1. Introduction

As the number of users on the World Wide Web increases everyday, its use in different areas is also growing. One of the most powerful aspects of the Web is that anybody who has Internet access can browse on the net. This enables sharing the information world wide.

One of the fast growing areas of the Web is distance education (or distance learning). The reason distance education on the Web is getting popular is because it has advantages over other types of distance education programs. It gives much more flexibility to the users. The users can take the courses they registered for at any computer connected to the Internet. They usually have a more flexible time frame to take their classes and their tests.

In terms of programming of these sites developers had until recently to use limited range of technology to choose from. These technologies usually involved Common Gateway Interface (CGI) programming, Javascript, and Microsoft's Active Server Pages. This paper demonstrates that Java servlets and JDBC can be used programming for these sites.

This paper discusses this new technology and explains how this technology can be used in programming in dynamic Web sites. Section 2 presents an introduction to Java language, Java servlets and JDBC. It also compares the present technology used in creating dynamic Web sites to the new Java servlet technology. Section 3 explains the software used in this study and their configuration. It also describes in detail how this project is done by explaining each servlet and its relation to the database class. Lastly, it describes the databse design. Section 4 gives the results and conclusion of the study, while Section 5 suggests the future work.

2. Background and Literature Review

This section begins by describing in Subsection 1, the alternatives that are currently available for programming interactive sites on the Web. In Subsection 2, we present Java servlets and how they can be integrated with JDBC. Finally Subsection 3 compares the present technologies with Java servlets.

2.1. Java Servlet Alternatives

Common Gateway Interface is one of the most commonly used server-side programming method to develop dynamic Web applications[11]. CGI is an application module that receives requests from a Web server. The application processes the data it receives and sends it back to the server. The server then sends the data to the client's browser. Server-side programming with CGI has some problems and limitations. CGI can be implemented in various ways, but typically it is implemented through the use of compiled and interpreted languages. C and C++ are the most commonly used compiled languages and Perl is the most commonly used interpreted language.

In terms of interpreted languages, Perl (Practical Extraction and Report language)[10] is the most popular one because of its superior string handling capabilities and extended fuctionality[9]. Although Perl is a very powerful language at processing text, it has a downside because it requires the server to start a new interpreter for each request. This takes quite a bit of time and resources.

One other thing to consider is that CGI runs as a process completely separate from the Web server. This requires additional coding for session tracking when session tracking is necessary on that site.

Server-side JavaScript is another solution for generating dynamic Web sites. This is achieved by embedding JavaScript into the HTML pages. However, server-side JavaScript is supported only on Netscape’s Enterprise and FastTrack Web servers.

Microsoft’s Active Server Pages (ASP) is another method of programming dynamic sites. Similar to server-side JavaScript, ASP is also embedded in HTML pages. ASP is supported only by Microsoft’s Web servers (Internet Information Server and Personal Web Server).

2.1.1. Common Gateway Interface

CGI is just a protocol, a formal process between a Web server and a separate server-side program[1]. The server encodes the client's form input data, and the CGI program decodes the form and generates output. The protocol is independent from the programming language used to program CGI.

CGI scripts called in two main ways (methods). The HTTP GET method is used in document retrievals where an identical request will produce an identical result, such as

a dictionary lookup. The HTTP POST method sends form data separate from the request. The GET method is only safe for short read-only requests, whereas POST is safe for forms of any size, as well as for updates and feedback responses. Therefore, by default, the CGI module uses POST for all forms it generates.

2.2. Java Servlets

2.2.1. Introduction

Java[7] is a relatively new and powerful programming language that provides many useful features to software developers. It inherited its strengths from C++ while eliminating most of the problems of C++. The syntax and programming logic of Java are very similar to that of C++. Java is simple, object oriented, portable, robust, and multithreaded. The syntax for Java is simple and a cleaned-up version of the version of C++. There is no need for header files, pointer arithmetic (or even a pointer syntax), structures, unions, operator overloading, and virtual base classes. Java is an object oriented language, like C++. The main difference between the two is in the area of inheritance. C++ supports multiple inheritance while Java supports only single inheritance.

Java is platform independent (portable). Once the Java code is compiled into byte code, it runs on every machine which has Java Virtual Machine (JVM) on it. This is one of the biggest advantages of Java over C++. The Java compiler detects many problems during compilation. Some of these problems are detected during run time with other language compilers. This makes Java a robust language. Java is also multithreaded. This feature gives Java a better real-time behavior[6].

The Java language was originally intended for use in small, embedded devices[2]. Its first use on the Web came in the form of applets. Until recently, Java has not been used in serious server-side Web development. Now, with the improvements in Java application programming interfaces (API's), especially the Servlet API, Java is becoming an important tool for server-side Web programming.

2.2.2. Java Servlets and Their Applications

A servlet is a dynamically loaded module that services requests from a Web server and is a generic extension to Java-enabled servers. The most common use of servlets is to extend Web servers providing secure, portable, and an easy-to-use replacement for other

server-side scripting methods, such as CGI. Servlets run entirely in the JVM and on the server-side; therefore they don't depend on browser compatibility (unlike Java applets).

Servlets can be used for any number of Web-related applications where dynamic Web pages are needed. One of the biggest applications for servlets is developing E-commerce sites since they are one of biggest trend in Web development. Servlets can be connected to a database, either on the same server or on a different machine, and make accessible the contents of that database on the Web through JDBC which will be described later. Since the servlet API includes classes and interfaces for session tracking, servlets can be easily used on sites where session tracking is needed.

2.2.3. Servlet Advantages

Today, servlets are one of the most exciting and new technologies in server-side Web development. Servlets are efficient, persistent, portable, robust, extensible and secure[3].

Servlets are efficient, because a servlet’s initialization code is executed only once when the Web server loads it for the first time. Once the servlet is loaded, handling new requests is only a matter of calling a service method. Since session tracking is built-in to the servlet API, servlets can maintain states between requests, which makes them persistent. Because servlets are written in Java, they are platform independent and portable. This feature enables servlets to be moved to a new operating system without changing the source code. Also because servlets are written in Java, they are developed with the access to the entire Java Development Kit (JDK) API. Therefore powerful Java packages, such as JDBC, Networking, Remote Method Invocation, etc., can be used in servlets. Servlets are secure, because they run on the server side, inheriting the security provided by the Web server. Servlets can also take advantage of the Java Security Manager API.

