**Online Testing System**

**ABSTRACT**

Now a days it is essential to assess our knowledge at regular time intervals . it is better if we have a platform where we can not only learn a subject but also assess it by taking quiz. In this project we tried to give all material related to a subject such as its syllabus, video modules, lecture notes. It also provides the student to evaluate the understanding level by taking quiz. To evaluate the performance over a period of time we are storing marks of each quiz in our database.

The whole process of assigning test and evaluating their scores after the test, was done manually till date. It is very time consuming. Also it is difficult to keep the answer sheets being generated as well as the maintenance of the record of each examination. The chance of loss of records is high and record searching is difficult. Result processing takes more time and the presence of more invigilators is a must if there are more number of students to write the exam.

Here the students can learn and assess themselves and their scores are recorded and maintained in the database which saves time and improves confidence in students as they assess themselves whenever they want.

**1. INTRODUCTION**

Online quiz is being launched because a need for a destination that is beneficial for both instructors and students. Students can download and share materials for particular subject, can give exams and view their results. This site is an attempt to remove the existing flaws in the manual system of conducting exams.

**1.1 Purpose:**

Our project “A Self learning Platform” fulfills the requirements of the teachers to conduct the quiz online. Only Instructor can add the materials and quizzes for all subject and also can put a notice if they want to display any information.

Students can give exams without the need of going to any physical destination. They can view the result at the same time.

Thus the purpose of this site is to provide a system that saves the efforts and time of both the teachers and students.

**1.2 Scope:**

The SELF LEARNING PLATFORM is to replace the existing systemwith a software solution. This system is accessed through Internet by hitting the URL provided by the organization. And the users who want to use this application should be registered with their details. This system allows both new and old users. And then he has his own user id and account. Then he will be treated as a member of the site and allowed to use the system by reading and giving exams there by assessing himself.

**2. SYSTEM ANALYSIS**

**2.1 Existing System**

Teaching , conducting exams and assessing student often involves lots of paper work, sheets and other manual data entry tasks. There is also little visibility among the guides and student as to what the status of each test is during the course of a semester.

Nowadays many web sites are there like NPTEL, MIT Course, Coursera and Udacity etc.

**2.2 Proposed System**

The SELF LEARNING PLATFORM is to replace the existing systemwith a software solution. This system is accessed through Internet by hitting the URL provided by the organization. And the users who want to use this application should be registered with their details. This system allows both new and old users. And then he has his own user id and account. Then he will be treated as a member of the site and allowed to use the system by reading and giving exams there by assessing himself.

SELF LEARNING PLATFORM

To overcome the problems in the existing system the proposed system is able to track the overall students tasks and saves each team student's time.

**DISCUSSION FORUM**

To overcome the problems in the current messaging system the proposed system provides a discussion form feature integrated in to the teamwork network to keep a running record of all conversations held by members of registered users.

Provides the following features.

* Faster processing.
* Maintaining all information regarding a particular subject.
* Better accuracy in learning a subject.
* Fast retrieval of Information.
* Easy to use and learn.
* Self evaluation.
* Statistics to check the progress.

**3. SYSTEM SPECIFICATIONS**

**3.1. Introduction**

The main aim of the system is to develop a web based program for reading, taking quizzes and displaying discussion purpose of a user and displaying them in GUI format.

This system contains following operations like

Registration

Login

Notices

Discussions

**3.1.1. Product Overview**

The objective of this system is to develop a web application to facilitate discussions. The login screen can be opened by clicking the icon given to the user. Multiple users can access the system simultaneously. Here the subject name is clickable. The user can click one subject one subject at a time. There is user verification, the user must give correct username while login. The username is previously denied and allocated to the user. Upon giving wrong username then it display a text of wrong user name in the login screen. The discussion forums contain unlimited number of queries and its answers sent by different registered users.

**3.2. Specific Requirements**

**3.2.1. External Interface Requirements**

User Interface User interface include various GUI screens.

1. Login screen

This screen contain user name and password, after entered user credentials click on a login button.

2. Registration screen

This will acquire to display username, password, department, date of birth ,email id.

3.Discussions screen

This will display all the discussions and the users can post the comment on this.

