able to allow customers to look at their account

balance in real time without have to worry that a hacker will be able to break into the bank's computer system. This is achieved through the use of the above-mentioned security components, which allow trade on the Internet to expand.

Expanding Market Reach

One of the major advantages of the Internet is its global availability. If we have a small company it is quite simple to expand the market reach beyond our geographic location and our current customer segments. Although this may relieve some of the pressure we experience in our current target market it will mean new pressure from competitors who are already on the Internet and are trying to get into our market.

On the Internet every company that offers goods, services or information is reduced to the same size: to the size of the customer's browser window. Therefore it is easy for a small online translation service and the way the company presents itself on the Web. This and what other people say about the online service are the basis for the decision. Marketing for the web site is important. Many people choose a web site because others are talking about it or because they have seen advertisements for it. If they have choice, customers will prefer to go to the web site whose brand name is well known. But other than with traditional shops, the customers most probably will also visit the second to double-check prices and offerings. Moving from one shop to the other costs only a few seconds and the customer does not feel the pressure of a shop assistant who may help the customer in making his or her decision.

Responsiveness

The Internet can support increased responsiveness to our customers easily. Increasing responsiveness to customers and partners is very important to tie customers to a

company. Being responsive give customers the feeling that they are treated well by the company.

Offering New Services

Offering new service is also a reason to go online. Introducing new services in traditional markets is difficult and expensive. The Internet on the other hand offers the possibility to introduce new services with very little start-up costs. New service should not only be provided for customers and partner, but also for employees. A service for the employees could be for example a search engine for the Intranet. The larger the company grows the harder it is to find relevant information on the internal network. A search engine is only helpful if all employees put their documents online. Even if they are not able to create HTML documents, it is fairly easy to upload existing word documents to the Intranet, which can be indexed by the search engine as well. The search hit-rate of non-web documents is lower than with HTML documents, but still much higher than not putting them online at all.

Cost Reduction

The cost for estate, service support and production can be reduced greatly through the use of the Internet. This results in more content customers and less overhead. So it is another very good reason to move business to the Internet.

Costs can also be reduced in the customer care center by offering frequently asked questions (FAQ) pages, where customers can find answer to frequently asked questions about a product or service. Newsgroups where customers can ask questions can also be very helpful. Other customer may be able to share their experiences and reduce the workload for the customer care center. In addition, companies can support employees and

business partners over their corporate Intranet, keeping them informed and soliciting their feedback.

A company web site can also help to reduce inventory costs by shortening the sales and supply cycles. By distributing information in electronic form, we can reduce material costs by saving on paper, the printing and the manual distribution. The customer is taking over parts of the distribution costs.

Cost saving should not be seen as a primary goal in the long term. In the long term everybody will have saved cost and increased the profit. In order to survive it is necessary to have a deep relationship with our customers. This will allow us to charge more money for a service than others do, because service quality is what matters, not the base product.

Strengthening Business Relationship

Implementing business-to-business communication on the Internet has a huge potential. In the past many industries have been using electronic data interchange (EDI) to simplify business process and reduce the cost of communication between the business partners. Through EDI suppliers, manufacturers, distributors and retailers are able to share information on the inventory and enhance the flow of information and goods through the supply chain. Passing on the information electronically reduced the cost of communication and the number of errors.

The disadvantage of EDI is that it is very expensive and time consuming to implement; therefore many SMEs have not implemented it. Once a company has implemented it, every partner that uses it needs to implement it as well. Even if two companies have an existing EDI infrastructure, the special connection between these companies needs to be

implemented. Consider a manufacturer with 50 suppliers, the cost are enormous for the manufacturer as it has to implement 50 EDI infrastructures.

The paradigm of EDI is good, but the technology was too expensive. With the Internet it has become accessible for all companies. Costs that have been reduced by fifty times are common and EDI on the web allows for more content. Exchange of multi-media information has been made possible and fosters much tighter relationships among participants. The real-time capability of the Internet provides a sense of teamwork and shared goals. EDI via the Internet enables all components and systems of a virtual value chain to communicate with each other automatically.

Generating Visibility

Another important goal, especially for small and medium-sized enterprises is to gain more visibility. The Internet allows a company to present itself at very low cost. Although buying a computer and setting up an Internet connection may not be cheap, once you have it, setting up new web pages and adding prices, products and information costs very little and the costs for reproduction are practically nothing. You do not need to replicate a catalogue, a brochure or a flyer. Put it onto the Internet and it replicates itself. Each user generates its won copy when accessing your web server. This is especially true when you use one-to-one marketing tools that allow the customer to see a personalized view of your products, services and information. Through co-branding, you are also able to present your products and services on other web sites.

Generating visibility is substantial for every company. The better known your company is, the more people will be interested in doing business with you. In the early years of the Internet being online was a synonym of being cool and forward-thing, but it was in no

way a must to be online. This may be true for certain industries, although to find an industry where this is still true. Missing the opportunity to present our own company on the Internet, even with only a simple web site, is something nobody can do today.

Chapter 3 Web architecture and technology

Over the past few years, we've witnessed some incredible changes in the way we think about computers. We are no longer bound by large and cumbersome applications on the desktop. With the introduction of the Internet and the World Wide Web, we can now access information, and do business, from virtually anywhere. The challenge that we now face is that of raising the bar again. With these recent advances in technology also comes the demand for faster, lighter, and more robust applications that we deliver across the Web.

Fortunately, we have some powerful tools to work with. In the past, if we wanted to deliver a database-driven application to customers over the Web, we were pretty much limited to writing CGI scripts to process form data and return some results. But in the last couple of years, a large number of technologies have appeared. The only trouble is trying to sort through the options and decide what are right for our organization and us.

3.1 Related Software

To open a web page in our browser, we usually either type in the URL or click an existing link to the URL. Once we submit this request and the web server receives it, the web server locates the web page and sends it back to the browser. The browser then displays the page. A URL also references each image in the page and the browser requests each image URL from the server in the same way it request the main HTML page.

The Web Browser

The web browser can be thought of as a universal user interface. Whether you're doing simple web browsing or transacting online banking, the web browser's responsibilities

are that of presenting web content, issuing requests to the web server, and handling any results generated by the request.

In the past couple of years, we've seen considerable advances in the browser market. Both Microsoft and Netscape have continually raised the bar, giving us some incredible power on the client side. Both of the major browsers, Microsoft Internet Explorer and Netscape Communicator, have evolved into fully programmable document containers. Each has its own object model allowing for scripts, or objects, to manipulate the elements of the document itself. Scripting languages like VBScript or JavaScript.

The Web Server

At the heart of any web interaction is the web server. The web server is a program running on the server that listens for incoming requests and services those requests as they come in. Once the web server receives a request, it then spring into action. Depending on the type of request, the web server might look for a web page, or it might execute a program on the server. Either way, it will always return some kind of results to the web browser, even if it's simply an error message saying that it couldn't process the request.

According to the Netcraft Web Server Survey (http://www.netcraft.com/survey), the leading web servers today are the Apache web server and Microsoft's Information Server. The apache web server has been developed as free software and has been contributed to by programmers around the world. Its power, flexibility, ease of use, and the availability for multiple platforms as contributed to its rise in popularity over the past few years. Microsoft's IIS, on the other hand, runs on the Windows NT operating system and is included as part of the Window NT system. While Microsoft's IIS offers a wide

range of features, its dependence on the Windows operating system may be holding it back. With the current rise in popularity that the Linux operating system has enjoyed, it is likely that the Apache web server will continue to gain ground on the competition. The following tables are the most current result of survey made by Netcraft.

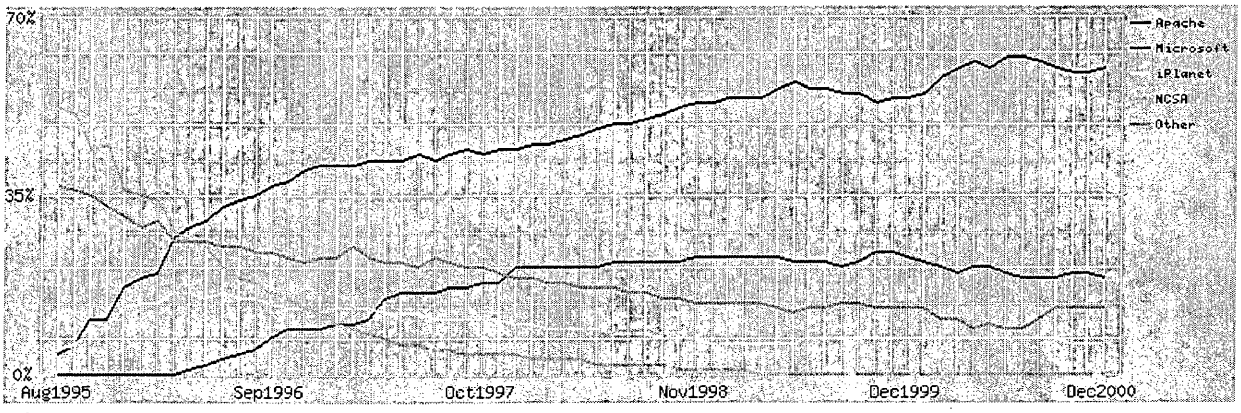
Top Developers

Developer	November 2000	Percent	December 2000	Percent	Change
Apache	14193854	59.69	15414726	60.04	0.35
Microsoft	4776220	20.09	5027023	19.58	-0.51
iPlanet	1643977	6.91	1722228	6.71	-0.20

Top Servers

Server	November 2000	Percent	December 2000	Percent	Change
Apache	14193854	59.69	15414726	60.04	0.35
Microsoft-IIS	4774050	20.08	5025017	19.57	-0.51
Netscape-Enterprise	1605438	6.75	1682737	6.55	-0.20
WebLogic	789953	3.32	890791	3.47	0.15
Zeus	640386	2.69	676526	2.63	-0.06
Rapidsite	347307	1.46	365807	1.42	-0.04
thttpd	226867	0.95	321944	1.25	0.30
tigershark	120213	0.51	139300	0.54	0.03

AOLserver	136326	0.57	125513	0.49	-0.08
WebSitePro	106618	0.45	110681	0.43	-0.02



3.2 Web Application Architecture

When discussing web application development, it is appropriate to introduce the concept of n-tier architecture. Typical client/server systems have fallen into the category of a two-tiered architecture. The application exists entirely on the client PC while the database sits out on a server somewhere in the organization. While this approach allows us to share the enterprise, it does have many drawbacks.

In a two-tiered application, the processing load is given to the PC while the more powerful server simply acts as a traffic controller between the application and the database. As a result, not only does the application performance suffer due to the limited resources of the PC, but the network traffic tends to increase as well. When the entire application is processed on a PC, the application is forced to make multiple requests for

data before even presenting anything to the user. These multiple database requests can heavily tax the network.

Another problem with a two-tiered approach is that of maintenance. Even the smallest of changes to an application would involve a complete rollout to the entire user base. After a few rollouts, it may become hard to manage which version exists where. Some users may not be ready for a full rollout and ignore the change while another group insists on making the change immediately. As a result, we now have two separate versions of the software to maintain.

To address these issues, the software community developed the notion of a three-tier architecture. The application is broken up into three separate logical layers, each with a well-defined set of interfaces. The first tier is referred to as the presentation layer and typically consists of a graphical user interface of some kind. The middle tier consists of the application logic and the third tier contains the data that is needed for the application. The middle tier (application logic) is basically the code that the user calls upon (through the presentation layer) to retrieve the desired data. The presentation layer then receives the data and formats it for display. This separation of application logic from the user interface adds enormous flexibility to the design of the application. Multiple user interfaces can be built and deployed without ever changing the application logic, provided the application logic presents a clearly defined interface to the presentation layer.

The third tier contains the data that is needed for the application. This data can consist of any source of information, including an enterprise database like Oracle or Sybase, a set of XML documents (data that has been stored in a well-formed documents conforming to

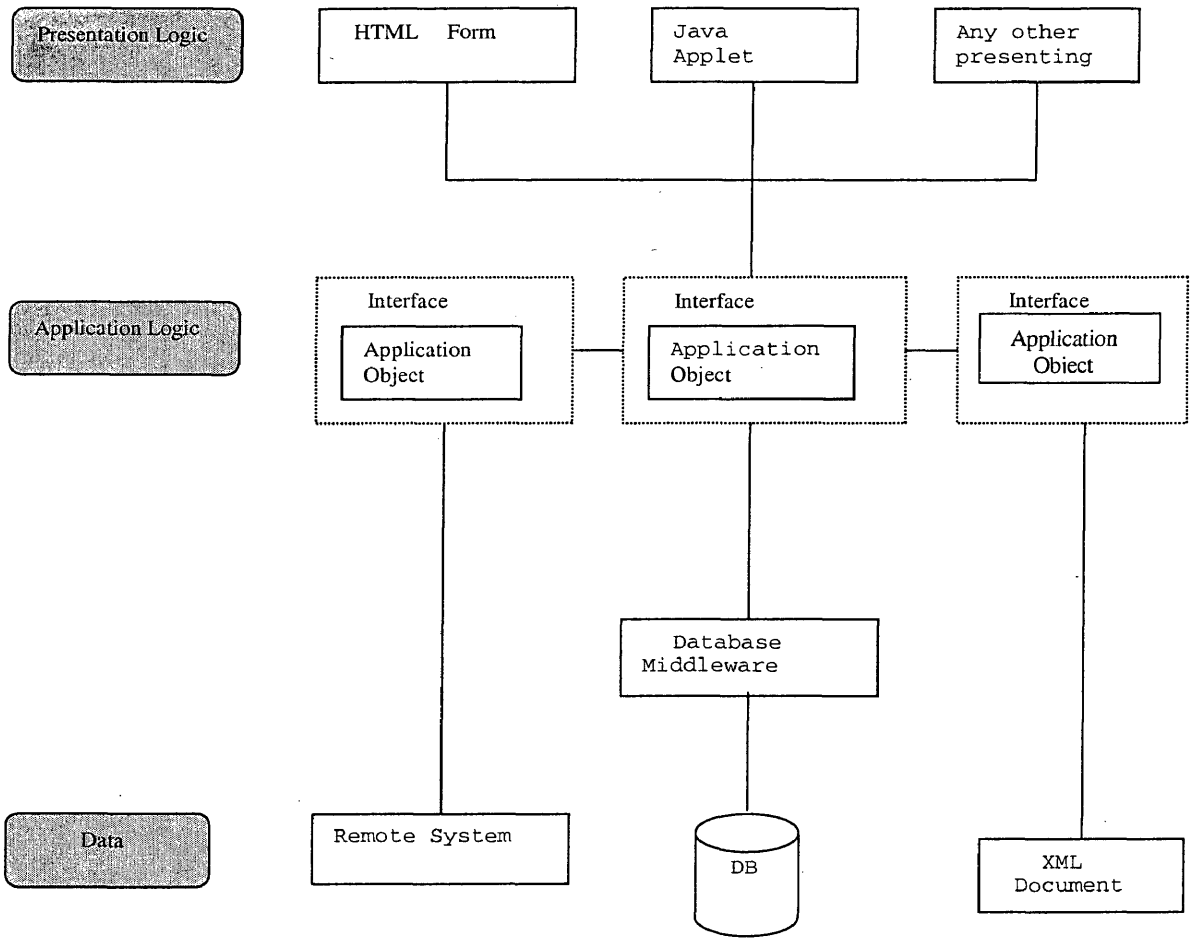
the XML specification), or even a directory service like an LDAP server. The data for an application is not limited to just a relational database. There are many different sources of enterprise data that our applications can access.

By employing a three-tier architecture, the issues of performance, network traffic, and maintenance have all been addressed. This approach is almost where we would like it, but not quite yet. What it lacks is reusability and scalability. We could still end up with a series of "stovepipes" within an organization. We'd have dozens of applications that don't communicate with one another.

This is where n-tier architecture comes in. To turn a three-tier system into an n-tier system, we simply extend the middle tier by allowing for multiple application objects rather just a single application (see figure below). These application objects must each have an interface, which allows them to work together. An interface can be thought of as a contract. Each object states through its interface that it will accept certain parameters and return a specific set of results. Application objects communicate with each other using their interfaces.

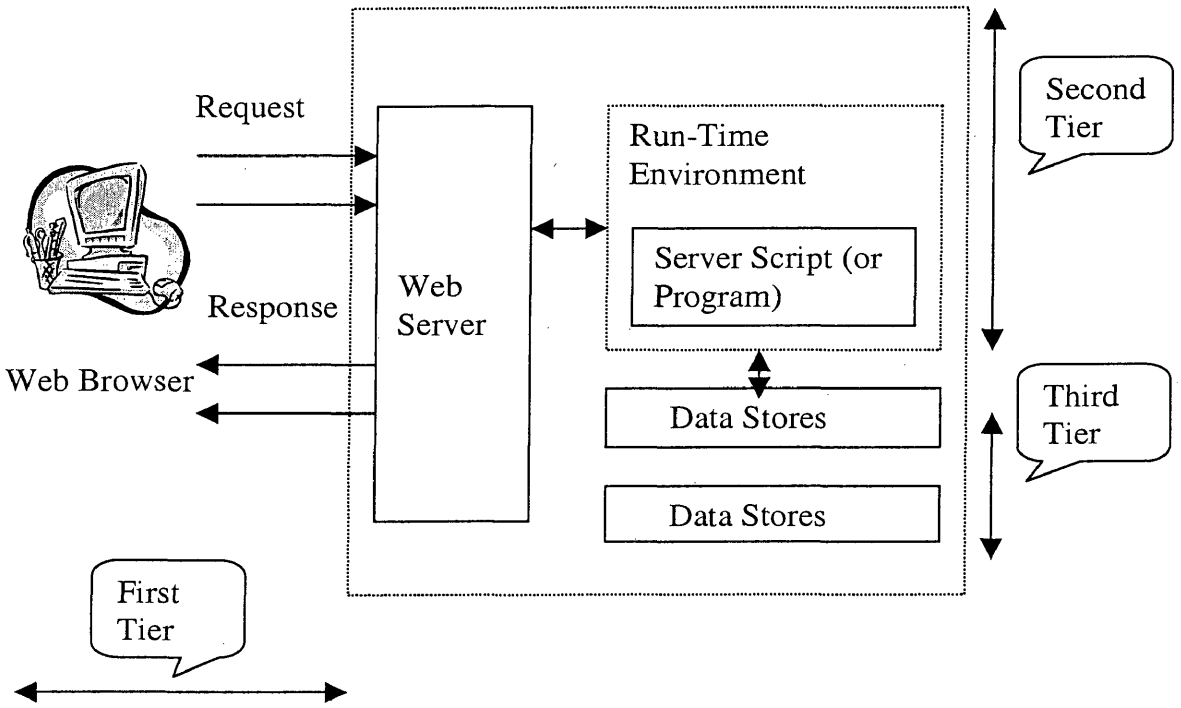
With n-tier architecture, we can now have multiple applications using a common set of business objects across an organization. This promotes the standardization of business practices by creating a single set of business functions for the entire organization to access. If a business rule changes, then changes have to be made to only the business object and, if necessary, its interfaces, and subsequently any object that access the interface. It is important to note that when designing an object and its interfaces, it is a good idea to make the interface as generic as possible to avoid a lot of changes later on.