2.2.4. Architecture of Servlets

There are two packages that constitute the Servlet API; these are `javax.servlet` and `javax.servlet.http[6]`. The `javax.servlet` package contains classes to support generic, protocol-independent servlets. These classes are extended by the classes in the `javax.servlet.http` package to add HTTP-specific functionality. The top-level package name is `javax` instead of the familiar `java`, to indicate that the Servlet API is a

standard extension.

Every servlet must implement the `javax.servlet.Servlet` interface. Most servlets implement it by extending one of two special classes: `javax.servlet.GenericServlet` or `javax.servlet.http.HttpServlet`. A protocol-independent servlet should subclass `GenericServlet`, while an HTTP servlet should subclass `HttpServlet`, which is itself a subclass of `GenericServlet` with added HTTP-specific functionality.

Unlike a regular Java program, and just like an applet, a servlet does not contain a `main()` method. Instead, certain methods of a servlet are invoked by the server in the process of handling requests. Each time the server dispatches a request to a servlet, it invokes the servlet's `service()` method.

A generic servlet should override its `service()` method to handle requests as appropriate for the servlet. The `service()` methods accepts two parameters: a request object and a response object. The request object tells the servlet about the requests made by the clients (web browsers), while the response object is used to return a response. Figure 1[6] shows how a generic servlet handles requests.

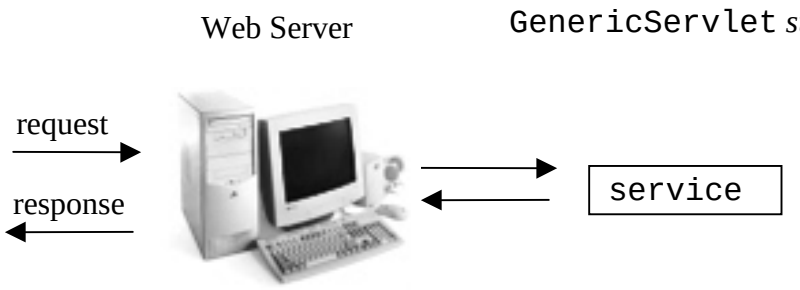

Figure 1. A generic servlet handling a request.

In contrast, an HTTP servlet usually does not override the `service()` method. Instead, it overrides `doGet()` method to handle GET requests and `doPost()` method to handle POST requests. An HTTP servlet can override either or both of these methods. The `service()` method of `HttpServlet` handles the setup and dispatching to all the `doXXX()` methods. Figure 2[6] shows how an HTTP servlet handles GET and POST requests.

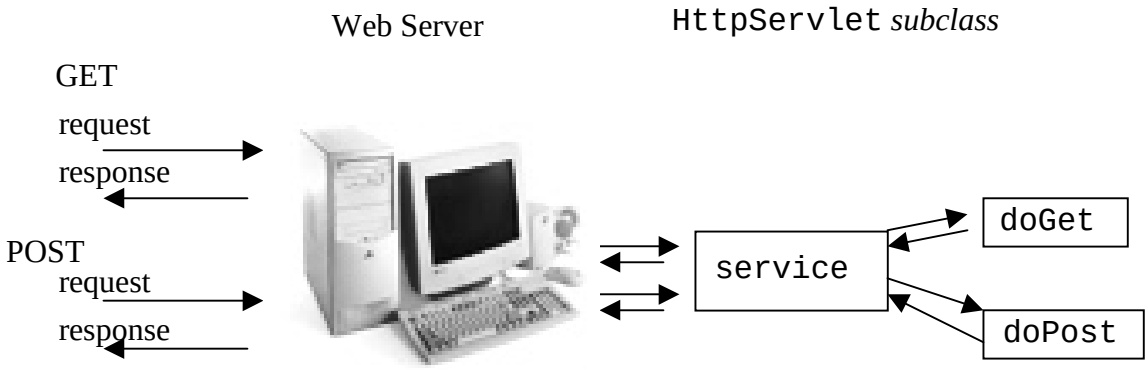

Figure 2. An HTTP servlet handling GET and POST requests.

The remainder in the `javax.servlet` and `javax.servlet.http` packages are largely support classes. For example, the `ServletRequest` and `ServletResponse` classes in `javax.servlet` provide access to generic server requests and responses, while `HttpServletRequest` and `HttpServletResponse` in `javax.servlet.http` provide access to HTTP requests and responses. The `javax.servlet.http` package also contains an `HttpSession` class that provides built-in session tracking functionality and a Cookie class that allows you to set up and process HTTP cookies.

2.2.5. Java Servlet Engines

As mentioned above, servlets are standard extensions to Java. However, the servlet API is not part of the core Java API. Therefore, the servlet API must be present on the server-side, in addition to JVM, in order for the servlets to work. The first servlet API was made available by the Java software division of Sun Microsystems. This product is called as Java Servlet Development Kit (JSDK)[6]. In addition to JSDK, there are many third party servlet engines (standalone and add-on) available from different manufacturers.

A standalone engine is a Web server that includes built-in support for servlets. Some of the standalone servlet engines include Sun's Java Web Server[6], the World Wide Web Consortium's Jigsaw Server[6], and Netscape's Enterprise Server (version 3.51 and later)[6]. One advantage of these servers is that once the Web server is installed and configured on the server-side, the servlet engine is also installed. Therefore, an add-on servlet engine does not need to be separately installed and configured.

An add-on servlet engine functions as a plug-in to an existing Web server. It adds servlet support to a server that was not originally designed with servlets in mind. Some of the add-on servlet engines include Java-Apache project's JServ module[6], Allaire Software's Jrun[6], and IBM's WebSphere Application Server[6].