**3.2.2. Hardware interface**

**Hardware Requirements:**

* Processor:: Pentium-III (or) Higher
* Ram:: 64MB (or) Higher
* Cache:: 512MB
* Hard disk:: 10GB

**3.2.2 Software Requirements:**

* Technology : Java and J2EE
* Web Technologies : HTML, CSS, JavaScript
* Database : Mysql
* JDK Version : JDK1.5
* Server : Tomcat5.5

3.**Additional Tools:**

HTML Designing : Dream weaver Tool

Development Tool kit : My Eclipse

Internet cable.

**Software interface**

The software interface required are

Server TOMCAT

Browser

Java programming language

**3.2.2. Software Product Feature**

Reading discussions and notices and display in GUI format.

Remotely login.

**3.2.3. Software System Attribute**

**Reliability** This product can effectively run in any environment with free from error.

**Availability** This system is available to user all the time.

**Security** The person whose login account is present in the server is expected to login to the system.

**Maintainability** This system is maintainable for any change or any fault in the future.

**Portability** This system can run in Linux like Ubuntu, OpenSuse platform and can also run anywhere

**Performance** It will provide better performance for its purpose.

**4. FEASIBILITY STUDY:**

The next step in analysis is to verify the feasibility of the proposed system. “All projects are feasible given unlimited resources and infinite time“. But in reality both resources and time are scarce. Project should confirm to time bounce and should be optimal in there consumption of resources. This place a constant is approval of any project.

Feasibility has applied to **Shopping Cart** pertains to the following areas:

* Technical feasibility
* Operational feasibility
* Economical feasibility

**4.1** **TECHNICAL FEASIBILITY:**

To determine whether the proposed system is technically feasible, we should take into consideration the technical issues involved behind the system.

Shopping Cart uses the web technologies, which is rampantly employed these days worldwide. The world without the web is incomprehensible today. That goes to proposed system is technically feasible.

**4.2 OPERATIONAL FEASIBILITY:**

To determine the operational feasibility of the system we should take into consideration the awareness level of the users. This system is operational feasible since the users are familiar with the technologies and hence there is no need to gear up the personnel to use system. Also the system is very friendly and to use.

**4.3. ECONOMIC FEASIBILITY**

To decide whether a project is economically feasible, we have to consider various factors as:

* + - * Cost benefit analysis
      * Long-term returns
      * Maintenance costs

The proposed **Shopping Cart** is computer based. It requires average computing capabilities and access to internet, which are very basic requirements hence it doesn’t incur additional economic overheads, which renders the system economically feasible.

**5. SOFTWARE DESIGN**

**5.1.Introduction:**

**Systems design** is the process or art of defining the architecture,

components, modules, interfaces, and data for a system to satisfy specified requirements. One

could see it as the application of systems theory to product development. There is some

overlap and synergy with the disciplines of systems analysis, systems architecture and

systems engineering.

**Uml Diagrams**

**Unified Modeling Language**:

The Unified Modeling Language allows the software engineer to express an analysis model using the modeling notation that is governed by a set of syntactic semantic and pragmatic rules.

A UML system is represented using five different views that describe the system from distinctly different perspective. Each view is defined by a set of diagram, which is as follows.

* User Model View

This view represents the system from the users perspective.

The analysis representation describes a usage scenario from the end-users perspective.

* Structural model view

In this model the data and functionality are arrived from inside the system.

This model view models the static structures.

* Behavioral Model View

It represents the dynamic of behavioral as parts of the system, depicting the interactions of collection between various structural elements described in the user model and structural model view.

**High Level Design**

**User Interface**

1. User should have minimum knowledge in using a browser (Internet Explorer or Netscape Navigator).

1. User Interacts with the software product through the WebPages and its options.
2. If user wants to use this system he should have the login id and password.

**Actors**

The main actors of the system are.