Since other objects communicate with the interface and not the object itself, changes to the object, and not the interface, are relatively simple and quick.



A web application typically follows a three-tiered model (see figure below). The first tier consists of the presentation layer, which in the case of a web application includes not only the web browser but also the web server, which is responsible for assembling the data into a presentable format. The second tier is the application layer. It usually consists of some sort of script or program. Finally, the third tier provides the second tier with the data that it needs. A typical web application will collect data from the user (first tier),

send a request to the web server, run the requested server program (second and third tiers), package up the data to be presented in the web browser, and send it back to the browser for display (first layer).



Collecting the Data

The first step of a web application usually, but not always, involve collecting some kind of data from the user. Traditionally, this was always handled using a simple HTML form. The user would type some information into some form fields, press a Submit button, and wait for the results.

Sending the Web Server a Request

In order for the web server to spring into action and execute a server program, the web browser needs to package up the user data an issue an HTTP request to the web server.

An HTTP request consists of the URL for the package or script that the user wishes to access, form data (if entered), and any additional header info (browser information, length and type request). A request is typically generated by the browser.

Executing the Server Script (or program)

An important function of the web server is that of passing a request to a specific script, or program, to be processed. The web server first determines which type of operating environment it needs to load by looking at the extension of the requested file (or the directory the file is located in). This is done through mapping. When a web server is configured, it is told how to handle specific file types. For example, typically anything in the cgi-bin directory will be treated as a CGI script, or anything with a .jsp extension will be treated as a JavaServer Page.

Once the web server determines that type of the requested file, it then loads any required runtime environment for the file to be executed. For example, if a CGI program were written in Perl, the web server would create a new process and load the Perl interpreter into it. For some types of programs it is not necessary to load a separate runtime environment. This is dependent on the web server and the technology being used. Either way, the web server fulfills its responsibility by directing the request to the right place.

Returning the Result to the Browser

The final step in a web application is to make some kind of response to the operation and return that to the browser. Generally speaking, the server script specifies the content type

and then writes the response to an output stream. When the web browser receives the response, it will first look at the response header and determine the mime type so that it knows how to render the data. The most common content type is "text/html", but the server can return XML, unformatted text, GIFs and even streamed audio.

3.3 Sever Side Programming Technologies

A server-side web program is the same as any other program with a few important exceptions. To make a program accessible to a web server, it must possess the following characteristics:

- The program should be able be invoked by the web server. When a user issues a request from a web browser, the web server has to be able to locate and execute the requested program.
- There must be a way in which the web server passes any form data to the program. When the web server invokes the program, it needs a way to pass in the HTTP request.
- Once the program is invoked, there has to be a standard entry point.
- After the program has processed the input data, it has to package up the result and send them back to the web server that will, in turn, send them back to the web browser. The exact division of responsibility may be blurred in some servers, but a web server in our sense just talks HTTP.

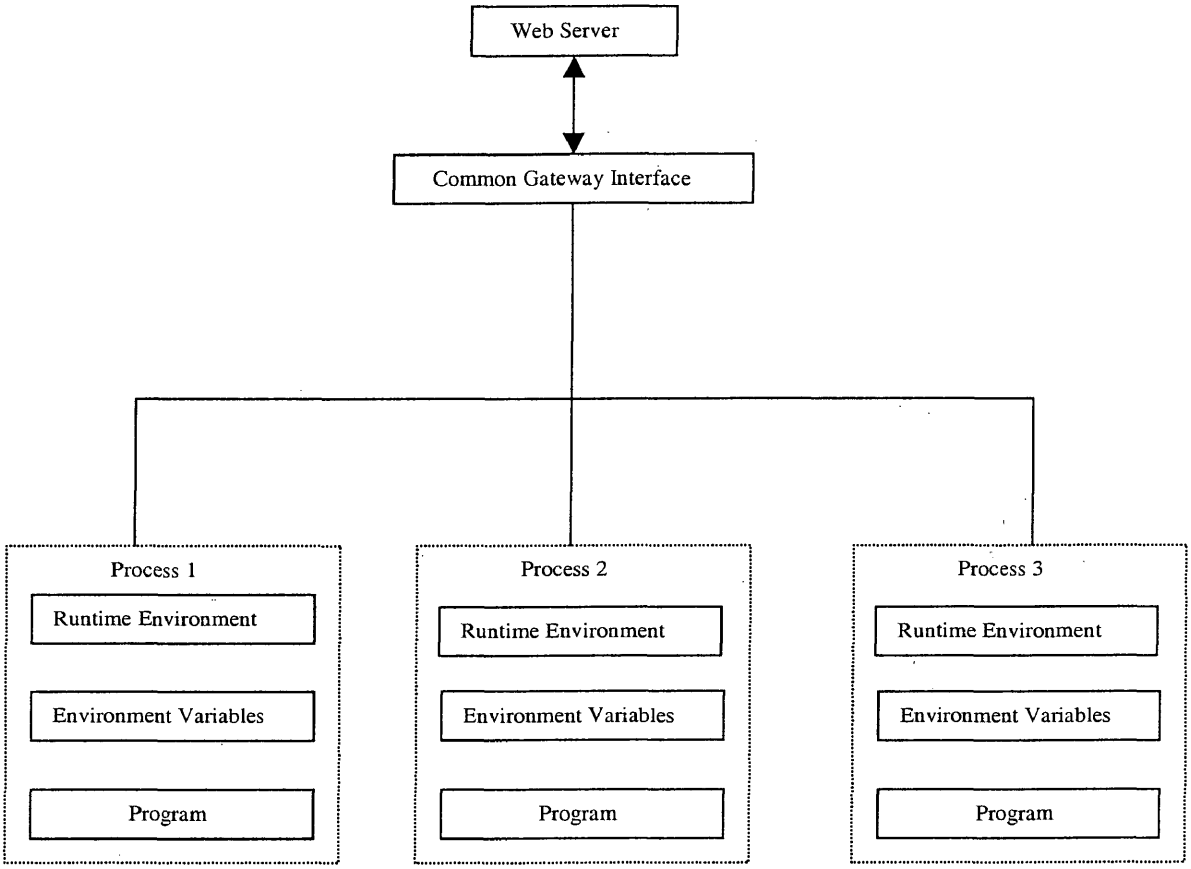
A few years back, the only real solution for bringing dynamic data to the web was Common Gateway Interface (CGI). CGI programs provided a relatively simple way to create a web application that accepts user input, queries a database, and returns some

results back to the browser. Both Microsoft and Netscape developed proprietary APIs that could be used to develop in-process code to service web requests. The latest server-side web technologies being offered are Active Server Page (ASP) and Java Servlets and JSP. Each of these technologies are described and compared in more detail below.

Common Gateway Interface (CGI)

Being the most common of the sever-side web technologies, just about every web server in existence today provides support for CGI programs. A CGI program can be written in just about any language, though the most popular language for CGI programming is Perl. Web Server implementing CGI act as a gateway between the user request and the data that it requires. It does this by first creating a new process in which the program will be run (see figure below). It will then load any required runtime environments as well as the program itself. Finally, it will pass in a request object and invoke the program. When the program is finished, the web server will read the response from standard output.

The biggest disadvantage to CGI programming is that it doesn't scale well. Each time a request is received by the web server, an entire new process is created. Each process consists of its own set of environment variables, a separate instance of whichever runtime environment is required, a copy of the program, and an allocation of memory for the program to use. It's not hard to imagine what might happen to a server when a large number of requests are received simultaneously. The resources of the server would be taxed very heavily, potentially causing the server to crash.



Technologies like Fast CGI and Apache's mod_perl help here. They both address performance issues, Fast CGI, by sharing single instances of each CGI program, and mod_perl by interpreting and executing Perl scripts within the Apache web server.

Active Server Page (ASP)

The latest web technology from Microsoft combines HTML, scripting, and server-side components in one file called an Active Server Page (ASP). When the server receives a request for an ASP file, it will first look for the compiled page and then execute it. If the page has not yet been compiled, the server will go ahead and compile and run it. The result of the ASP file is a finished web page returned to the browser.

An Active Server Page can be written using HTML, Jscript, and VBScript. Through scripting, the Active Server Page can access server-side components. These components can be written in any language as long as it presents a COM (Microsoft's component specification) interface. An advantage to ASP files is that everything is executed on the server. This helps to ensure that the pages are browser-independent, limited only by what the server can do.

One real disadvantage to Active Server Pages is that they can only be used with a Microsoft web server (IIS, PWS) on a Microsoft operating system. There are ports to other platforms and web servers, but the lack of wide COM support reduces their effectiveness. The other problem is that the mix of script and HTML, basically two sets of information threaded together, can become a maintenance nightmare.

Java Servlets

Servlets are Java technology's answer to Common Gateway Interface programming. They are programs that run on a Web server, acting as middle layer between a request coming from a Web browser or other HTTP client and databases or applications on the HTTP server. Their job is to:

- Read any data sent by the user. This data is usually entered in a form on Web page, but could also come from a Java applet or a customer HTTP client program.
- Look up any other information about the request that is embedded in the HTTP request. This information includes details about browser capabilities, cookies, the host name of the requesting client, and so forth.

- Generate the results. This process may require talking to a database, executing an RMI or CORBA call, invoking a legacy application, or computing the response directly.
- Format the results inside a document. In most cases, this involves embedding the information inside an HTML page.
- Set the appropriate HTTP response parameter. This means telling the browser what type of document is being returned, setting cookies and caching parameters, and other such task.
- Send the document back to the client. This document may be sent in text format, binary format, or even in a compressed format like gzip that is layered on top of other underlying format.

In principle, servlets are not restricted to Web or application servers that handle HTTP requests, but can be used for other types of servers as well. For example, servlets could be embedded in mail or FTP servers to extend their functionality.

The Advantages of Servlets over CGI

Java servlets are more efficient, easier to use, more powerful, more portable, safer, and cheaper than traditional CGI and many alternative CGI-like technologies.

Efficient

With traditional CGI, a new process is started for each HTTP request. If the CGI program itself is relatively short, the overhead of starting the process can dominate the execution time. With servlets, the Java Virtual Machine stays running and handles each request using a lightweight java thread, not a heavyweight operating system process. Similarly, in

traditional CGI, if there are N simultaneous requests to the same CGI program, the code for the CGI program is loaded into memory N times. With servlets, however, there would be N threads but only a single copy of the servlet class. Finally, when a CGI program finishes handling a request, the program terminates. This makes it difficult to cache computations, keep database connections open, and perform other optimizations that rely on persistent data. Servlets, however, remain in memory even after they complete a response, so it is straightforward to store arbitrarily complex data between requests.

Convenient

Servlets have an extensive infrastructure for automatically parsing and decoding HTML form data, reading and setting HTTP headers, handling cookies, tracking session, and many other such high-level utilities.

Powerful

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

Portable

Servlets are written in Java programming language and follow a standard API. Consequently, servlets written for any server can run virtually unchanged on other server.

Secure

One of the main sources of vulnerabilities in traditional CGI program stems from the fact that they are often executed by general-purpose operating system shells. So the CGI programmer has to be very careful to filter out characters such as backquotes and semicolons that are treated specially by the shell. This is harder than one might think, and weaknesses stemming from this problem are constantly being uncovered in widely used CGI libraries. A second source of problems is the fact that some CGI programs are processed by language such as c, c++, which do not automatically check array or string bounds. So programmers who forget to do this check themselves open their system up to deliberate or accidental buffer overflow attack. Servlets suffer from neither of these problems. Even if a servlet executes a remotes system call to invoke a program on the local operating system, it does not use a shell to do so. And of course array bounds checking and other memory protection features are a center part of the java programming language.

Access to Enterprise Java API

Since servlets are an extension of the Java platform, they can access all of the Java APIs. A Java servlet can send and received email, invoke methods of remote objects using RMI or CORBA, obtain directory information using the JNDI package, make use of an Enterprise JavaBean, or any other part of the java platform.

JavaServer Pages

JavaServer Page (JSP) is similar to Microsoft's Active Server Page. A JavaServer Page contains HTML, Java code, and JavaBean components. JSP provides a way to embed

components in a page, and to have them do their work to generate the page that is eventually sent to the client. When a user requests a JSP file, the Web server will first generate a corresponding servlet, unless one already exists. The Web server then invokes the servlet and returns the resulting content to the web browser.

JSP technology enables us to mix regular, static HTML with dynamically generated content from servlets. Many Web pages that are built by CGI programs are primarily static, with the parts that change limited to a few small locations. But most CGI variations, including servlets make us generate the entire page via our program, even though most of it is always the same. JSP lets us create two parts separately. JavaServer Pages provide a powerful and dynamic page assembly mechanism that benefits from the many advantages of the Java platform.

Chapter 4 Choose and Installation of Project Related Software

In this chapter, we discuss the criteria we use to choose the software for this project and how to install these software.

4.1 Choose of the hosting operating system

Based on following consideration, Microsoft Windows 2000 Professional is chosen as the hosting operating system for this project:

- *Reliability.* Windows 2000 Professional is the most reliable Windows ever. It is up to 30 percent faster and, according to NTSTL tests, and 13 times more reliable than Windows 98. Based on a comparative stress test, ZD Labs concluded that the reliability of Windows 2000 Professional far exceeds that of Windows 98 and Windows NT Workstation 4.0.

- *Performance.* The Windows 2000 Professional operating system performs significantly better than Windows 95 and Windows 98, and is comparable to Windows NT 4.0 in tests running the most popular business applications, according to ZD Labs.

- *Availability.* The Advanced Server and Data center Server editions of the Windows 2000 Server Family let us increase our system's availability using the clustering technologies included with the operating system, which let us couple servers together to support specific tasks.

- *Hardware support.* Windows 2000 supports a wide range of hardware and peripherals.

- *Scalability.* Windows 2000 Server Family includes three versions, each able to reliably and affordably handle larger loads. We can start with Windows 2000 Server, and move up as needed.

System requirements to run Windows 2000:

- 133 MHz or higher Pentium-compatible CPU.

- 64 megabytes (MB) of RAM recommended minimum; more memory generally improves responsiveness.

- 2GB hard disk with a minimum of 650MB of free space.

- Windows 2000 Professional supports single and dual CPU systems

4.2 Choose of the DBMS

Based on following consideration, Microsoft SQL Server is chosen as the DBMS for this project:

- *Easy installation.* Microsoft SQL Server can be easily installed on Windows 2000 professional system.
- *Performance.* According to the result of the competitive usability test conducted by AIR (American Institute for Research), The database administrators who used SQL Server 7.0 performed better on all measures than the database administrators who used Oracle 8i.

- *Speed.* According to a newly published, record-setting TPC-C result, Microsoft SQL Server™ 2000 is the fastest database in the world.
- *Cost.* Three independent studies have found that SQL Server has a lower total cost of ownership than Oracle

System requirement:

- Personal computer with an Intel Pentium or compatible 166-megahertz (MHz) or higher processor.
- Windows 2000 Professional, with SP5 or later (for this project).
- 64MB memory (standard edition).
- 130 MB hard disc space (for typical installation).
- CD-ROM drive.
- VGA or higher-resolution monitor.
- Microsoft Internet Explorer version 5.0 or later

4.3 Choose of the Web Server

For this project, we choose Apache as our web server because Apache is a freely available, easy to use, yet fully functional Web Server. Apache is proving to be more popular than servers being sold commercially by Internet powerhouses Microsoft and Netscape.

Apache is part of a tradition of freely available software that is made available by researchers working at institutions such as the National Center for Supercomputing Applications (NCSA) at the University of Illinois. It isn't designed to work with one single operating system, but has been ported to many different systems. It comes with

modules-which are mini-programs that perform common functions-and it can be custom configured and programmed for individual situations.

In the simplest terms, Apache for Windows works like this:

- The server thread receives a request for a file (or program) via a URL.
- The thread locates the correct file (or runs the program).
- The server returns the result in the form of the file that was requested (or the output of the program that was executed.)

Apache is designed to run on Windows NT 4, but is will also run on Windows 95, 98, and 2000. In the course of writing this project, all four platforms were tested.

System requirement:

- Windows 2000 Professional (for this project).
- TCP/IP networking installed.
- Pentium II 266 CPU (for this project).
- 64MB RAM (for this project).

Download and Installing Apache:

The latest version (for this project, version 1.3) of Apache for Window32 can be downloaded from www.apache.org. For this project, because we would like to run Apache as a Windows NT service, we install Apache in the C:\Apache Group\Apache directory.