2.3. JDBC Data Access

JDBC is a Java API for executing Structured Query Language (SQL) statements. Sun Microsystems says that JDBC is a trademark and is not an acronym for Java Database Connectivity, however, it is offen associated with Java Database Connectivity. It consists a set of classes and interfaces. JDBC makes it possible to write database applications using a pure Java API [8].

In its simplest form, JDBC makes it possible to do three things: a) establish a connection with a database, b) send SQL statements, and c) process the results. JDBC cannot be used to create databases. Therefore, in order to access a database using JDBC, a database has to be created with a Relational Database Management System (RDBMS).

Open Database Connectivity (ODBC) is probably the most widely used programming interface for accessing the relational databases. ODBC, developed by Microsoft, was the first standard database driver. ODBC drivers provide a common API to database clients. However, using ODBC drivers in Java has its own drawbacks. It is written in C language, which is not an object-oriented language. It uses pointers and other programming structures that Java does not support. Also, ODBC drivers must be installed on the client side. This requires that an ODBC driver has to be present on the client side in order for Java applets to be run on the client's Web browser. As mentioned above, JDBC provides a common database programming API for Java programs.

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. Figure 3[4] shows a simplified version of a two-tier architecture.

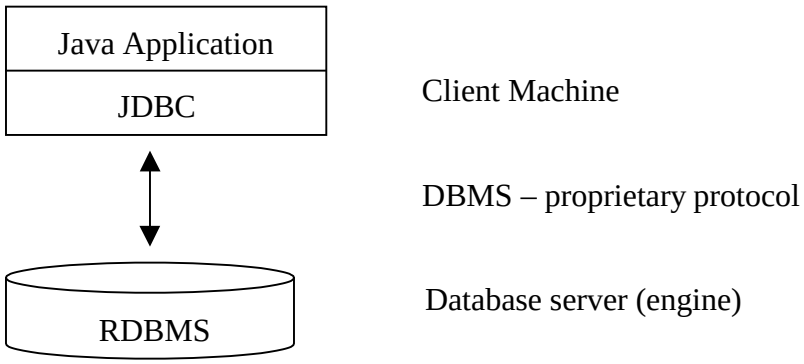

Figure 3. JDBC two-tier model.

In the three-tier model, commands are sent to a “middle tier” of services, which then sends SQL statements to the database. The database processes the SQL statements and sends the results back to the middle tier, which sends them to the user. In many cases the three-tier architecture can provide performance advantages. Figure 4[4] show the three-tier model.

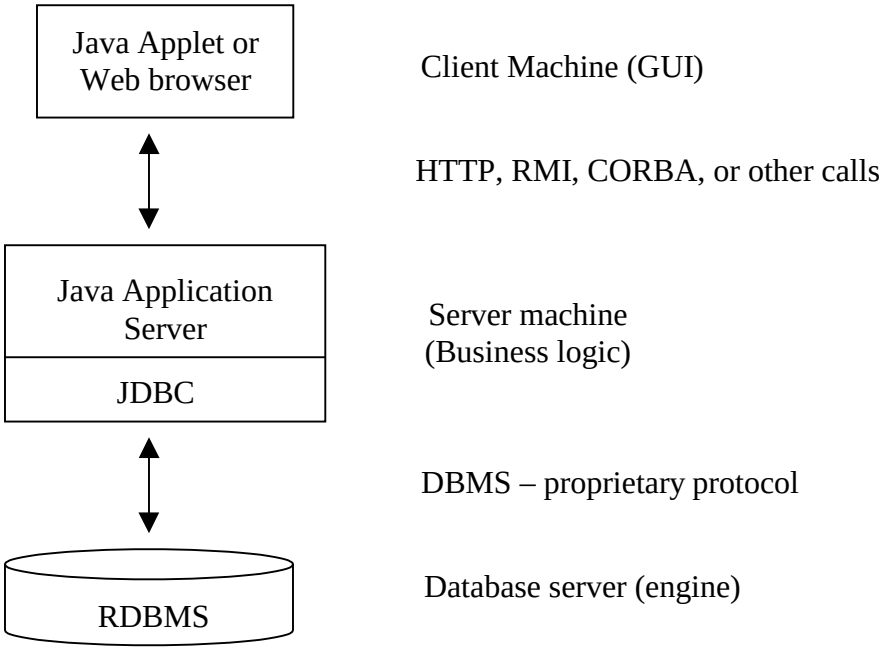

Figure 4. JDBC three-tier model.

2.4. Present Technology vs Java Servlets

As the World Wide Web continues to grow, its use for different purposes are also increasing. One of the rapidly growing uses of the Web is on-line education and testing. This can be considered as a form of distance learning. The advantages of on-line education and testing are many. Since a site on the Web is accessible from all over the world, everybody who has authentication to use the site can benefit from that site. The limitations of time and location are eliminated in on-line education. On-line testing provides instant results (scores/grades) to those who take it, while minimizing grading and correcting by hand.

The examples of the sites that provide Web based learning and testing include Web Course Tools (WebCT)[12], Blackboard.com[13], Learn University.com[14], University of Phoenix (Distance Learning section)[15], etc. Programming of these kind of sites typically involves a server-side scripting language and a RDBMS. As discussed in earlier sections, CGI programming dominates server-side dynamic Web programming. CGI technology is old and has its downsides. A small portion of this type of Web programming involves JavaScript, however JavaScript cannot be used for creating secure sites.

Java servlets have a number of advantages (as discussed in the previous sections) over the existing technologies. Some of the advantages are security, platform independent and full access to Java API. Session tracking and interaction with a RDBMS are necessary features that these kind of sites should posses. Since session tracking is a built-in feature of servlets and database access from servlets is very easy through JDBC, developing dynamic Web sites and Web database applications with servlets are simple and efficient.

3. Methodology

This section describes the different pieces of software used in this study and their configuration. It also describes the project by explaining the components of it, such as the servlets, JDBC class and the database design.

3.1. Software Used in This Study

All the pieces of software used in this study are compatable with and installed on a Red Hat Linux 6.0, kernel 2.2.5-12, operating system. We used the Apache Web Server that comes with the above Red Hat distribution as the Web server. Although, Linux comes with a third party Java compiler, Kaffe Virtual Machine, I downloaded and installed the JDK for Linux provided by Blackdown.org which follows the Sun Microsystems specifications for JDK. The version of JDK, I used, was 1.1.7v3.