1. Instructor

2.Student

**5.2. Detailed Design Diagram**

**Primary Model Elements of UML**

**Error! Reference source not found.**

**Node**: A Hardware device upon which software nay reside and/or execute.

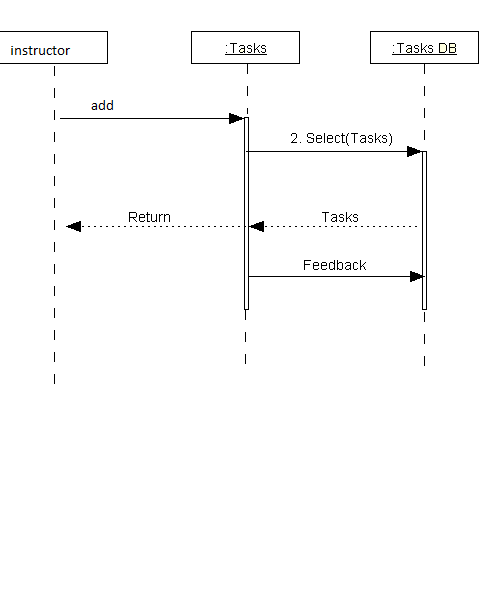
**Note**: A Comment, Explanation or annotation.

`

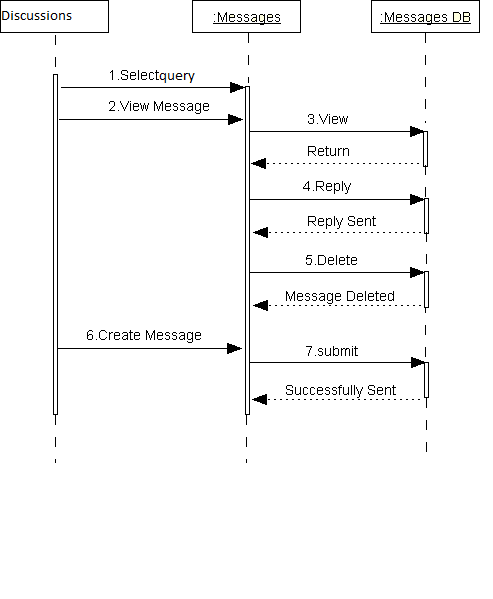
Package: A container of elements.