Configuring the Apache Web Server

In Apache, a directive is a command that tells Apache how to performs a specific function. As Webmaster, we tell Apache what to do by adding, deleting, or combining directives and their corresponding argument. For this project, we use Notepad to edit the

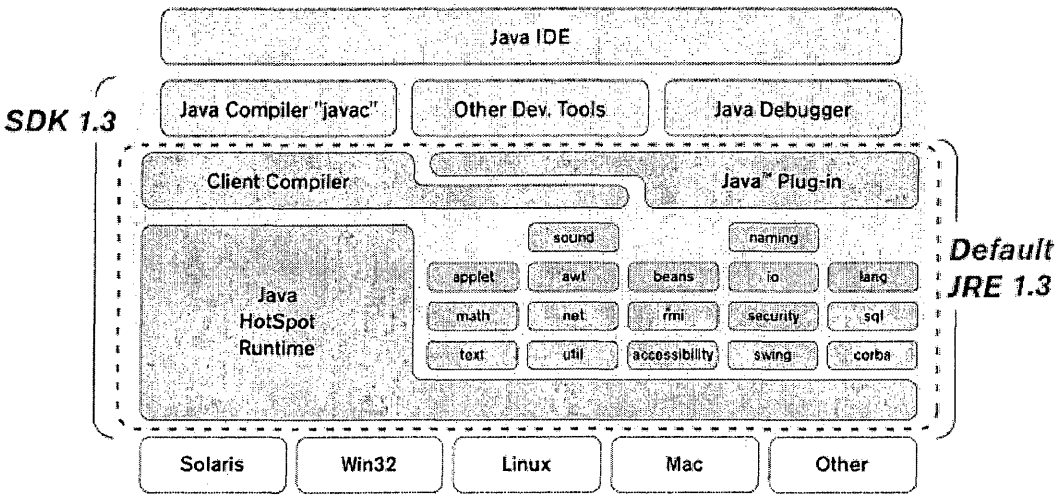
essential httpd.conf file under C:\Apache Group\Apache \conf directory, change the active directives and their arguments as following:

- ServerTyype standalone
- Port 80
- ServerAdmin binshu@hotmail.com
- Serveroot c:/Apache Group/Apache
- DocumentRoot "c:/Apache Group/Apache/htdocs"
- ScoreBoardFile logs/apache_status
- P idFile logs/httpd.pid
- Timeout 300
- ThreadsPerChild 50
- MaxRequestsPerChild 0
- KeepAliveTimeout 15
- KeepAlive On
- UserDir "c:/Apache Group/Apache/users/"
- DirectoryIndex index.html
- AccessFileName .htaccess
- TypesConfig conf/mime.types
- DefaultType text/plain
- HostnameLookups Off
- ErrorLog logs/error.log
- CustomLog logs/access.log common

- Alias /icons/ "c:/Apache Group/Apache/icons/"
- ScriptAlias /cgi-bin/ "c:/Apache Group/Apache/cgi-bin/"

4.4 Download and install Java 2, Standard Edition, Version 1.3.0

The Java 2 SDK is a development environment for building applications, applets, and components that can be deployed on implementations of the Java 2 Platform. The Java 2 SDK includes tools useful for developing and testing programs written in the Java programming language and running on the Java platform. Version 1.3 of the Java 2 SDK includes development tools, runtime environment, additional libraries, and offers significant improvements in functionality and performance over previous versions.



The whole Java 2 SDK version 1.3 is downloaded from http://java.sun.com and installed in C:\jdk1.3 directory.

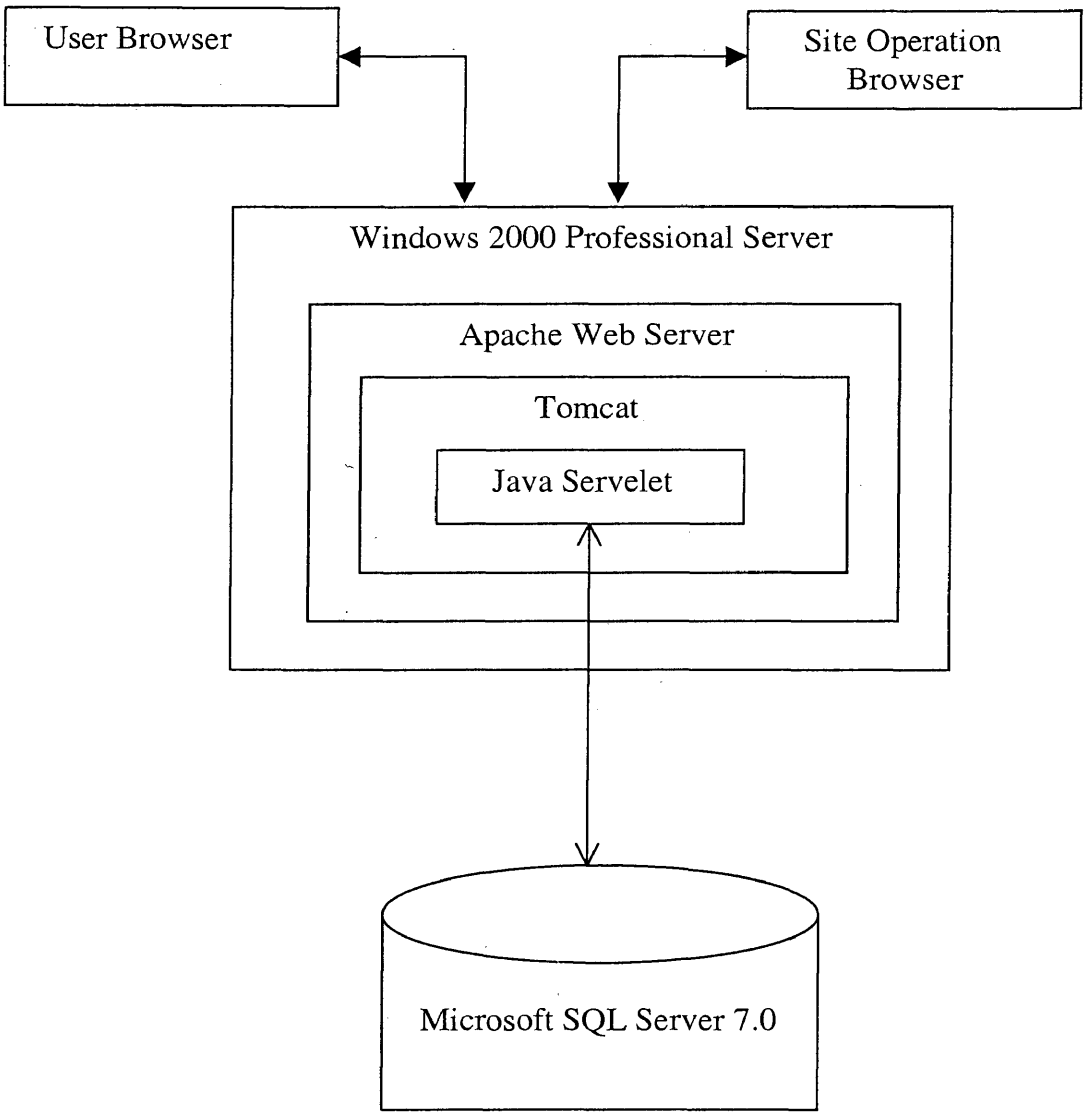
4.5 Download and install Tomcat

Tomcat is the official reference implementation of the Java Servlet 2.2 and JavaServer Page 1.1 technologies. It's developed under the Apache license in an open and

participatory environment. In this project, we use Tomcat version 3.2.2 as our servlet container. Tomcat version 3.2.2 can be downloaded from http://jakarta.apache.org/builds/jakarta-tomcat/release/v3.2.2

Chapter 5 Design and implementation of project

In this project, we will simulate the whole process of web business by implementing an on-line air ticketing system using Java servlet and JSP. As usual, our design includes front end user interface design, middle layer software design and back end database design. Following graph represents the general structure of our system.



5.1 Database design

The database we design for this project includes four tables: customer, ticket, transactions, and orders. The attributes and properties of these four table are as following:

Customer table:

We use customer table to store each customer's personal information, which includes customer login name, password, customer's first name, customer's middle name, customer's last name, phone number, e-mail address, credit card number, credit card type (the three credit card type we accept are: master card, visa card, and American express), and credit card billing address. Field customer_id serves as the primer key of the customer table. Table 5.1 shows attributes of the table:

Table 5.1

Field Name	Data Type	Size
customer_id	char	10
password	char	10
first_name	char	10
middle_name	char	10
last_name	char	10
address_1	char	50
address_2	char	50
city	char	10
state	char	3
zip	char	10

phone	char	15
email	char	20
card_type	char	2
card_number	char	19
exp_date	datetime	8

Orders table

Orders table is used to stored the order information, which includes order id, customer id of the customer who makes the order, when the order is made, order status (shipped or under processing) and the total dollar amount of the order. The primer key of this table is order id, which is the system time in microsecond when the specific order is made. In our project, we assume this system time is unique for each order. Table 5.2 shows the attributes of the table.

Table 5.2

Field Name	Data type	Size
order_id	char	20
customer_id	char	20
order_date	datetime	8
status	char	2
total_amount	float	8

Transaction table

Transactions table is used to stored transaction information, which includes the customer id of the customer who made the order, the ticket id of the tickets that are bought, the quantity of tickets that are bought, the unit price of the ticket, and total amount of the transaction. Field order_id and field ticket_id serve as the combined primer key of this table. Table 5.3 shows the attributes of the table.

Table 5.3

Field Name	Data type	Size
order_id	char	20
ticket_id	char	10
quantity	int	4
unit_price	float	8
amount	float	8

Ticket table

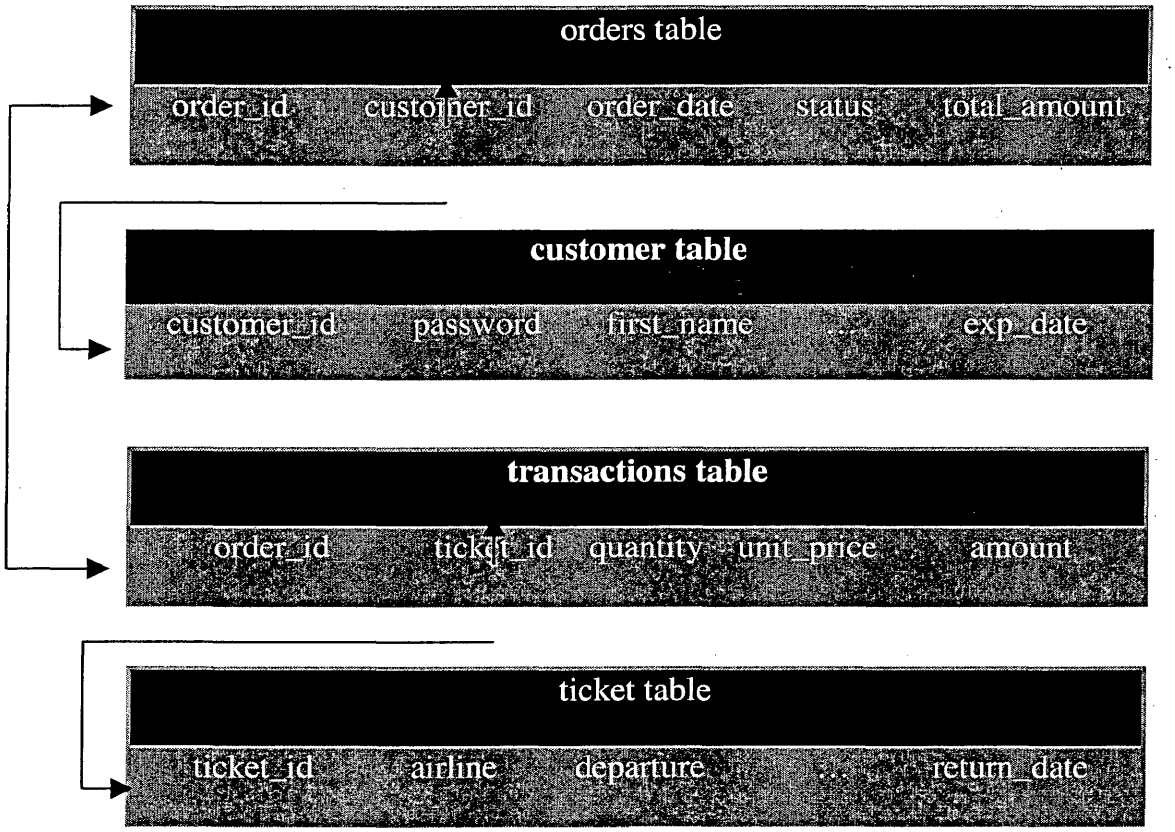
Ticket table is used to stored ticket information, which includes the air company name, the departure airport, the arrival airport, the departure flight departure time, the departure flight arrival time, the return flight departure time, the return flight arrival time, the dates of leave and return, and the flight numbers of departure and return flight. Table 5.4 shows the attributes of the table.

Table 5.4

Field Name	Data Type	Size
ticket_id	char	20
airline	char	20
departure	char	3
arrival	char	3
departure_airport	char	50
arrival_airport	char	50
flight1	char	20
departure_time1	datetime	8
arrival_time1	datetime	8
flight2	char	20
departure_time	datetime	8
arrival_time	datetime	8
price	float	8
depart_date	datetime	8
arrival_date	datetime	8

Database Diagram

The following diagram illustrates the connection and relationship between customer table, ticket table, transactions table and orders table.



5.2 Web page, JSP, and Java Servlet design

Home page

Figure 5.1, Figure 5.2, and Figure 5.3 show the design of home page. This home page is a gateway to our web application. We generally introduce the main function, characteristic, the advertisement and service of our online air ticketing system. From this page, customer can go to:

- Registration page.
- Sign in page.
- Search page.
- About us page

Figure 5.1

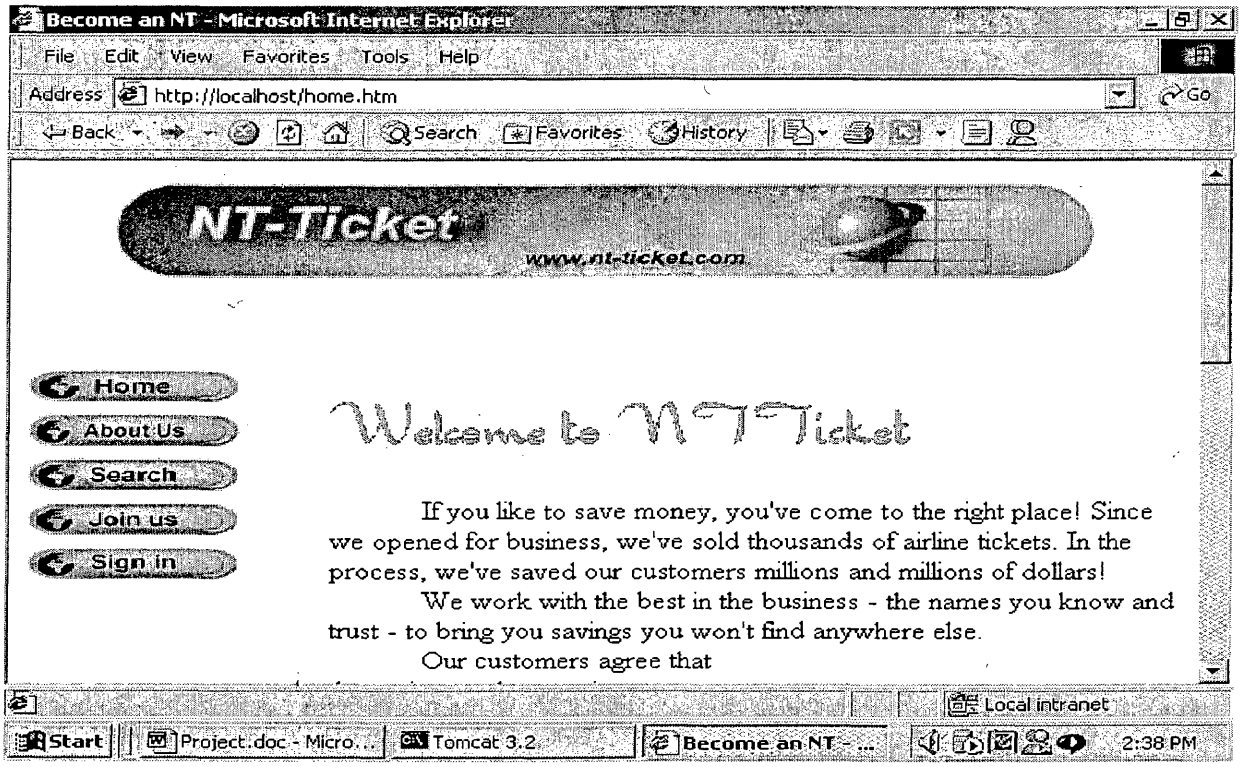


Figure 5.2

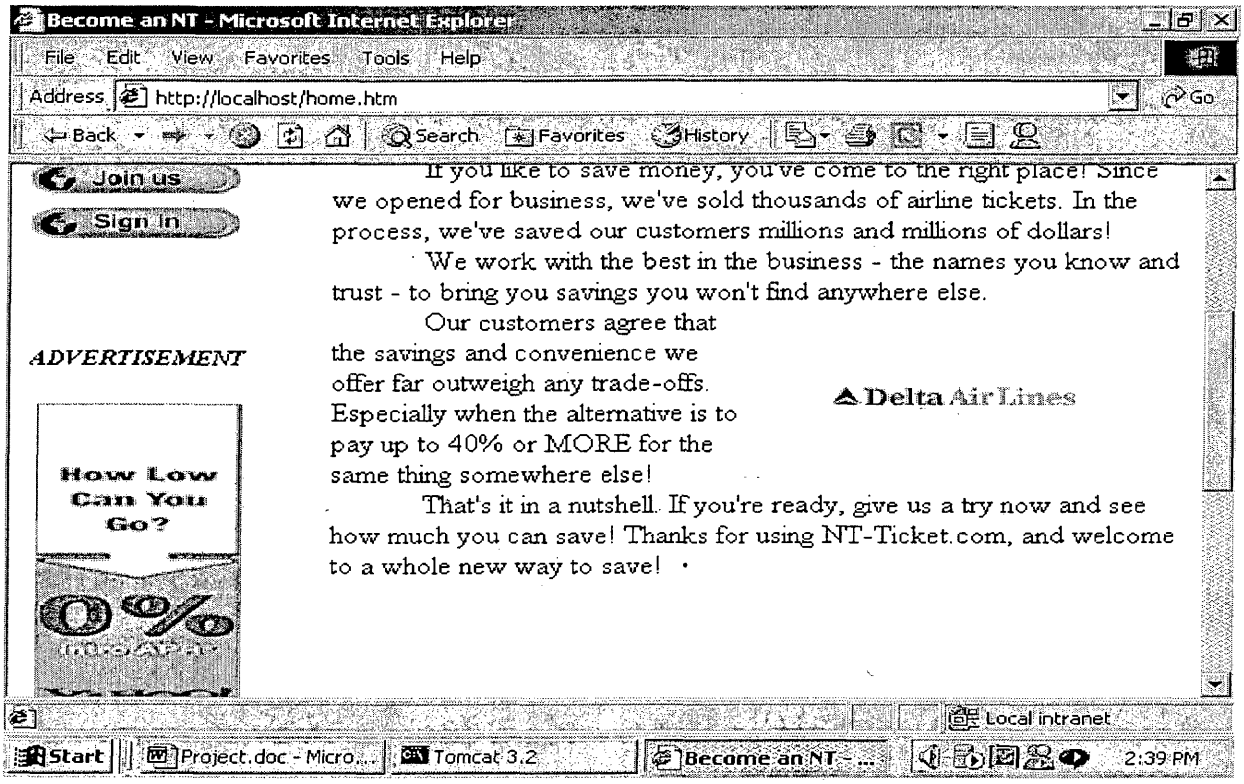
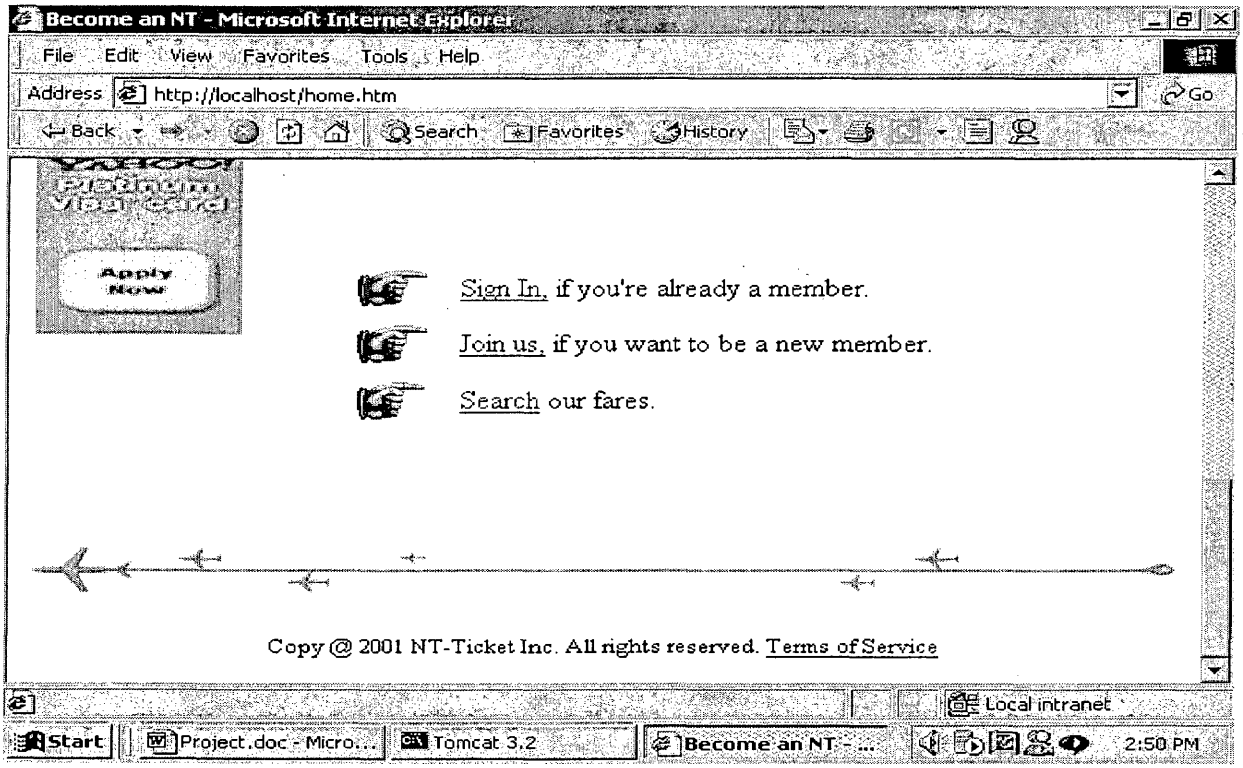


Figure 5.3



Page joinus.htm, servlet "register", registerok.jsp, and registererror.jsp:

Figure 5.4, Figure 5.5, and Figure 5.6 show the design of joinus.htm page.

Figure 5.4

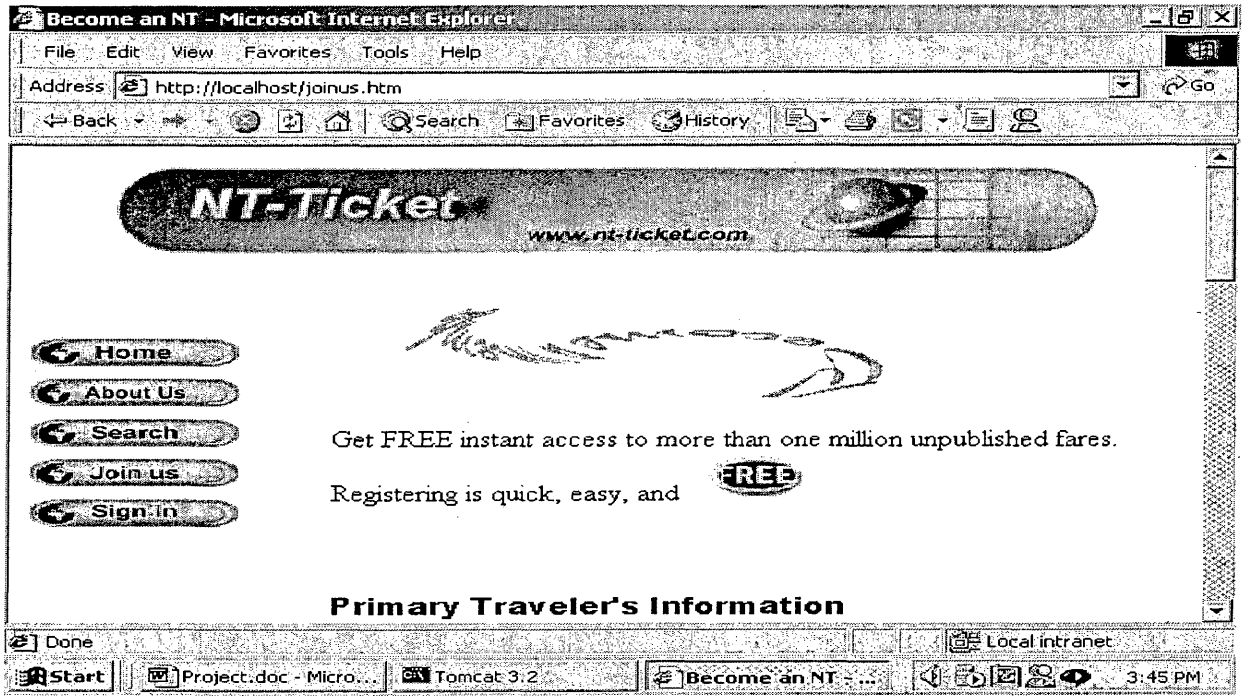


Figure 5.5

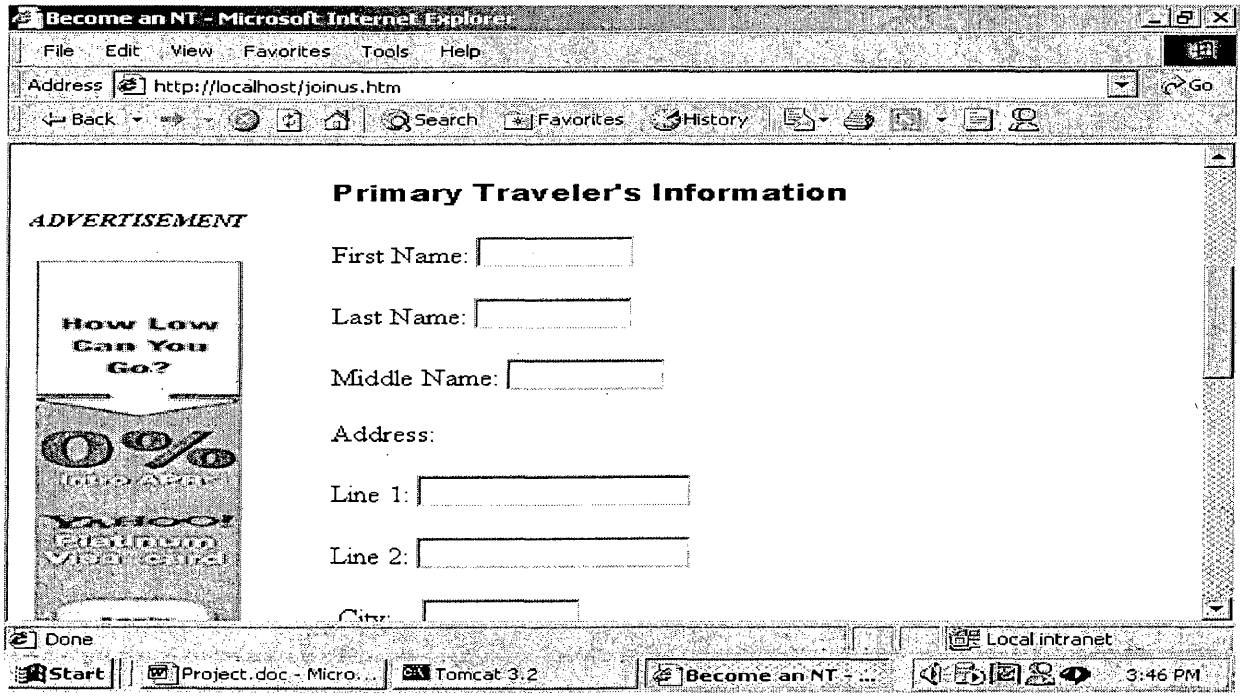
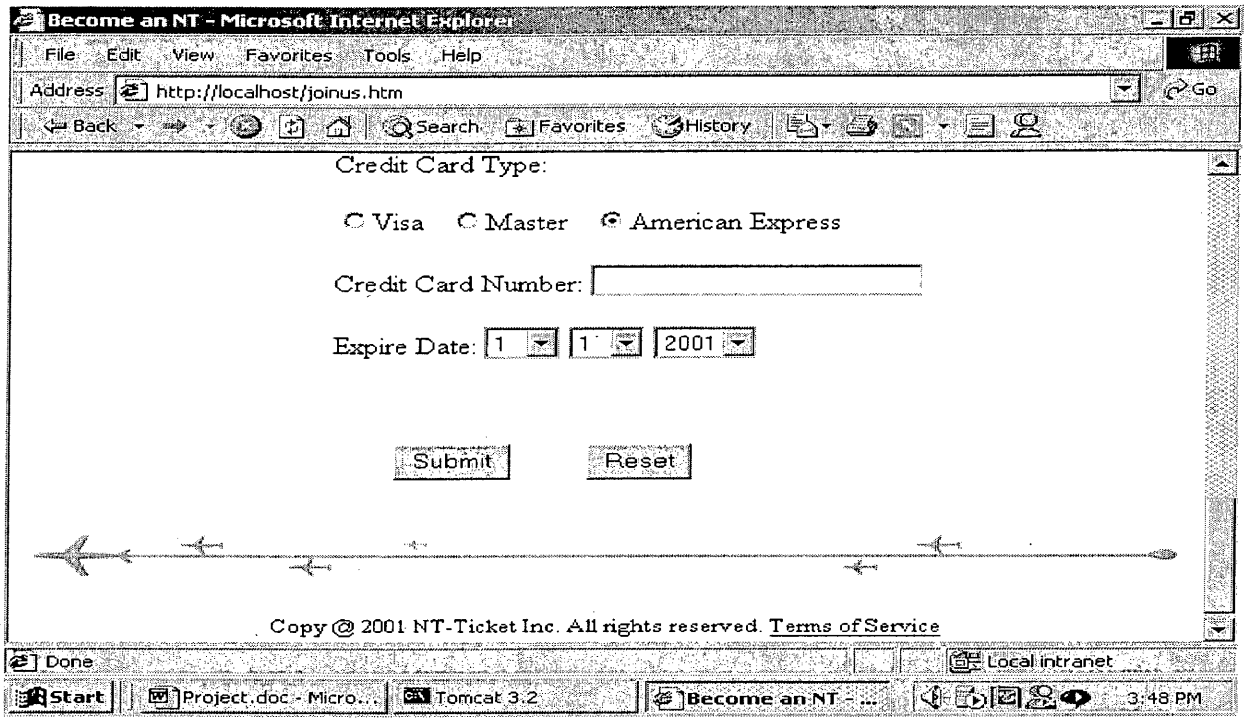


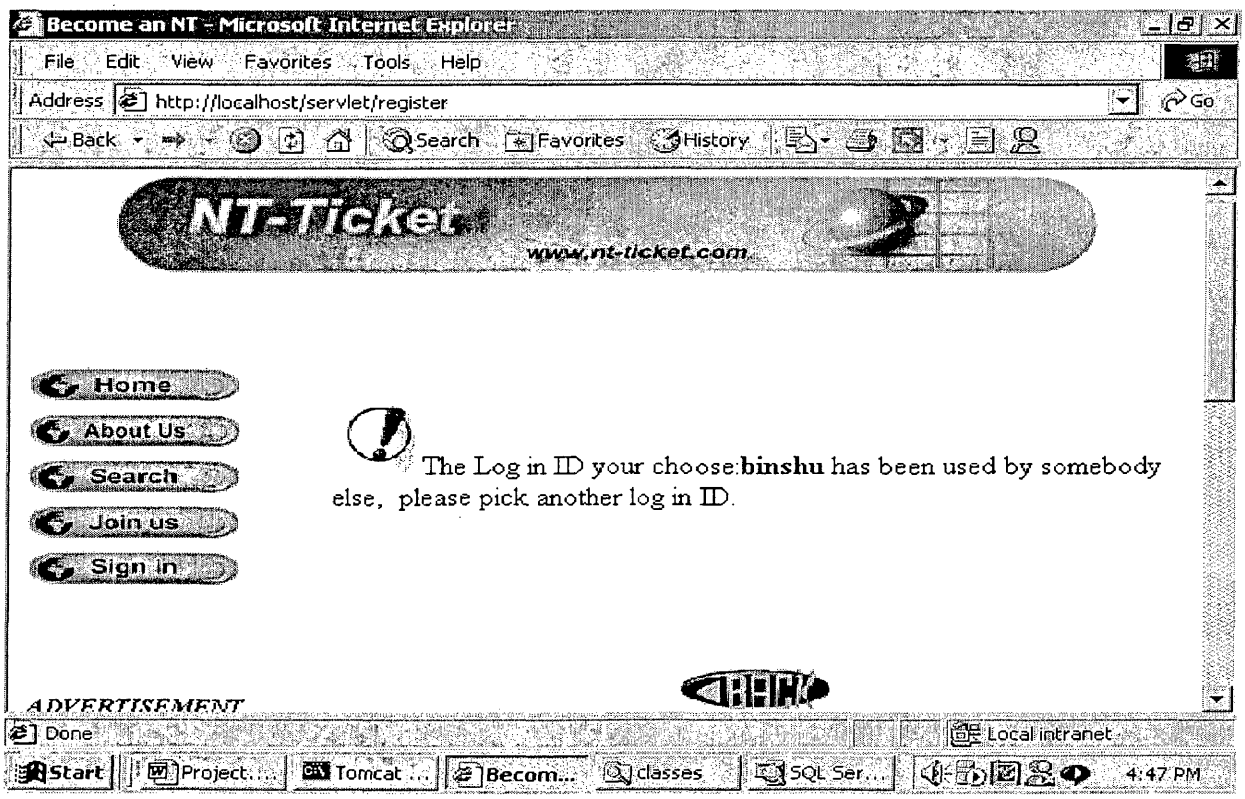
Figure 5.6



Page joinus.htm is the place where new customer can enter their personal information, such as e-mail address, payment method, billing address, etc. When the user clicks the "submit" button on the bottom of the page, the "register" servlet will be invoked. As the user's login name will serve as the primer key (customer_id field) of customer table, it must be unique. So the "register" servlet will first check the "customer" table to see if other customer has already used the same login name, if not, "register" servlet will enter the new customer's information into "customer" table, create a new "ShoppingCart" object for current session, and comes up with registerok.jsp, otherwise, registererror.jsp will be brought up. The source code of "register" servlet can be found in project attachments. Figure 5.7 shows the design of registerok.jsp, and Figure 5.8 show the design of registererror.jsp.