As a servlet engine, I downloaded and installed the Java Servlet Development Kit 2.1 available from Sun Microsystems. Currently, this is the latest servlet engine available. In order for servlets to function correctly, some features of this engine had to be configured before its use. Some of these features are the port number that the servlets run, servlet directory, and servlet properties file which holds the names of the compiled Java servlets.

I used Mini SQL 2.0.11 as my database engine. This is a freely available RDBMS (for non-profit organizations) from Hughes Technologies, Australia (www.Hughes.com.au). This database server is very small and compared to its size very powerful. Mini SQL is designed and developed to run on UNIX flavor operating systems. Its installed version takes less than 3 megabytes space on hard disk.. This RDBMS provides a SQL monitor and supports standard SQL commands. It posses the properties of many large RDBMS's without the overhead of those. In order to connect my Java code to this database server, I used mSQL-JDBC 1.0 driver. This driver is also freely available from the Center for Imaginary Environments (www.imaginary.com/Java) to whoever wants to use it.

3.2. The Project

This study involved, in terms of Java servlets and JDBC, sixteen servlet programs and one database program. While servlets generated the Web pages, the database program functioned as a bridge between the servlets and the database server. Most of the servlets created objects of the database class and accessed the functions of the database class, which will be explained in detail later. In addition to Java servlets and JDBC, this project involved some database design and programming. The database design consisted of five tables. The database design will be explained later.

Session tracking was an essential part of this project. It was accomplished by passing the user name from one servlet to another as a hidden field (since user name was the unique identifier).

Table 1 gives servlet names and the file name which generates that servlet. A general flow diagram for the servlets is given in Figure 5.

Table 1. Servlets and their brief descriptions.

Servlet Name	File Name
Welcome	welcome.java
Welcome2	welcome2.java
QABoard	qaboard.java
List	list.java
Message	message.java
Reply	reply.java
EntryRep	enrtyrep.java
Post	post.java
Entry	entry.java
Sample	sample.java
SamAns	samans.java
Quiz	quiz.java
Answer	answer.java
Grades	grades.java
Change	change.java
Passwd	passwd.java

The first function of the database class (quizdb.class) is `connect()` function, which makes a connection to the database server on the local host. If the connection is established and the database (quiz1) is opened successfuly, this function returns true, otherwise false. The next function is the `loginCheck()` function (called from the `Welcome` servlet), which takes the user's name and password from the `Welcome` servlet and searches the relevant database table (passwd) for these entries. If the user name and password are found in the database and matched, this function returns true, otherwise false. Explanations of the tables will be described later. The `stop()` function calculates the duration of quiz, and inserts the duration into the taken table. This function also marks the database that student has taken the quiz.

The `insertGrade()` and `getGrades()` functions of the database class inserts a student's quiz grade into the grade table and retrieves and returns a student's grades from

the grade table, respectively. The `insertGrade()` function is called from the `Answer` servlet and the `getGrades()` function is called from the `Grades` servlet. There are some functions which involve with the security when a student attempts to take a quiz called from the `Quiz` servlet. The `taken()` function searches the database to see if the student has already taken that particular quiz. The `ipCheck()` function checks the IP address of the remote computer that a student is trying to view the quiz page. Because students are restricted to take quiz only from a set of computers, access from other computers will be denied. The `timeCheck()` function checks the time of the quiz time and student's connection time. If the student is trying to access the quiz page other than the quiz time, the student's access to this page will be denied.

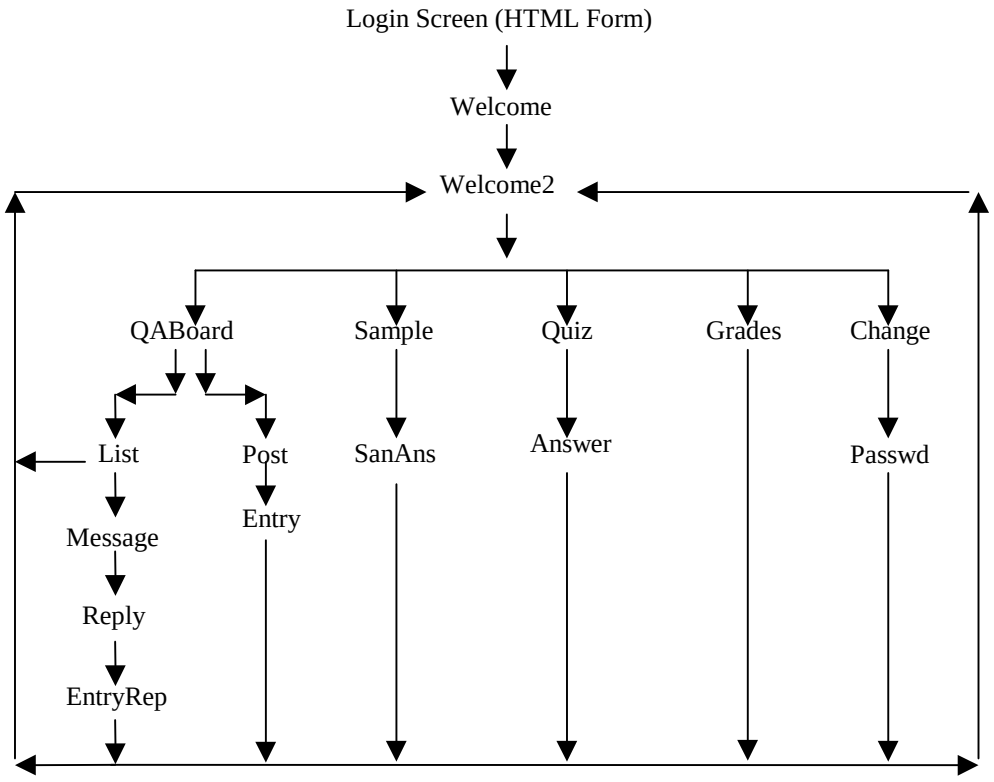

Figure 5. Flow chart for the servlets.