UML DIAGRAMS

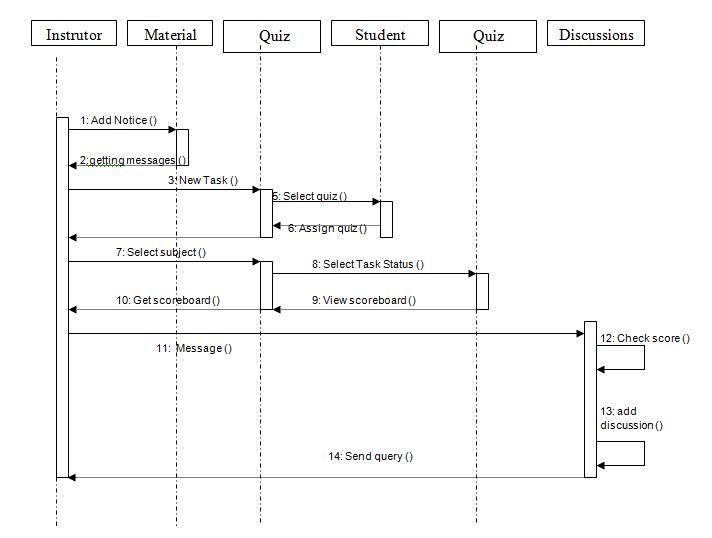
**Sequence -1**



**Sequence-2**

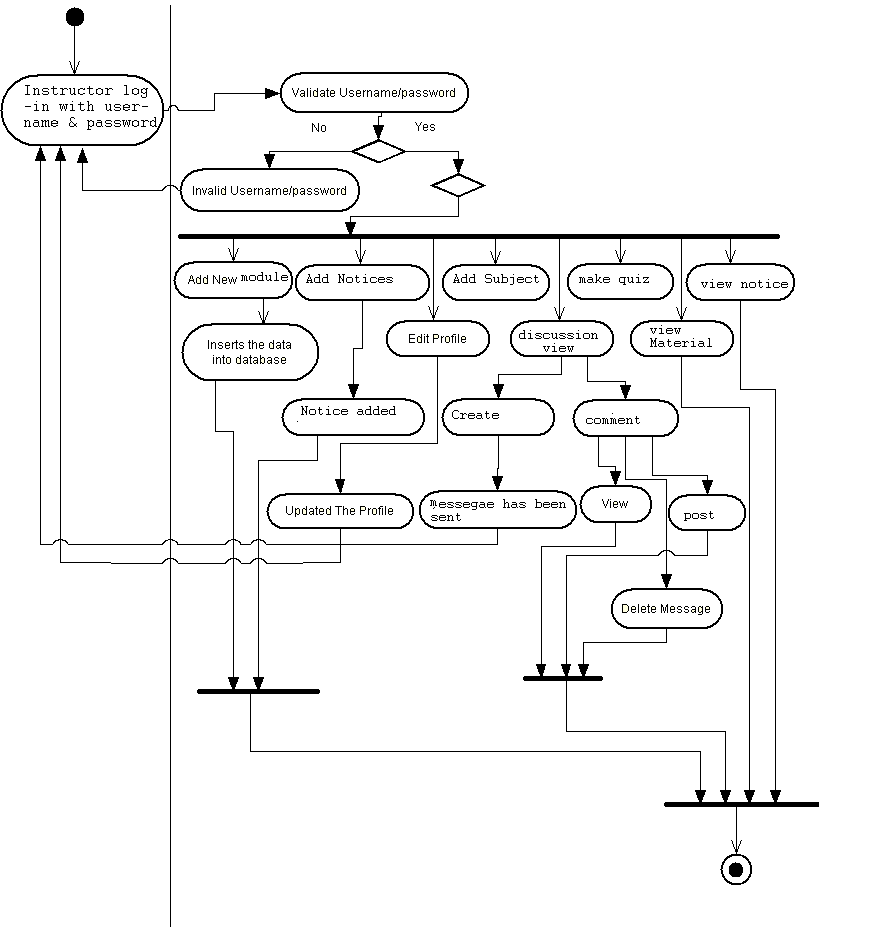
****

**Sequence-3**

****

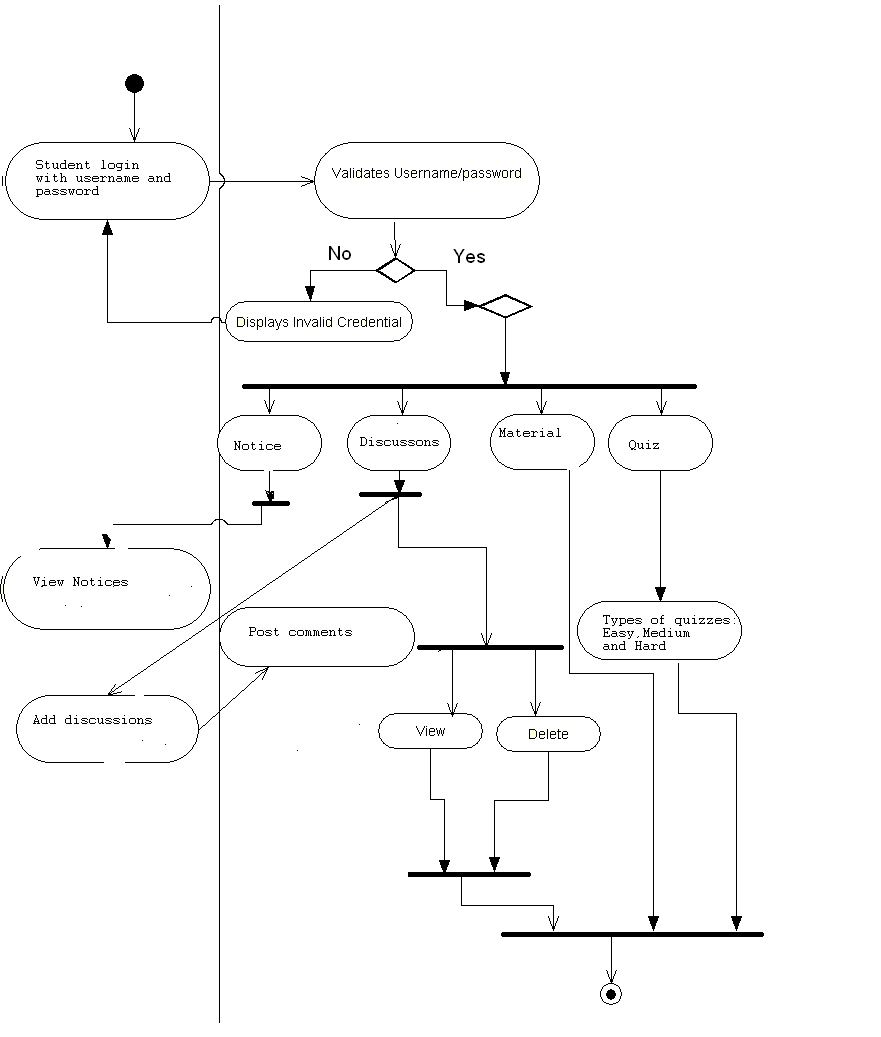
**Activity-1**

**Instructor System**

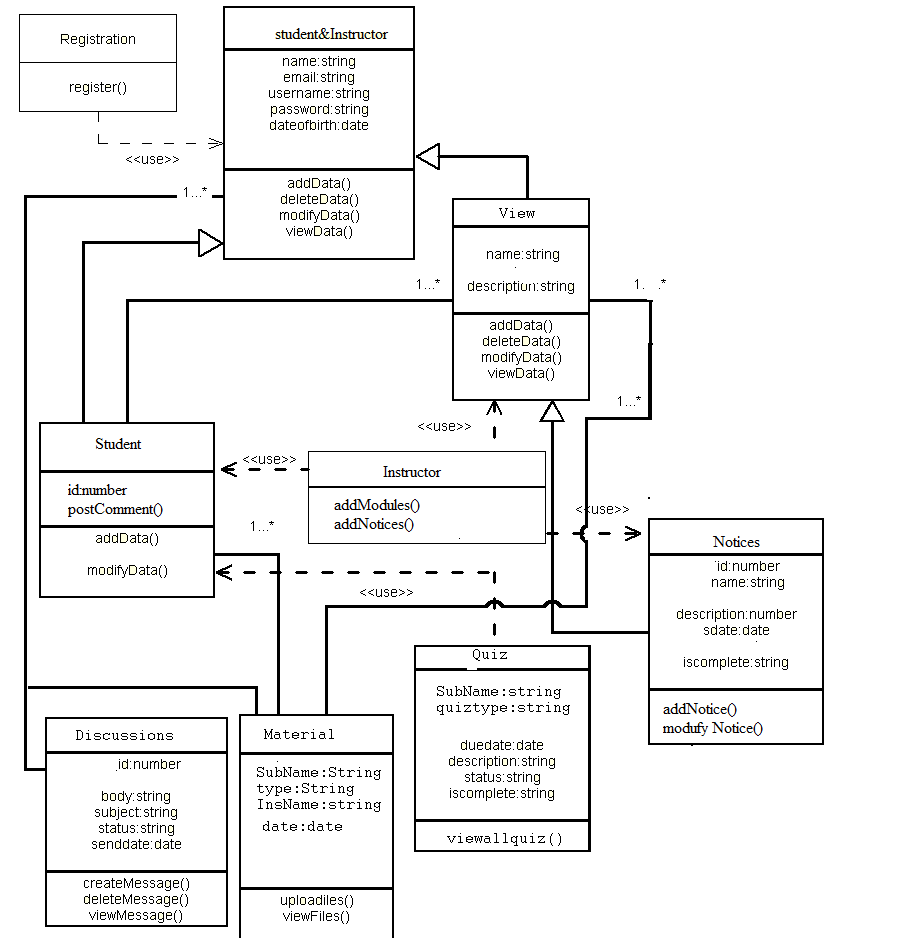


**Activity-2**

**Student System**



**Class Diagram with associations**

****

**Module List**

1. Student or instructor registration

2. Member login & password

3. Add Material

4.Add Notices and Display

5. Quiz

6. Discussion forums

7. Display Result statistics

**6.TECHNOLOGY DESCRIPTION**

**6.1 Introduction to java**

**About Java**:

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

* Java is a programmer’s language
* Java is cohesive and consistent
* Except for those constraint imposed by the Internet environment. Java gives the programmer, full control

Finally Java is to Internet Programming where c was to System Programming.

**Importance of Java to the Internet**

Java has had a profound effect on the Internet. This is because; java expands the Universe of objects that can move about freely in Cyberspace. In a network, two categories of objects are transmitted between the server and the personal computer. They are passive information and Dynamic active programs. in the areas of Security and probability. But Java addresses these concerns and by doing so, has opened the door to an exciting new form of program called the Applet.