Figure 5.7

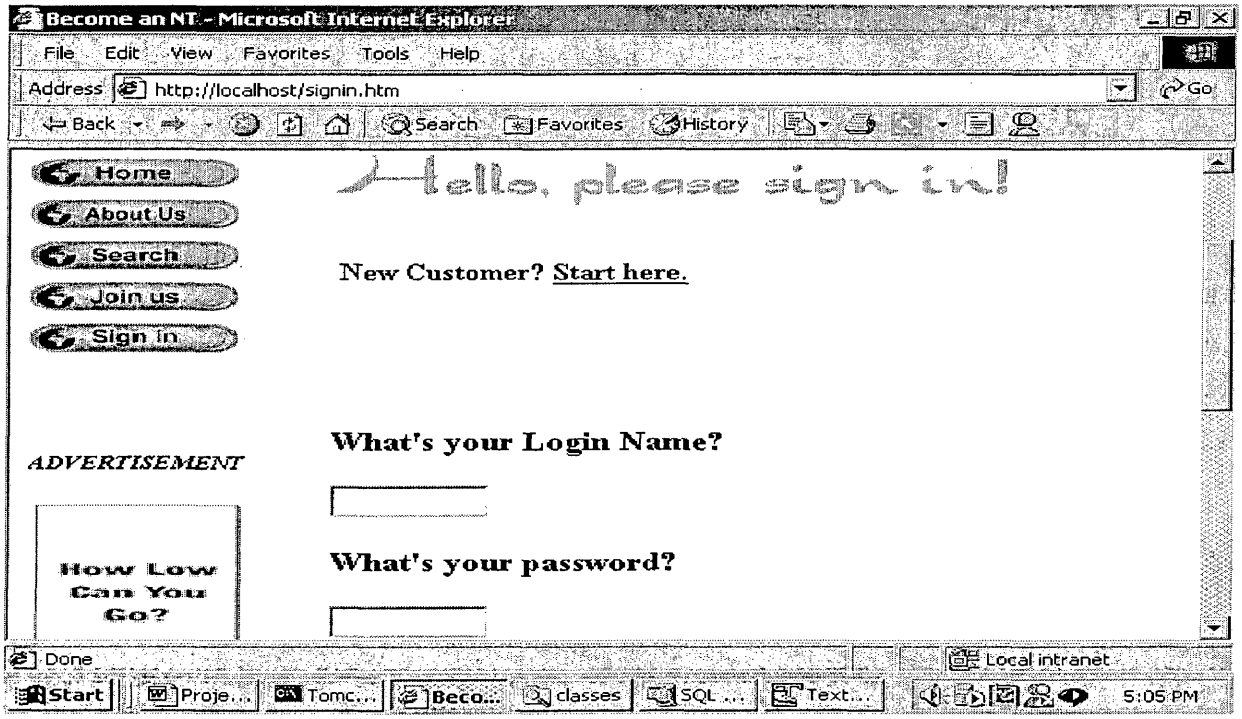
Figure 5.8



Page signin.htm, signinerror.jsp, and servlet "signin":

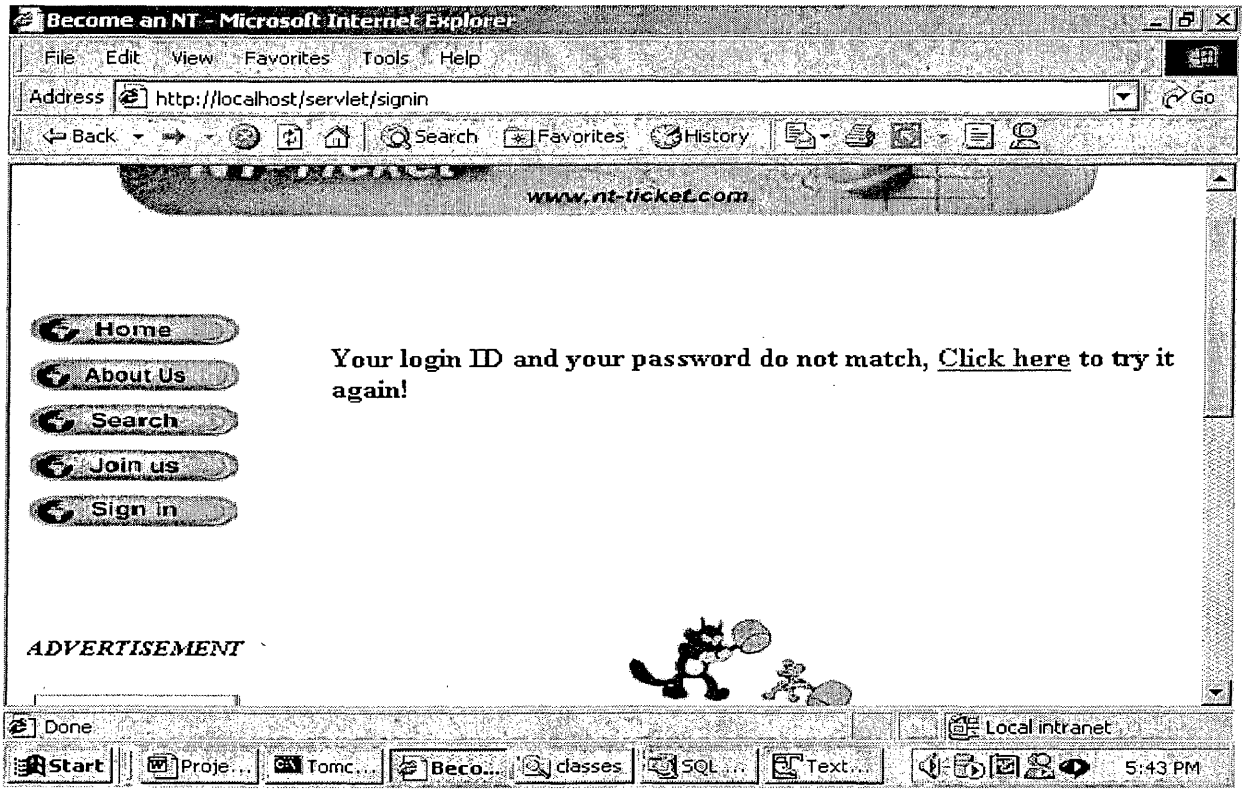
The page signin.htm is the place where current customers can log in using their login name and password. Figure 5.9 shows the signin.htm page.

Figure 5.9



When user clicks the "submit" button on the bottom of the signin.htm page, "signin" servlet is invoked. Servlet "signin" first checks "customer" table to see if login name and the password user enters match each other, if yes, it will create a "ShoppingCart" object for current session and come up with "Search" page, otherwise, "signinerror.jsp" will be invoked. The source code of "signin" servlet can be found in attachment. Figure 5.10 show the design of "signinerror.jsp". The source code of the "signin" servlet can be found in attachment.

Figure 5.10



Servlet "search"

Ours policy is that only a member can search our database for air fair. Servlet "search" is in charge of reinforcing this policy. Whenever a user wants to get access to our "search" page, for instance, the user clicks "search" button, "search" servlet is invoked. Servlet "search" first check if the user has already signed in as a member by checking if current session has a "ShoppingCart" object, if yes, "search" page will be brought up, otherwise, user will be brought to "signin.jsp".

searchpage.jsp

searchpage.jsp is the place where a member can search our air fair. Figure 5.11 and 5.12 show the design of searchpage.jsp.

Figure 5.11

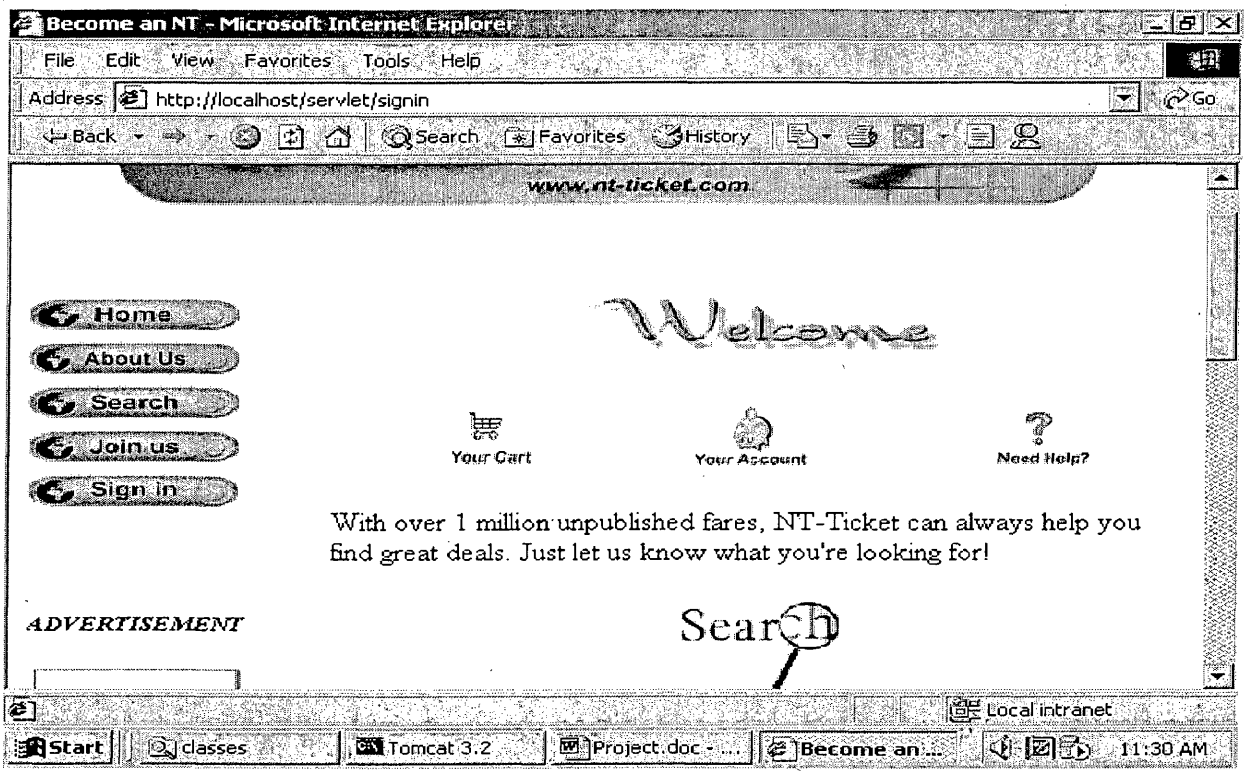
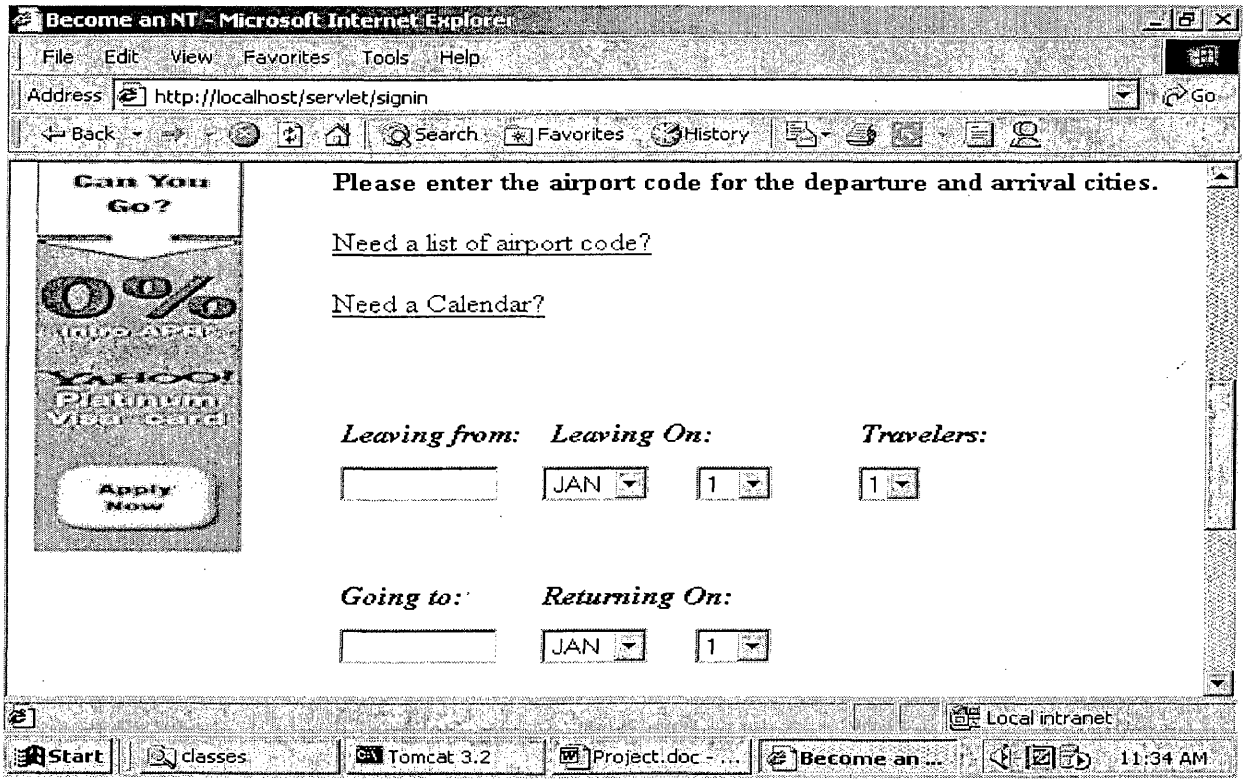


Figure 5.12



From "searchpage" page, a member can

- Search the air fair by entering itinerary information in the form contained in the page and click the submit button.
- Get his or her account information by clicking the "Your Account" icon.
- Get current items in his or her shopping cart by clicking the "Your Cart" icon.
- Get the help page by clicking "Need help?" icon.
- Get airport code list by clicking "Need a list of airport code?" link.
- Get a calendar by clicking "Need a calendar?" link.

searchok.jsp, searcherror.jsp, and "search" servlet

After the user enter his or her itinerary information into the form contained in searchpagejsp and clicks the "submit" button, "search" servlet will be invoked. Servlet "search" first check "ticket" table for all available flights that meet user's requirement, if it can get a result, servlet "search" will bring up searchok.jsp which will show up all the available flight information, otherwise, it will come up with searcherror.jsp which reminds the user that something might be wrong with his or her request. The source code of servlet "search" can be found in attachment. Figure 5.13 and figure 5.14 show the design of searchok.jsp. Figure 5.15 shows the design of searcherror.jsp.