There are also some database related functions involving the Question/Answer (Q/A) posting board. These functions are `getSubjects()`, `getOneSubject()`,

`getMessage()`, `insMessage()` and `insReply()`. The `getSubjects()` function gets the message number, the user name who posted that message, posting date and the subject of that message from the subjects table. This function gets all the entries from the subject table. The `getOneSubject()` function does the same thing with `getSubjects()` function, except this function gets only one message from the database. The `getMessage()` function gets the message body of a particular message from the message table. The `insMessage()` function inserts the message number, user name, posting date and time, subject of the message and the message body into subject and message tables. The `insRepy()` function inserts a reply to a previously posted message in to the database.

The first screen that comes on when the program is started is the login screen (Figure 6). This is the only page that is generated by HTML. The page after the login screen is the welcome screen. This screen is generated by `Welcome` servlet. This servlet takes the user name and password from the previous login page and checks if they exist in the database using the `loginCheck()` function of the database class. If the user didn't enter user name or password, or if user name/password couldn't be found in the database, this servlet generates an error page displaying the appropriate error message. If the user name and password are found in the database, this servlet calls the next servlet.

The next page, generated by Welcome2, is the main screen (Figure 7) for this project. This page has links to other pages, such as Q/A posting board, sample and real online quiz pages, etc.

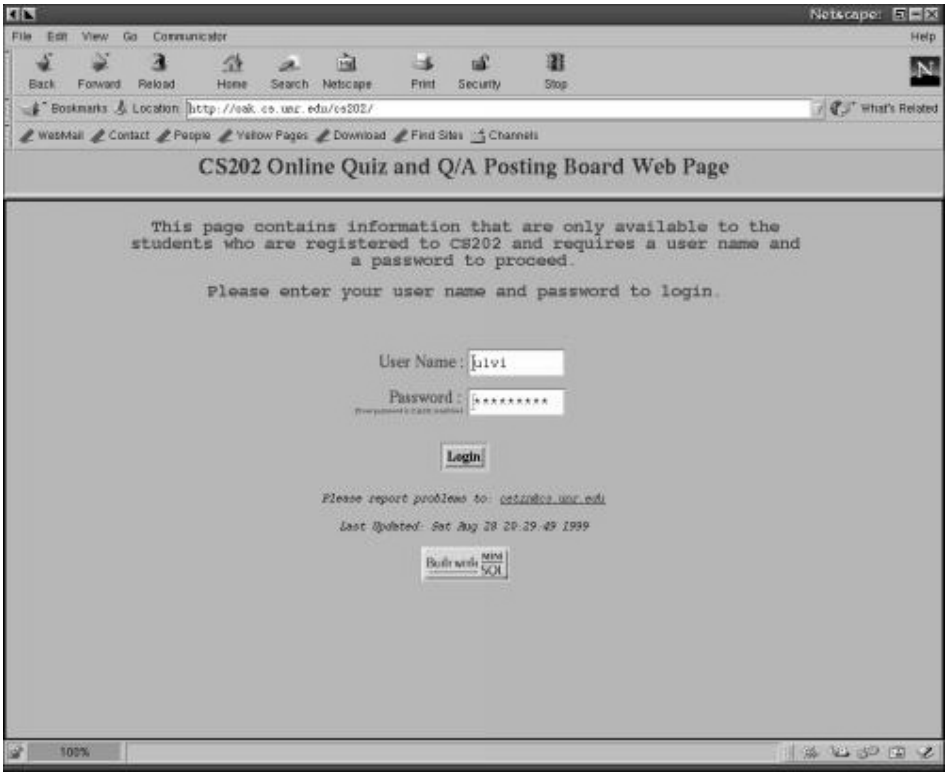

Figure 6. Login screen.

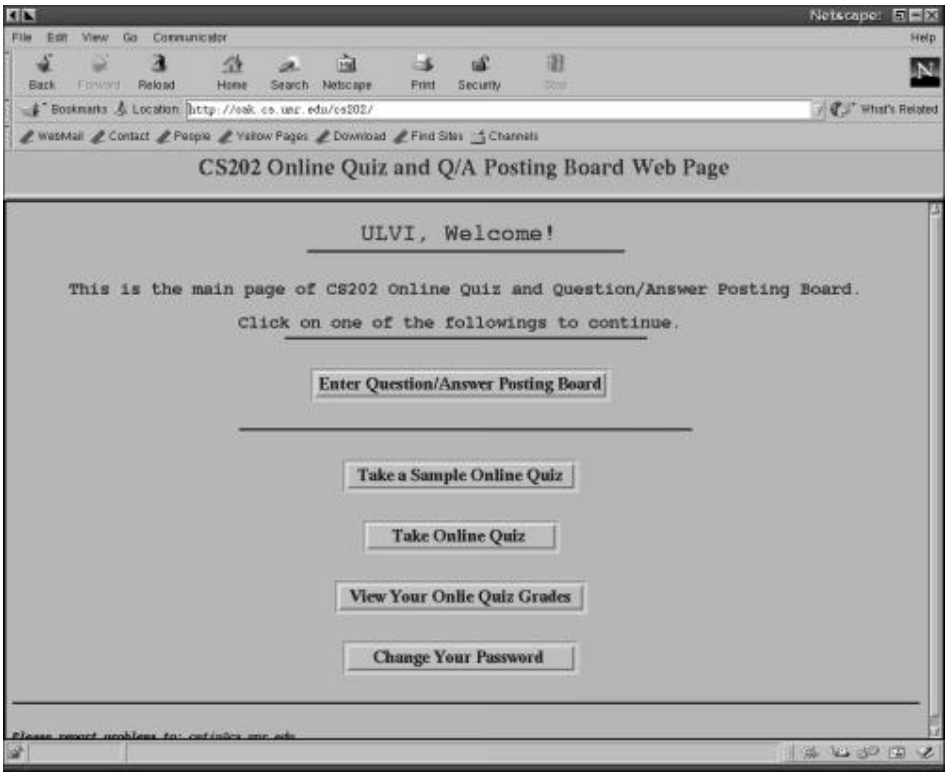

Figure 7. Main screen.