**Applications and applets***.*

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

**Java Architecture**

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

# Compilation of code

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

**Compiling and interpreting java source code.**

**Source code**

**Pc compiler**

**Macintosh compiler**

**SPARC Compiler**

**Java Byte code**

**Platform independent**

**Java interpreter**

**Java interpretermacintosh**

**)))**

**Java interpreter(SPARC)**

During run-time the Java interpreter tricks the byte code file into thinking that it is running on a Java Virtual Machine. In reality this could be an Intel Pentium windows 95 or sun SPARCstation running Solaris or Apple Macintosh running system and all could receive code from any computer through internet and run the Applets.

**Simple**:

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

### Object oriented

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

### Robust

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

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

**6. 2 Servlets/JSP**

**Introduction**

A Servlet Is a generic server extension. a Java class that can be loaded

Dynamically to expand the functionality of a server.Servlets are commonly used with web servers. Where they can take the place CGI scripts.

A servlet is similar to proprietary server extension, except that it runs inside a Java Virtual Machine (JVM) on the server, so it is safe and portable

Servlets operate solely within the domain of the server.

Unlike CGI and Fast CGI, which use multiple processes to handle separate program or separate requests, separate threads within web server process handle all servlets. This means that servlets are all efficient and scalable.

Servlets are portable; both across operating systems and also across web servers. Java Servlets offer the best possible platform for web application development.

Servlets are used as replacement for CGI scripts on a web server, they can extend any sort of server such as a mail server that allows servelts t extend its functionality perhaps by performing a virus scan on all attached documents or handling mail filtering tasks.

Servlets provide a Java-based solution used to address the problems currently associated with doing server-side programming including inextensible scripting solutions platform-specific API’s and incomplete interface.

Servlets are objects that conform to a specific interface that can be plugged into a Java-based server.Servlets are to the server-side what applets are to the server-side what applets are to the client-side-object byte codes that can be dynamically loaded off the net. They differ form applets in than they are faceless objects(with out graphics or a GUI component).They serve as platform independent, dynamically loadable,plugable helper byte code objects on the server side that can be used to dynamically extend server-side functionality.

For example an HTTP servlet can be used to generate dynamic HTML content when you use servlets to do dynamic content you get the following advantages:

* They’re faster and cleaner then CGI scripts
* They use a standard API( the servlet API)
* They provide all the advantages of Java (run on a variety of servers without needing to be rewritten)

**A t t r a c t i v e n e s s o f S e r v l e t s:**

They are many features of servlets that make them easy and attractive to tuse these include:

* Easily configure using the GUI-based Admin tool]
* Can be Loaded and Invoked from a local disk or remotely across the network.
* Can be linked together or chained, so that on servlet can call another servlet, or several servlets in sequence.
* Can be called dynamically from with in HTML, pages using server-side include-tags.
* Are secure-even when downloading across the network, the servlet security model and servlet and box protect your system from unfriendly behavior.,

**Advantages of the servlet API**

One of the great advantages of the servlet API is protocol independent. It assumes nothing about:

* The protocol being used to transmit on the net
* How it is loaded
* The server environment it will be running in
* These quantities are important, because it allows the Servlet API to be embedded in many different kinds of servers.There are other advantages to the servelt API as well These include:
* It’s extensible-you can inherit all your functionality from the base classes made available to you
* It’s simple small, and easy to use.

**Features of Servlets:**

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

Servlets are classes and interfaces from two packages,javax .servlet and javax.servlet.http.The java.servlet package contains classes t support generic, protocol-independent servlets.The classes in the javax.servelt.http package To and HTTP specific functionality extend these classes

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

Unlike a java program, a servlet does not have a main() method,Instead the server in the process of handling requests invoke certain methods of a servlet.Each time the server dispatches a request to a servlet, it invokes the servelts Service() method,

A generic servlet should override its service() method to handle requests as appropriate for the servlet.The service() accepts two parameters a request object and a response object .The request object tells the servlet about the request, while the response object is used to return a response

InContrast.anHttp servlet usually does not override the service() method.Instead it overrides doGet() to handle GET requests and doPost() to handle Post requests. An Http servlet can override either or both of these modules the service() method of HttpServlet handles the setup and dispatching to all the doXXX() methods.which iswhy it usually should not be overridden

The remainders in the javax.servlet and javax.servlet.http.package are largely support classes .The ServletRequest and ServletResponse classes in javax.servlet provide access to generic server requests and responses while HttpServletRequest and HttpServletResponse classes in javax.servlet provide access to generic server requests and responses while HttpServletRequest and HttpServletResponse in javax.servlet.http provide access a HTTP requests and responses . The javax.servlet.http provide contains an HttpSession class that provides built-in session tracking functionality and Cookie class that allows quickly setup and processing HttpCookies.