Figure 5.13

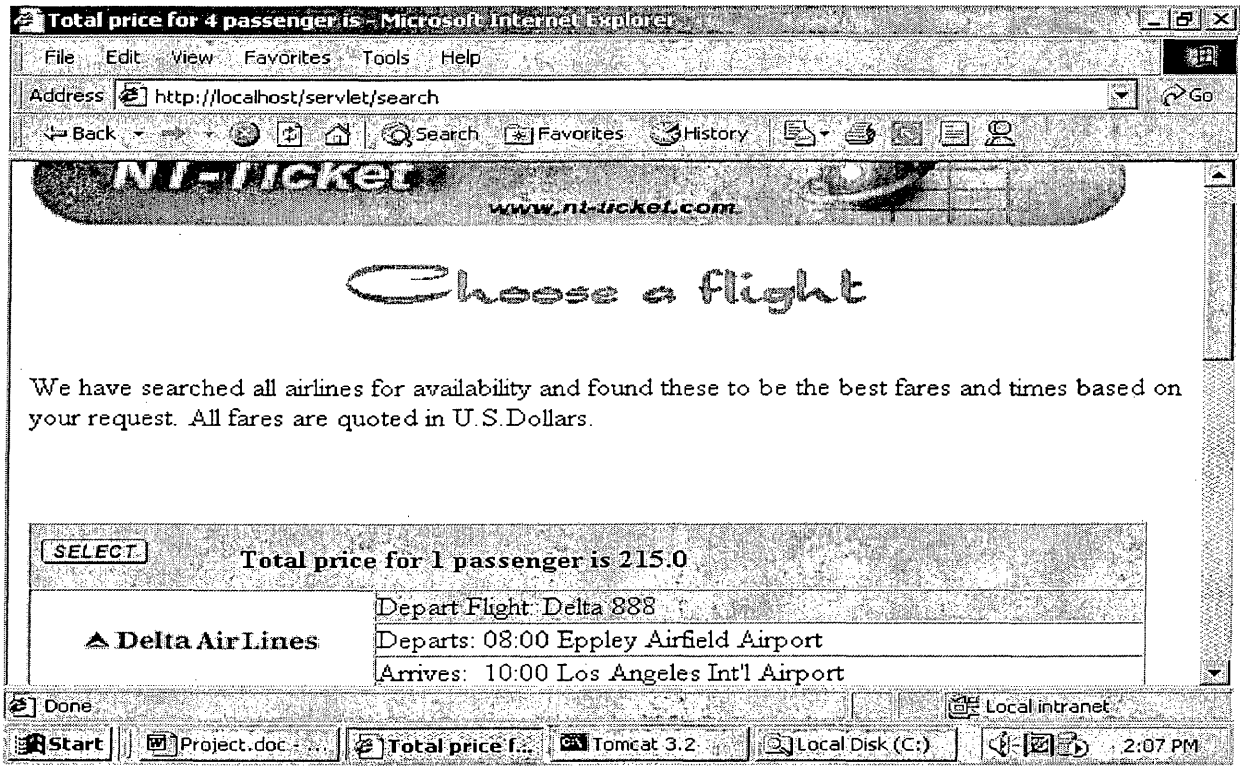


Figure 5.14

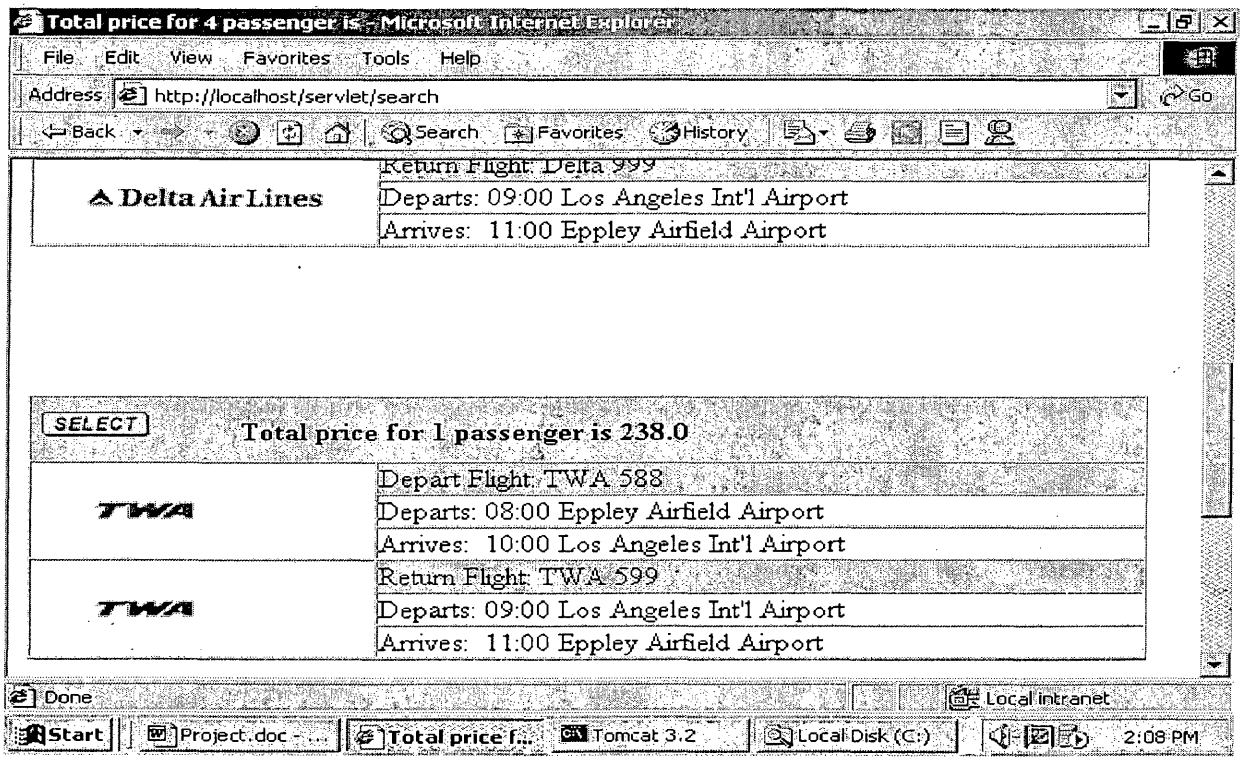
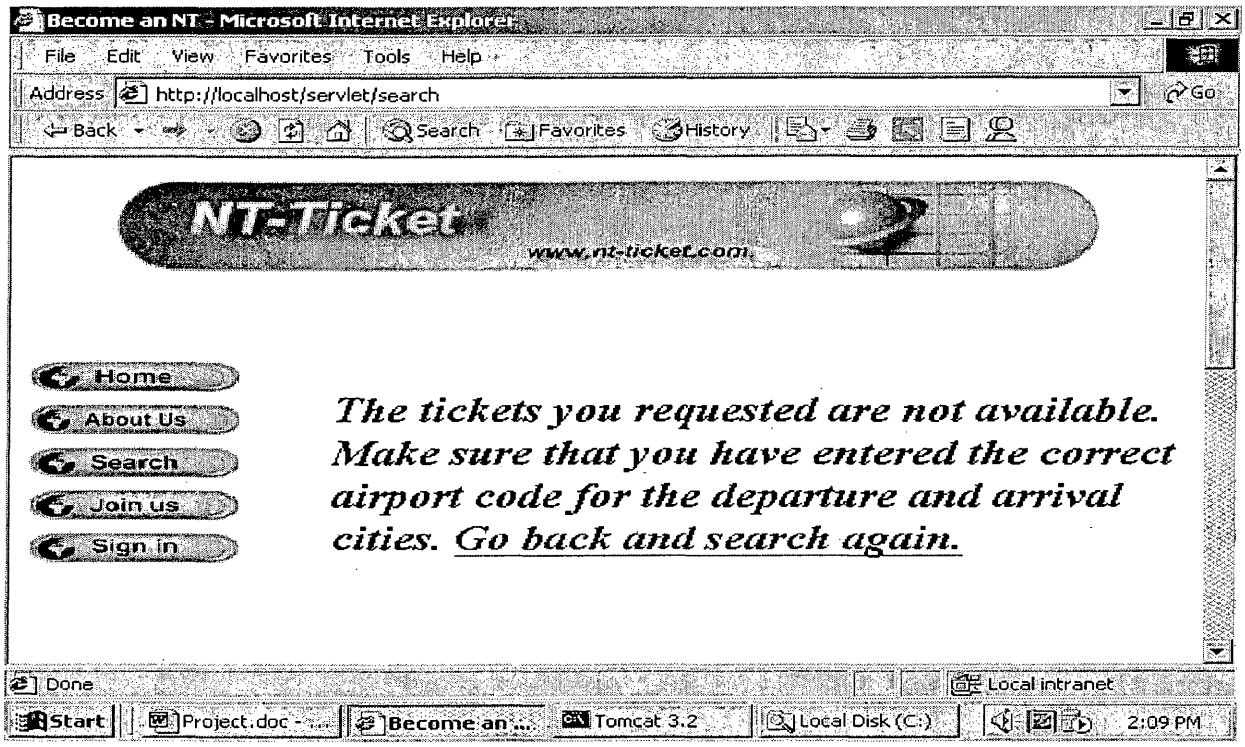


Figure 5.15



orderpage.jsp

While users are in searchok.jsp, they can click "select" icon to chose the ticket they want. This will bring users to orderpage.jsp. orderpage.jsp will first create an "TicketOrder" object that encapsulate the ticket's information, and then insert this object into "ShoppingCart" object in current session. Source code of orderpage.jsp can be found in attachment. Figure 16 and figure 17 show the design of orderpage.jsp.

Figure 5.16

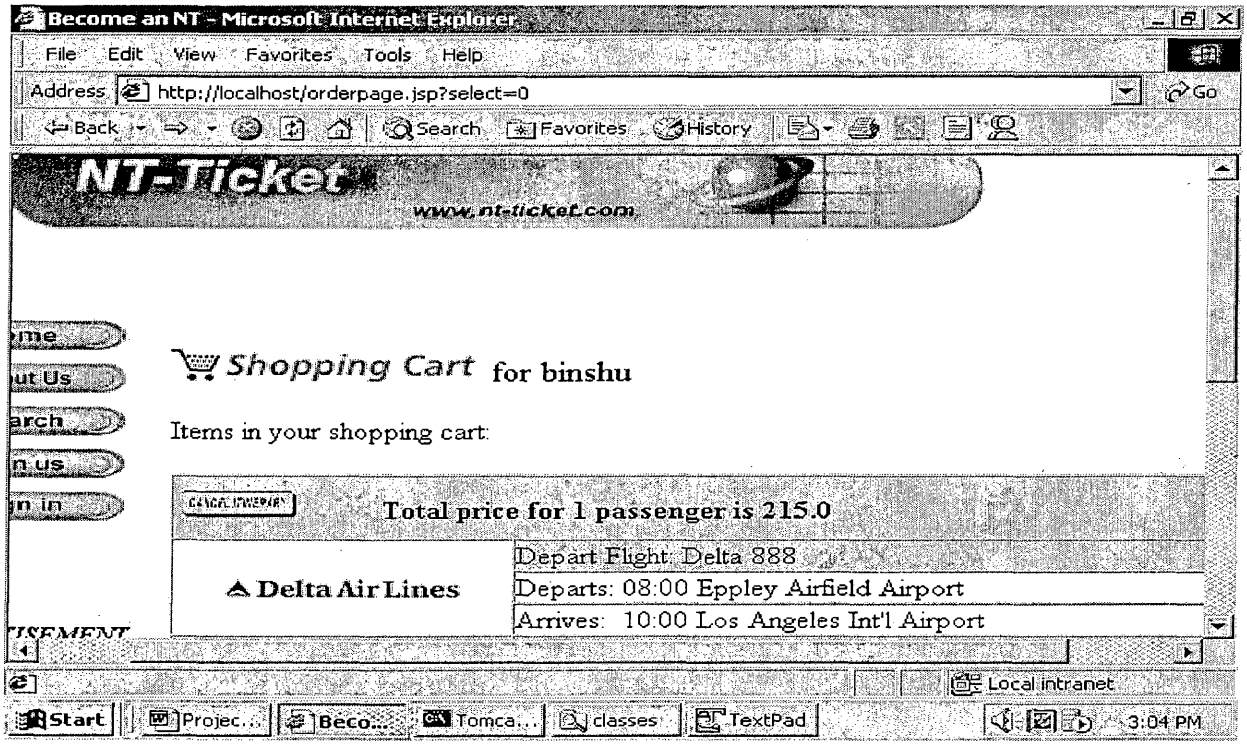
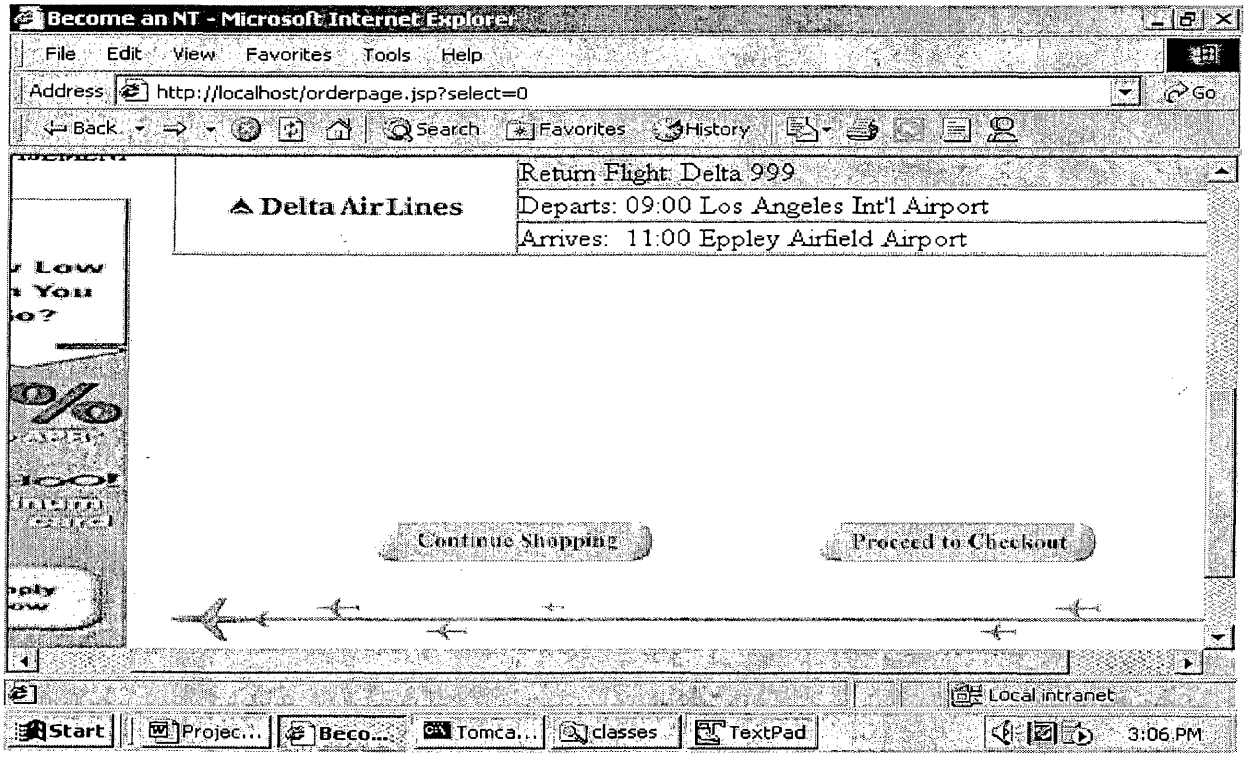


Figure 5.17



From orderpage.jsp, users can

- Delete a specific itinerary in shopping cart by click the "delete itinerary" icon shown up on that itinerary, this will bring users to a new orderpage.jsp.
- Go to searchpage page by clicking "Continue Shopping" button.
- Go to verifypage.jsp by clicking "Proceed to Checkout" button.

Servlet "update" and verifypage.jsp

When users are in orderpage.jps, they can click "Proceed to Checkout" button to go to verifypage.jsp. Figure 5.18, figure 5.19 and figure 5.20 show the design of verifypage.jsp.

Figure 5.18

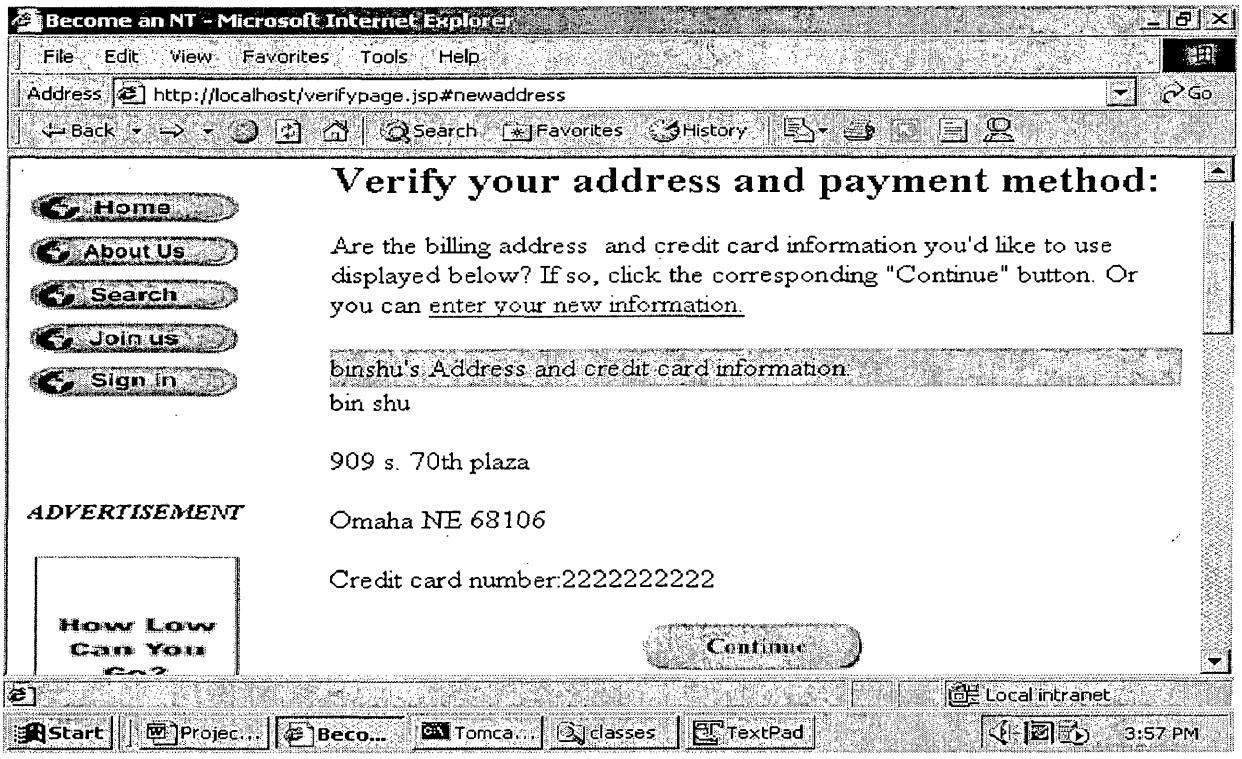


Figure 5.19

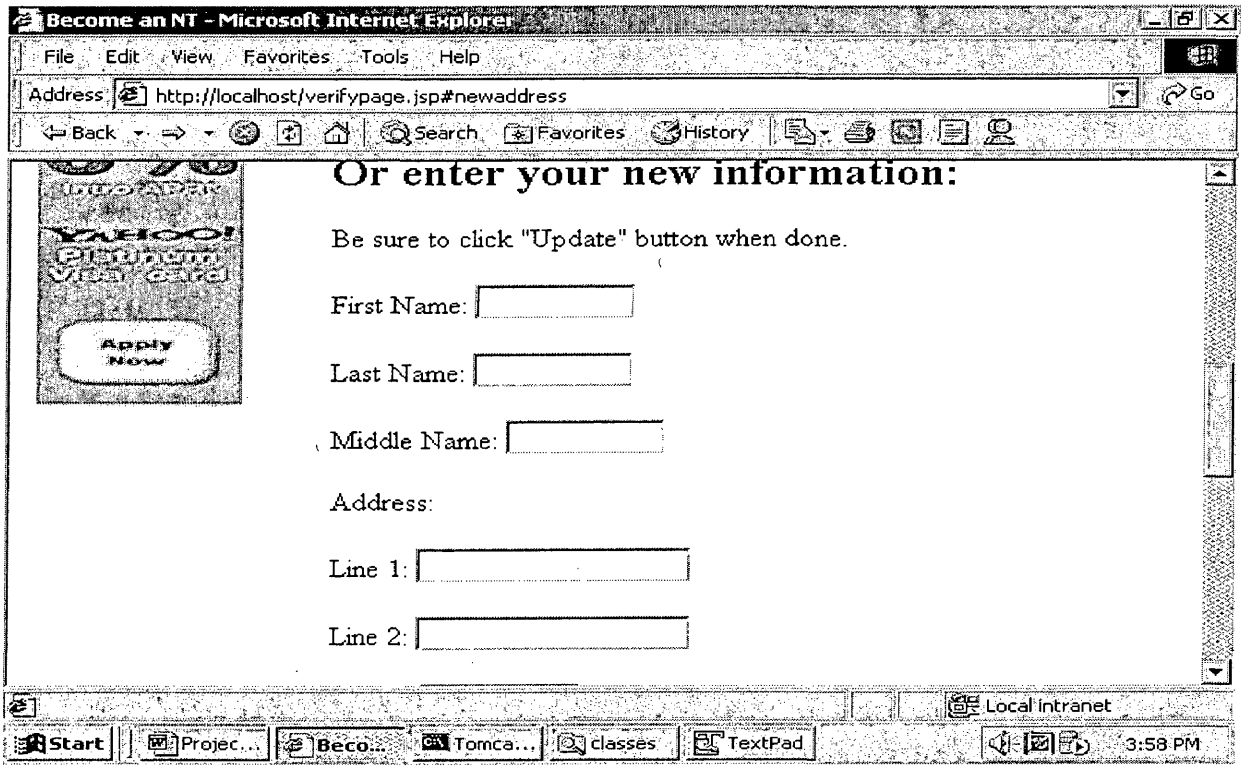
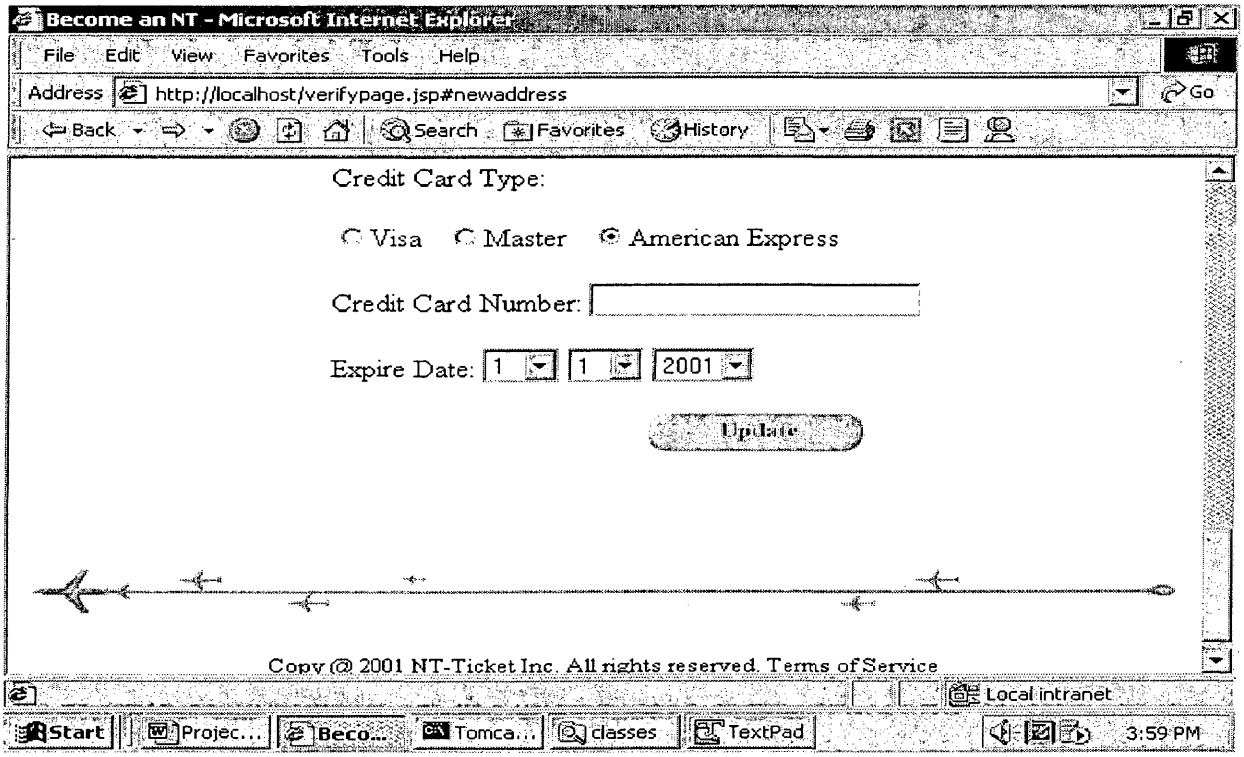


Figure 5.20



Basically, the first part of verifypage.jsp is the user's personal information. If this information haven't changed since the last purchase, user can then click "Continue" button proceed to checkout.jsp page. Otherwise, user can enter the current information into the form contained in verifypage.jsp and then click "Update" button located at the bottom of the page. This will invoke "update" servlet, which will in turn update the user's information in "customer" table according to the information in the form submitted by the user, and then bring up checkout.jsp page.