The Q/A posting board page (Figure 8), generated by QABoard, has two links, one is to view the previously posted messages and reply to a particular message, and the other is to post a new message. The page that shows the previously entered messages, generated by List, gets the entries from the database and displays them as links to the messages using the `getSubjects()` function of the database class. If the user clicks on one of the links, the message body is displayed on the next page and an option to reply to this message. The user can also choose to post a new message without replying to a message.

The main page includes two links to the quiz pages. One of the links is to the sample quiz page (Figure 9). This page, generated by Sample servlet, displays how the real quiz will look like and explains how it will work and familiarize the students with the real quiz. There is no time limit on this page. After the student submits the answers, the next page, generated by SamAns servlet, displays the correct answer for each question and the score of that user. The real quiz page is the page where students take the online quizzes (Figure 10). This page is generated by Quiz servlet. When this page is loaded, it first checks if the student has taken the quiz before by calling `taken()` function of the database class, then checks the IP address of the remote computer by calling `ipCheck()` and the time by `timeCheck()` functions. This servlet also starts a Java timer to measure the duration of the quiz. A JavaScript timer shows the quiz time starting from the loading of the quiz page. The purpose of this JavaScript timer is to show the elapsed time to the student. The timer is stopped when the student presses the submit button when he/she finishes taking the quiz. This servlet passes the duration measured by the Java timer to the next servlet. The next page, generated by Answer servlet, calculates the duration of the quiz and updates the taken table in the database, inserting the duration of the quiz and marking the taken field as true. This servlet also displays the correct answers of the questions and the student's grade. It inserts the student’s grade for that quiz into the grades table by calling `insertGrade()` function.

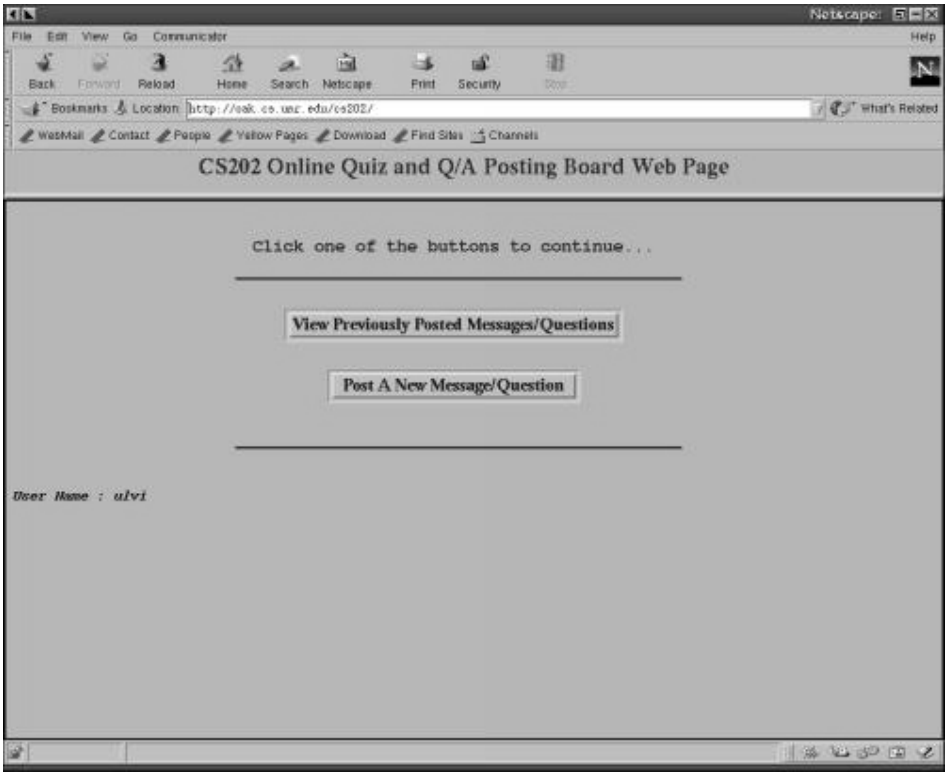

Figure 8. Question/answer posting board main page.

Finally, the main page has two links to two other features. By following the links, the students can view their previously taken online quiz grades (Figure 11). This page, generated by `Grades` servlet, uses the `getGrade()` function of the database class to retrieve the grades of the student. The students can also change their existing passwords (Figure 12). The page, generated by `Change` servlet, takes the existing and new

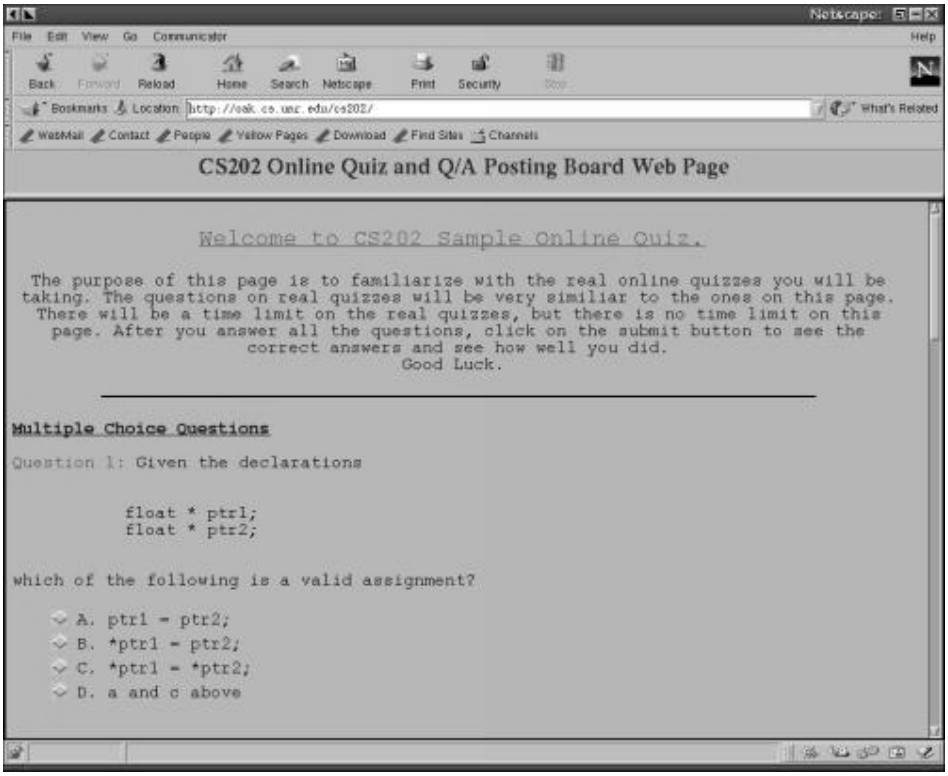

Figure 9. Sample quiz page.

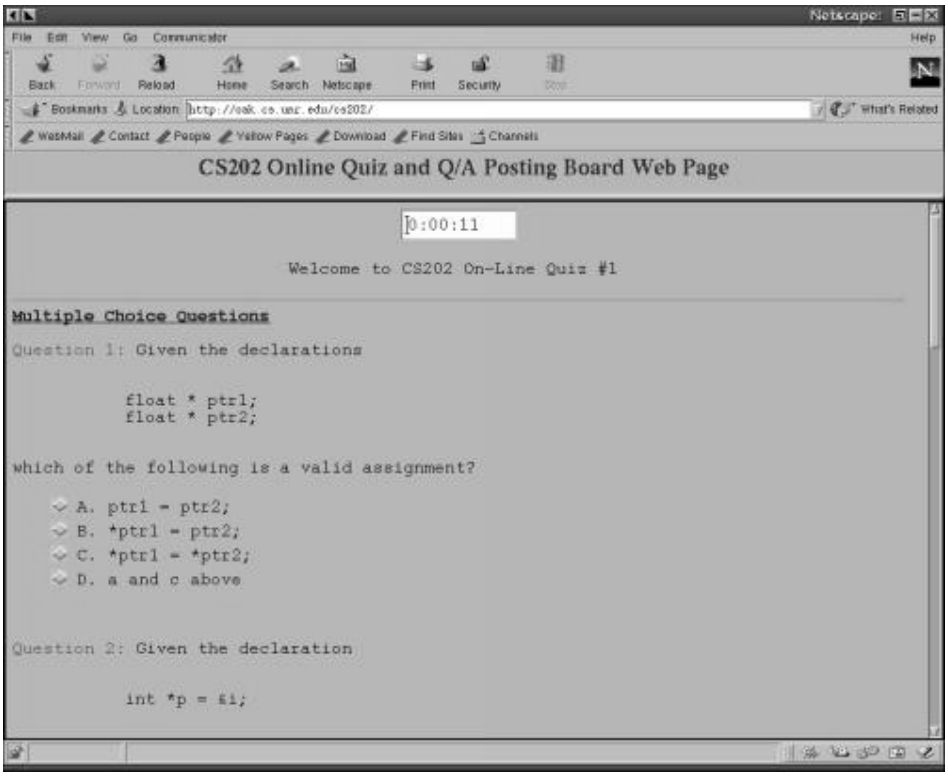

Figure 10. Online quiz page.

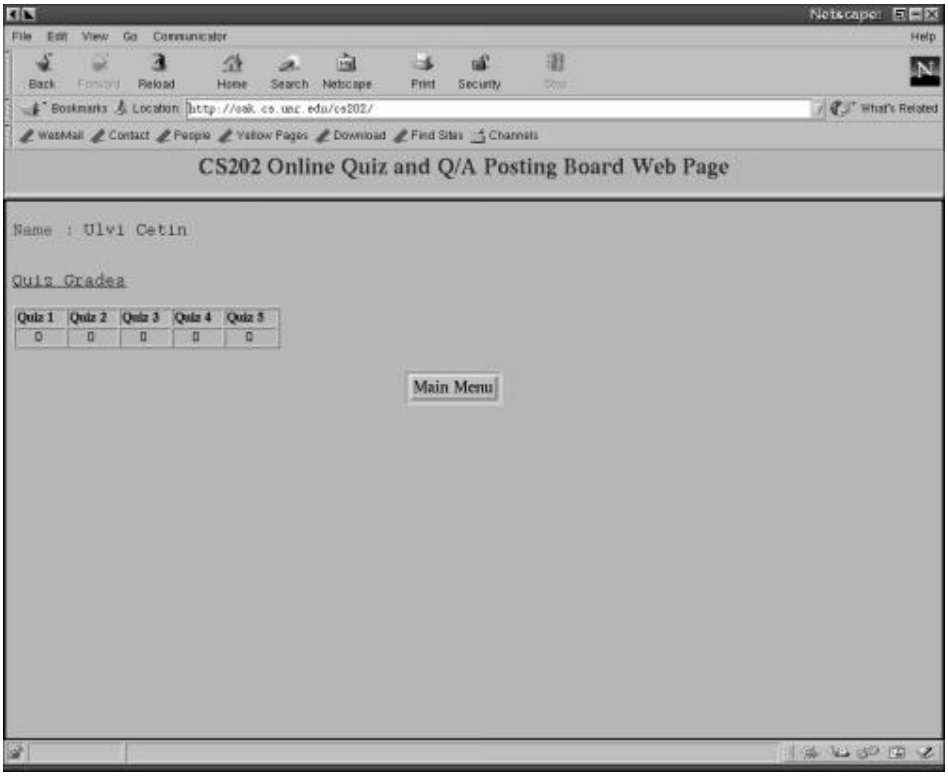

Figure 11. Grade viewing page.