Servlet "confirmorder", checkout.jsp, and confirmpage.jsp

When users click "Update" or "Continue" buttons in verifypage.jsp, they will be brought to checkout.jsp. In checkout.jsp, users can have two choices:

- Click "Cancel Order" button to invoke "confirmorder" servlet to delete all the items in shopping cart and then go to searchpage.jsp.
- Click "Place Order" button to place order, which will in turn invoke "confirmorder" servlet to update "transactions" table and "orders" table, then delete all the items in shopping cart and go to confirmpage.jsp.

The source code of servlet "confirmorder" can be found in attachment. Figure 5.21 and figure 5.22 show the design of checkout.jsp. Figure 5.23 shows the design of confirmpage.jsp.

Figure 5.21

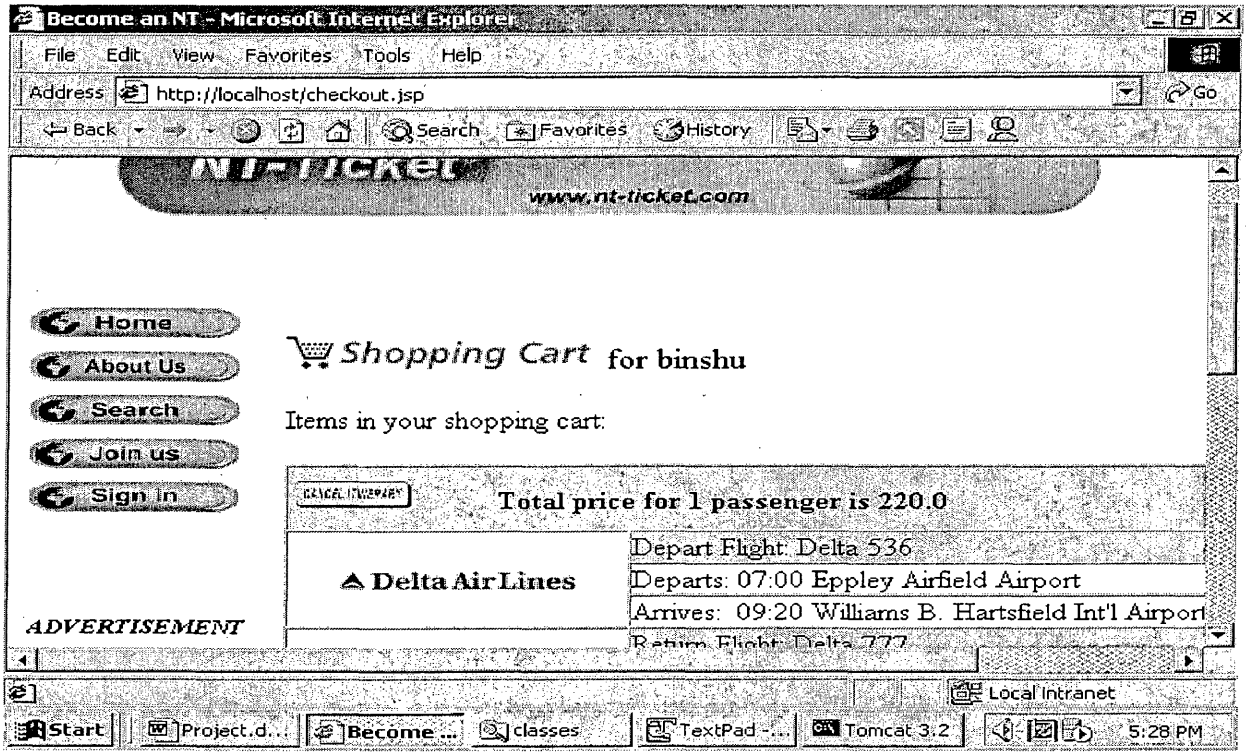


Figure 5.22

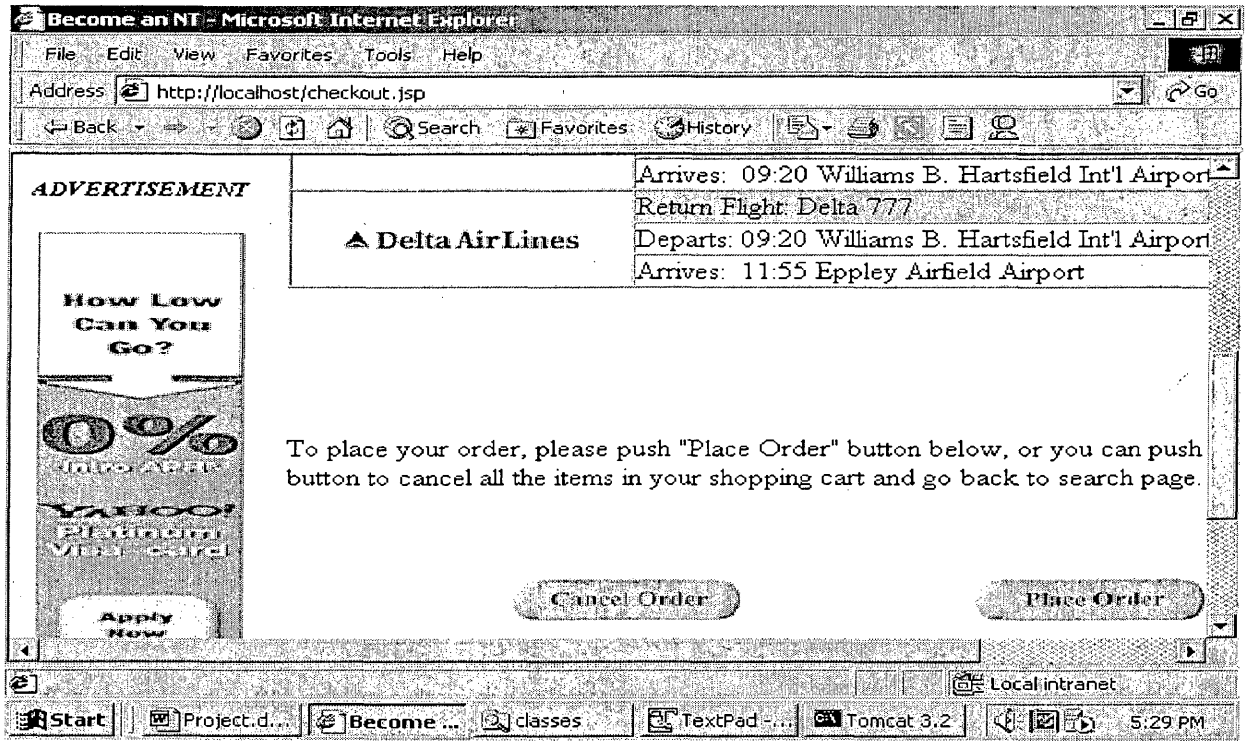
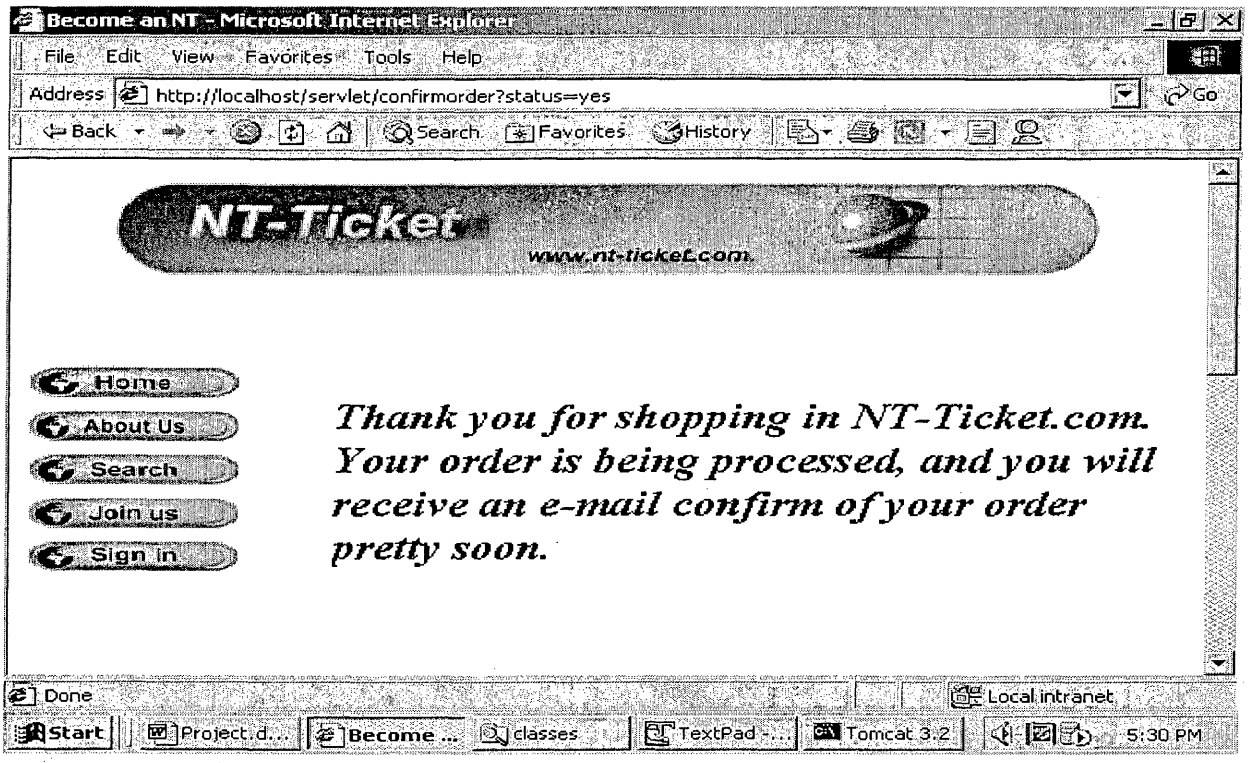


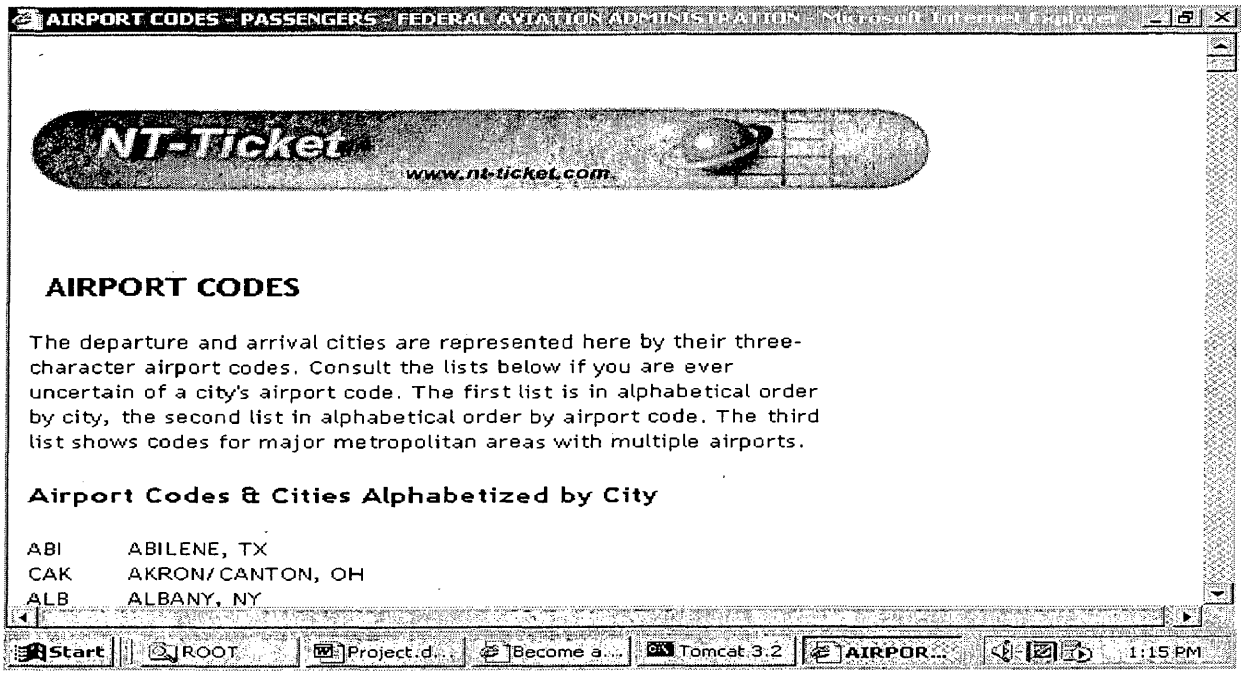
Figure 5.23



Page "airport_codes.htm"

The departure and arrival cities are represented in this project by their three-character airport codes. Page "airport_codes.htm" is a list that can be consulted by users to get their departure and arrival city's airport codes. Figure 5.23 shows the design of "airport_codes.htm".

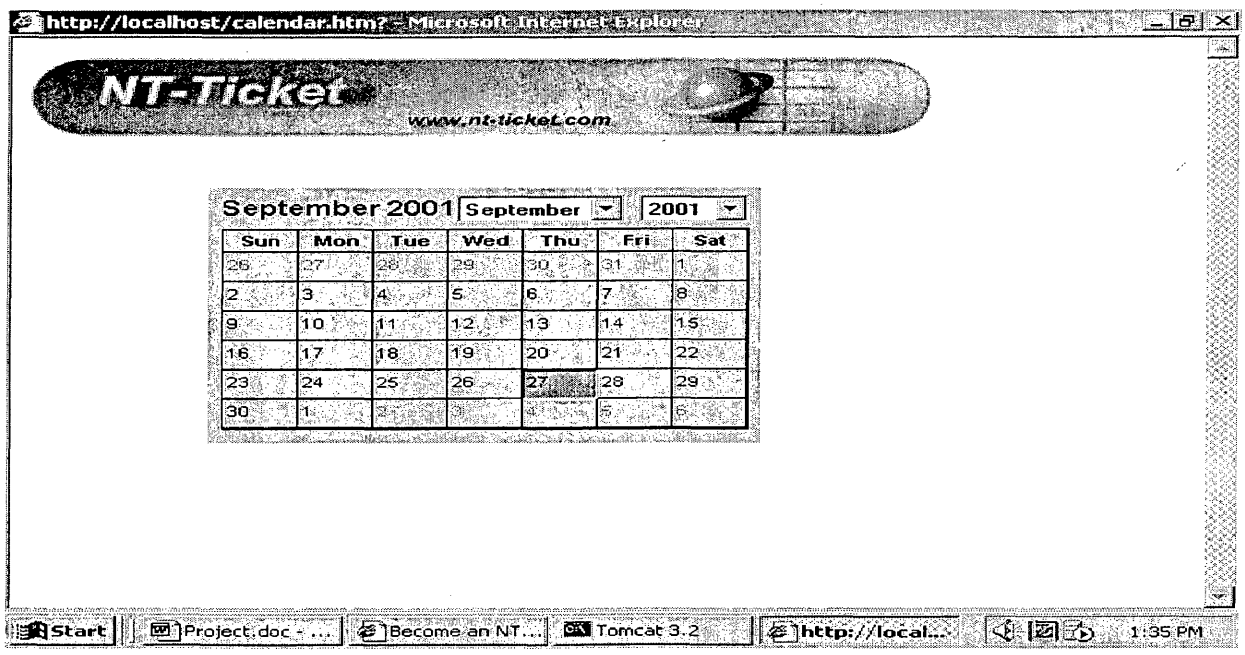
Figure 5.23



Page "calendar.htm"

This page contains a convenient calendar. User can use this calendar to check their itinerary if they want. Figure 5.24 shows the design of "calendar.htm" page.

Figure 5.24



Servlet "accountupdate" and accountpage.jsp

Users can check their account information from searchpage.jsp. By clicking "Your Account" icon, users are brought to accountpage.jsp. Figure 5.25, figure 5.26, and figure 5.27 show the design of accountpage.jsp.