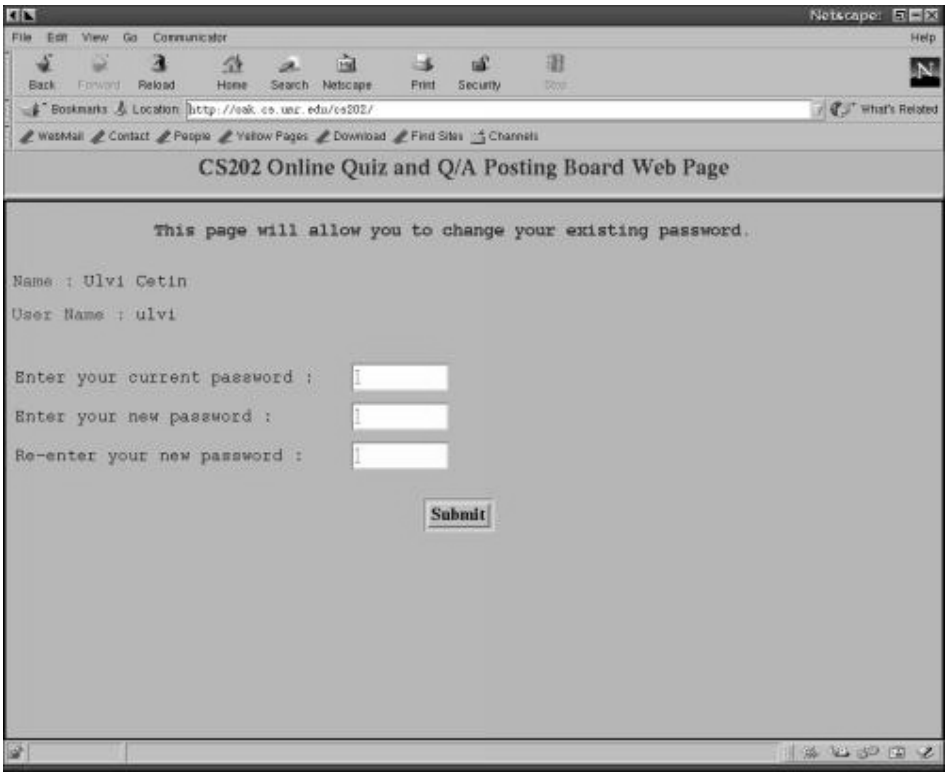

Figure 12. Password changing page.

passwords of the student and calls the next servelt (`Passwd`). `Passwd` servlet updates the student's password (by `changePasswd()` function) after checking its validy.

The database design, as mentioned earlier, involved 5 tables. These tables are password, taken, grade, subject and message. The primary key was the user name for tables password, taken, grade and subject, and the primary key was message number for table message. The password table contains general information about the users. This table stores the first and last names of the users, user names, passwords, e-mail addresses and a field that specifies whether the user has super user privileges. The taken table includes a field for the user name, a filed for the time for each student to finish that particular quiz (duration), and a field for if the student has taken that quiz previously. The grade table stores the user names and the grades of the students for each quiz. The subject table contains user fields for name, the number of the message, message insertion date and time and the subject of the message. The message table stores the number of the messages and the message body.

4. Results and Conclusions

This study involved server-side programming with Java servlets to produce dynamic Web pages and Web database applications. As mentioned before, currently, CGI is the most commonly used medhod in this field. Java servlets are new, exciting fast growing technology in web programming.

This study demonstrated that Java servlets can be used on-line testing/courseware programming. It utilized instant grading for student's quizzes, eliminating human grading/corrections. The grades were stored in the database. This also simplifies the calculation of student's average final course grades.

The Q/A posting board provides a useful tool for message/question posting and a discussion board. Students can benefit from it by posting new questions and even by answering some of the questions.

The database design used is a three-tier model. The database class (`quizdb`) acted as the middle application server between the Java servlets and the database engine. Although speed of database queries wasn't much of an issue in this project, the three-tier model is used here because it usually provides better performance.

5. Future Work

In the future, more functionality may be added to this project. Some of these functionalities may include, automatic assignment submission and grading, chat room(s), posted classnotes, etc.

Although the quizzes are timed, there is no automatic submission feature on this page when the quiz ends. This feature could be added on the quiz page. So that, at the end of the quiz, the next page (answer page) loads automatically before student submiting the quiz page.

JavaServer Pages (JSP) is another new and exiting technique in dynamic web site programming. JaveServer Pages are standard extension of the Java Servlet API. Possible future studies may involve compining servlets and JSP to generate better results.

References

[1] Tom Christiansen and Nathan Torkington. *Perl Cookbook*. O'Reilly and Associates, Inc., Sebastopol, CA, 1996.

[2] Bruce Eckel. *Thinking in Java: A Definitive Introduction to Object-Oriented Programming in the Language of the World Wide Web*. Prentice Hall, Upper Saddle River, NJ, 1998.

[3] James Goodwill. *Developing Java Servlets*. Sams Publishing, Indianapolis, IN, 1999.

[4] Graham Hamilton, Maydene Fisher and Rick Cattell. *JDBC Database Access with Java: A Tutorial and Annotated Reference*. Addison Wesley Longman Inc., Reading, MA, 1997

[5] Cay S. Horstmann and Cary Cornell. *Core Java 2, Volume I: Fundamentals*. Prentice Hall, Upper Saddle River, NJ, 1998.

[6] Jason Hunter and William Crawford. *Java Servlet Programming*. O'Reilly and Associates, Inc., Sebastopol, CA, 1998.

[7] Jamie Jaworski. *Java 1.2: Unleashed*. 4th Edition. Macmillan Computer Publishing, Indianapolis, IN, 1998.

[8] George Reese. *Database Programming with JDBC and Java*. O'Reilly and Associates, Inc., Sebastopol, CA, 1997.

[9] Mark Swank and Drew Kittel. *World Wide Web Database Developer's Guide*. Sams Publishing, Indianapolis, IN, 1996.

[10] Larry Wall, Tom Christiansen and Randal L. Schwartz. *Programming Perl*. 2nd Edition. O'Reilly and Associates, Inc., Sebastopol, CA, 1996.

[11] William E. Weinman. *The CGI Book*. New Riders Publication, Indianapolis, IN, 1996.

[12] URL: www.webct.com

[13] URL: www.blackboard.com

[14] URL: www.learnu.com

[15] URL: www.uophx.edu