Figure 5.25

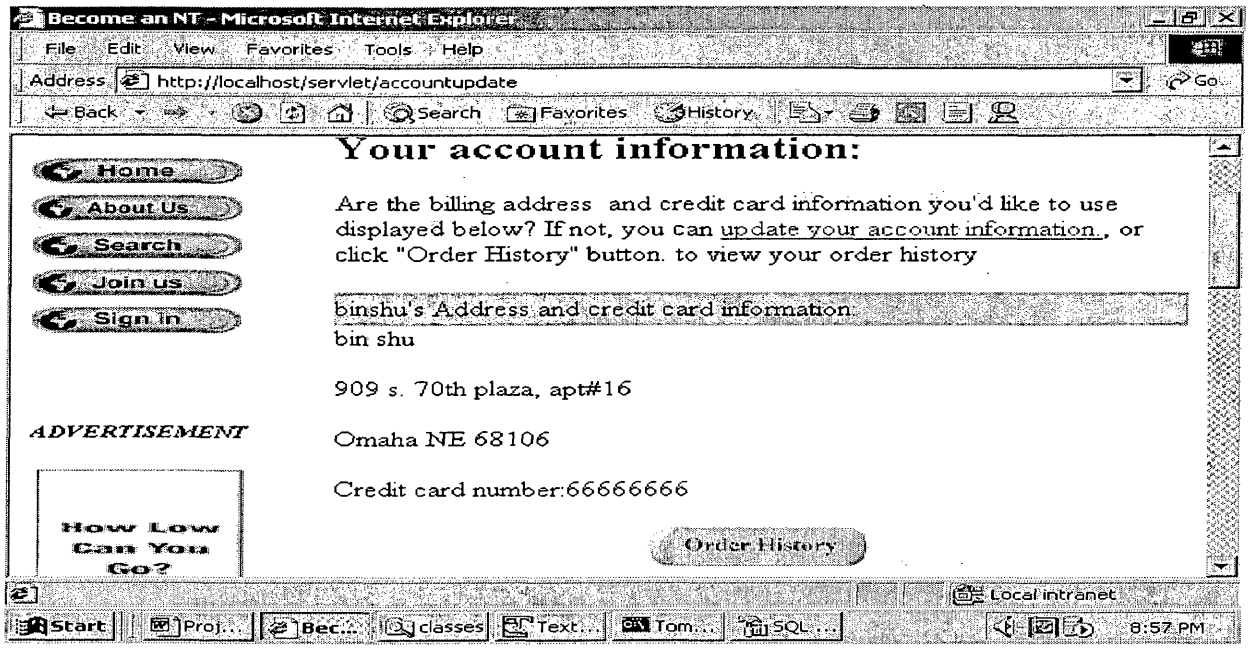


Figure 5.26

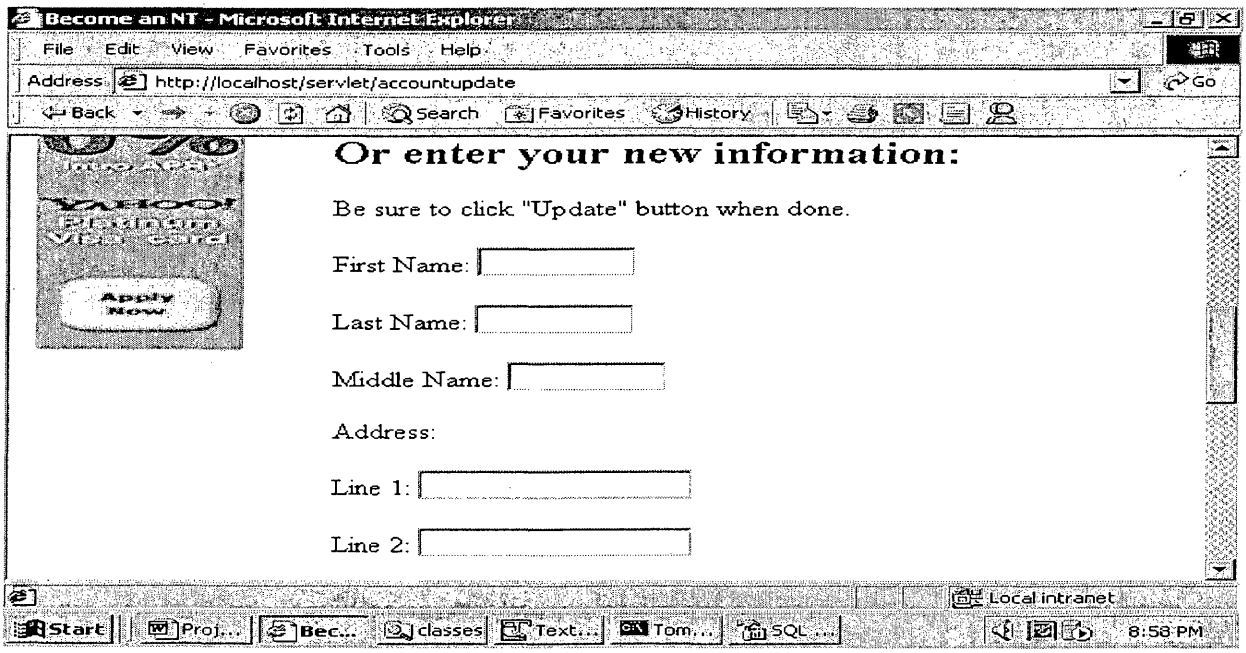
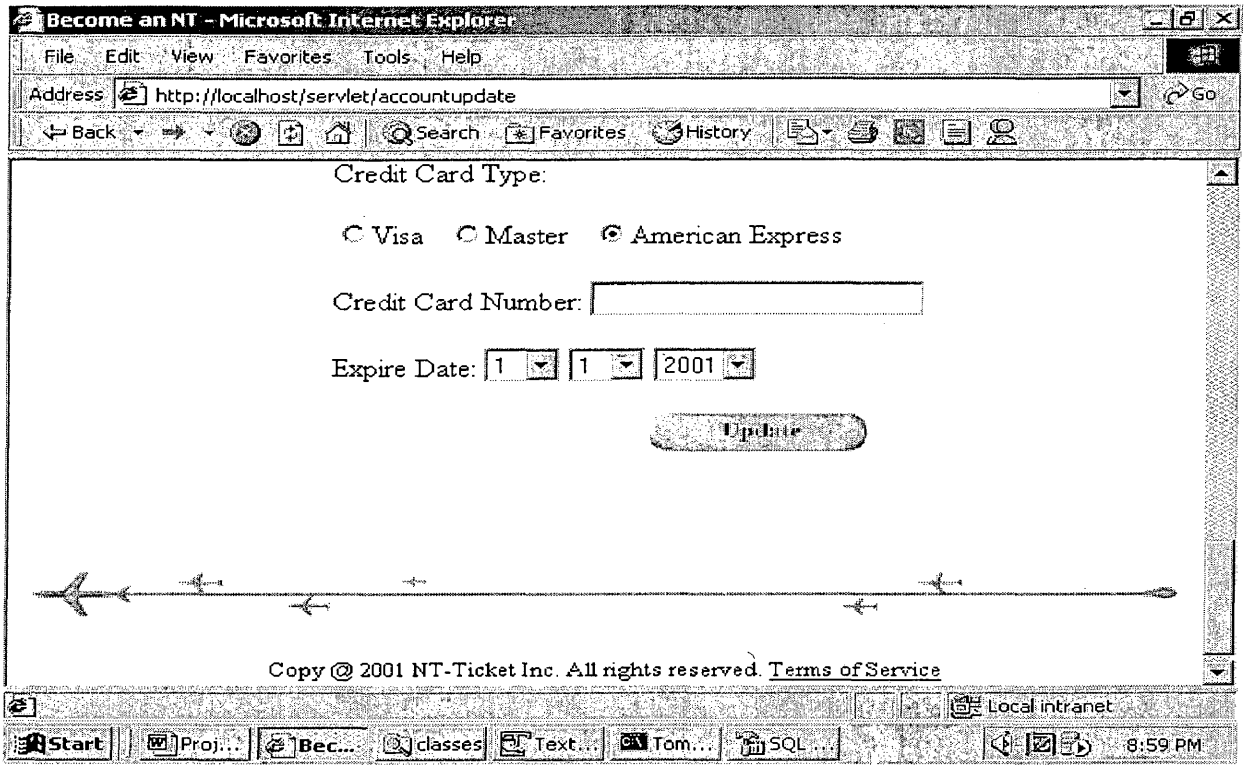


Figure 5.28



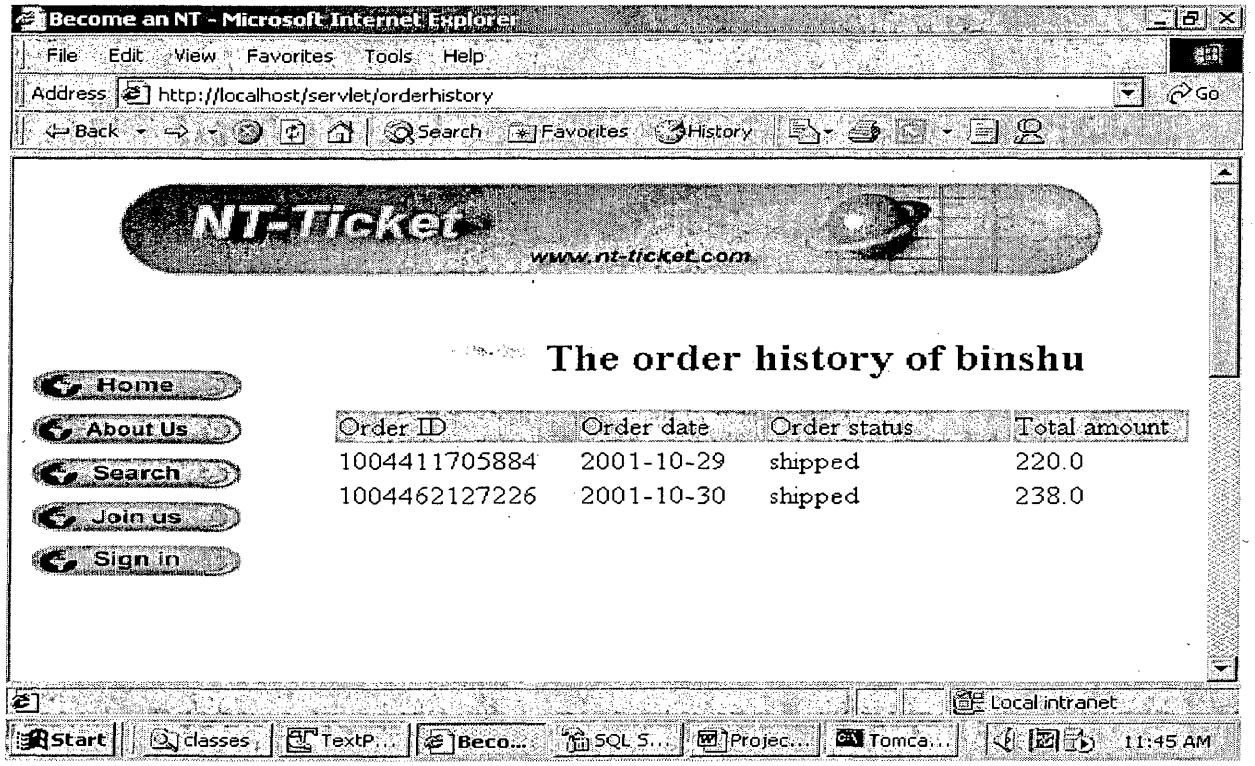
From this page, users can get their order history by clicking "Order History" button. Or if users feel that they need to update their billing address and credit card information, they can fill out the form contained in the page and click "Update" button. This will invoke the "accountupdate" servlet, which will then update the information in "customer" table accordingly. Servlet "accountupdate" can be found in attachment.

Servlet "orderhistory" and orderhistory.jsp

When users click "Order History" button in acountpage.jsp, servlet "orderhistory" will be invoked. This servlet will get information from table "orders", create some "order" objects if necessary ("order" object is a java object we create to encapsulate information such as order status, order date about an order, we'll discuss this object in next chapter),

put those object into session, and then show up orderhistory.jsp. Figure 5.28 shows the design of orderhistory.jsp. Source code of "orderhistory.jsp" can be found in attachment.

Figure 5.29



Chapter 6 Project discussion

In this chapter, we discuss two major issues we address in this project: object orient design and database connection pooling.

6.1 Object orient design of the project

The object-oriented philosophy has been increasingly adopted across all facets of software development. Java is an object-oriented language. We design four objects: order, ticket, ticketorder, and ShoppingCart throughout this object. Following paradigms illuminate the design of these four objects. Their code implementation can be found in attachment.

ShoppingCart
attribute: private Vector TicketsOrdered; private String owner;
methods: public ShoppingCart(String s) public String getOwner() public Vector getTicketsOrdered() public synchronized void addTicket(Ticket t) public synchronized void setNumOrdered(Ticket t, int numOrdered)

Order
attribute: private String order_id; private String order_date; private String status; private String total_amount;

methods:
public void setID(String s);
public String getID();
public void setDate(Timestamp t);
public String getDate();
public void setStatus(String s);
public String getStatus();
public void setTotal(float f);
public String getTotal();

Ticket
attribute: private String TicketID; private String Company; private String Departure; private String Arrival; private String flight1; private String flight2; private Timestamp Depart_time1; private Timestamp Arrival_time1; private Timestamp Depart_time2; private Timestamp Arrival_time2; private float cost;

methods:
public Ticket();
public String getTicketID();
public String getArrival();
public void setFlight1(String flight1);
public String getDeparture();
public String getFlight1();
public void setFlight2(String flight2);
public String getFlight2();
public Timestamp getDepartureTime1();

```
public Timestamp getArrivaltime1();

public Timestamp getDeparturetime2();

public Timestamp getArrivaltime2();

public String getCompany();

public float getCost();

protected void setTicketID(String ticketid);

protected void setDeparturetime1(Timestamp depart_time1);

protected void setCompany(String company );

protected void setDeparture(String departure);

protected void setArrival(String arrival);

protected void setArrivaltime1(Timestamp arrival_time1);

protected void setDeparturetime2(Timestamp depart_time2);

protected void setArrivaltime2(Timestamp arrival_time2);

protected void setCost(float cost);
```

TicketOrder
attribute: private Ticket ticket; private int number;

methods:
public TicketOrder(Ticket ticket,int number);
public Ticket getTicket();
public String getTicketID();
public String getCompany();
public String getFlight1();
public String getFlight2();
public String getDeparture();
public String getArrival();
public Timestamp getDepartureTime1();
public Timestamp getArrivaltime1();
public Timestamp getDepartureTime2();
public Timestamp getArrivaltime2();
public float getUnitCost();
public int getNumber();
public void cancelOrder();
public float getTotalCost();
protected void setNumber(int n);
protected void setTicket(Ticket t);

6.2 Database connection pooling

Throughout the whole project, our middle layer servlet and front-end jsp files us JDBC to make connection to back end database. JDBC provides a standard library for accessing relational database. Using the JDBC API, we can access a wide variety of different SQL database with exactly the same Java syntax.

After we have used JDBC for a while, it become obvious that opening a connection to a database is a time-consuming process. This is because that opening a connection might involve a series of low-level communication between network protocols and database connection protocols. For short queries, it takes much longer to open the connection than to perform the actual database retrieval.

To solve this problem, we can use technique called connection pooling. Essentially, a connection pool is an object holding connections to backend database and managing the number and status of connection. In our project, we use a class named "ConnectionPool" for connection pooling: pre-allocating database connections and recycling them as clients connect.

Our ConnectionPool class is able to perform the following tasks:

1. Pre-allocate the connection. This task is performed in the class constructor. Allocating more connection in advance speeds things up if there will be many concurrent requests later.
2. Manage available connections. If a connection is required and an idle connection is available, put it in the list of busy connections and then return it. The busy list

is used to check limits on the total number of connections as well as when the pool is instructed to explicitly close all connections.

3. Allocate new connection. If a connection is required, there is no idle connection available, and the connection limit has not been reached, then start a background thread to allocate a new connection. Then, wait for the first available connection, whether or not it is the newly allocated one.
4. Wait for a connection to become available. This situation occurs when there is no idle connection and we've reached the limit on the number of connections. This waiting should be accomplished without continual polling.
5. Close connections when required. We don't always have to close connections explicitly because connections are closed when they are garbage collected. But sometimes we want explicit control over the process.

Implementation and source code of ConnectionPool.java can be found in attachment.

Chapter 7 Conclusion

Over the last few years, the Internet has evolved from a communication medium, and subsequently a forum for advertising and online shopping, to a platform for complete business transactions between organizations. This evolution is irreversible. Only companies that take advantage of the Internet for faster business process and cost saving will be able to keep their customers.

In the meantime, the Internet is also changing the concept of programming applications. More and more new technologies are emerging rapidly every day. We are moving towards pervasive computing and towards electronic services.

During the next few years, there will be two electronic commerce roads-one for consumers and one for corporations. Consumers will be driven by the desire to obtain information, to communicate with others, and eventually, to buy and sell products or services. They will experiment with innovations as rapidly as the can be offered. On the other hand, corporations are driven by the need to become more efficient and "do more with less". There has been a growing emphasis on developing electronic applications that streamline current business practice, in addition to developing applications that are focused on new business-to-business or business-to-consumer applications.

It is conceivable that in the year 2010, many of us will look back at the last one or two decade as an era of profound change, perhaps even a historic turning point.

Bibliography

- Nabil R. Adam, Oktay Dogramaci, Aryya Gangopadhyay, Yelena Yesha, 1999, *Electronic Commerce*
- Paul Mayer, 2000, *The Business of E-commerce*
- Paul Timmers, 2000, *Electronic Commerce*
- Craig Fellenstein, Ron Wood, 1999, *Exploring E-Commerce*
- Ravi Kalakota, Marcia Robinson, *E-Business*
- Daniel Amor, 2000, *The E-business (R)EVOLUTION*
- Marty Hall, 2000, *Core Servlets and Java Server Pages*
- Wrox Press, 1999, *Professional Java Server Programming*
- Greg Holden, 1999, *Apache Server for Windows*
- *http://www.amazon.com*
- *http://www.priceline.com*
- *http://www.lowestfair.com*