**Loading Servlets:**

Servlets can be loaded from their places. From a directory that is on the CLASSPATH. The CLASSPATH of the JavaWebServer includes service root/classes/, which is where the system classes reside

From the <SERVICE\_ROOT/servlets/directory.This is not in the server’s classpath. A class loader is used to create servlets form this directory.New servlets can be added-existing servlets can be recompiled and the server will notice these changes. From a remote location.For this a code base like is required in addtion to the servlet’s class name.Refer to the admin Gui docs on servlet section to see how to set this up.

Loading Remote Servlets

Remote servlets can be loaded by:

* Configuring the admin Tool to setup automatic loading of remote servlets.
* Selecting up server side include tags in .html files
* Defining a filter chain Configuration

**Invoking Servlets**

A servlet invoker is a servlet that invokes the “server” method on a named servlet.If the servlet is not loaded in the server,then the invoker first loades the servlet(either form local disk or from the network) and the then invokes the “service” method.Also like applets,local servlets in the server can be identified by just the class name.In other words, if a servlet name is not absolute.it is treated as local.

A Client can Invoke Servlets in the Following Ways:

* client(browser) can invoke the servlet directly using a URL, once it has been mapped using the SERVLET The client can ask for a document that is served by the servlet.

The ALIASES Section of the admin GUI

* The servlet can be invoked through server side include tags.
* The servlet can be invoked by placing it in the servlets/directory
* The servlet can be invoked by using it in a filter chain

**The Servlet Life Cycle:-**

The Servlet life cycle is one of the most exciting features of Servlets.This life cycle is a powerful hybrid of the life cycles used in CGI programming and lower-level NSAPI and ISAPI programming.

The servlet life cycle allows servlet engines to address both the performance and resource problems of CGI and the security concents of low level server API programming.

Servlet life cycle is highly flexible Servers hava significant leeway in how they choose to support servlets.The only hard and fast rule is that a servlet engine must confor to the following life cycle contact:

* Create and initialize the servlets
* Handle zero or more service from clients
* Destroy the servlet and then garbage Collects it.

It’s perfectly legal for a servlet t be loaded, created an initialzed in its own JVM,only to be destroyed an dgarbage collected without hancdling any clientrequest or after handling just one request

The most common and most sensible life cycle implemntations for HTTP servelts are:

Single java virtual machine and astatine persistence.

**Init and Destroy**:-

Just like Applets servlets can define init() and destroy() methods, A servlets init(ServiceConfig) method is called by the server immediately after the server constructs the servlet’s instance.Depanding on the server and its configuration, this can be at any of these times

* When the server states
* When the servlet is first requested, just before the service() method is invoked
* At the request of the server administrator

In any case, nit() is guaranteed to be called before the servlet handles its first request

The init() method is typically used to perform servlet initialization creating or loading objects that are used by the servlet in handling of its request. In order to providing a new servlet any information about itself and its environment, a server has to call a servelts init() method and pass an object that implement the ServletConfig interface.

This ServletConfig object supplies a servlet with information about its initialization parameters.These parameters are given to the servlets and are not associated with any single request.They can specify initial values, such as where a counter should begin counting, or default values, perhaps a template to use when not specified by the request,

The server calls a servlet’s destroy() method when the servlet is about to be unloaded. In the destroy() method, a servlet should free any resources it has acquired that will not be garbage collected. The destroy() method also gives a servlet a chance to write out its unsaved. cached information or any persistent information that should be read during the next call to init().

**Session Tracking:**

HTTP is a stateless protocol, it provides no way for a server to recognize that a sequence of requests is all from the same client. This causes a problem for application such as shopping cart applications. Even in chat application server can’t know exactly who’s making a request of several clients.

The solution for this is for client to introduce itself as it makes each request, Each clients needs to provide a unique identifier that lets the server identify it, or it needs to give some information that the server can use to properly handle the request, There are several ways to send this introductory information with each request Such as:

**User authorization:**

**One way to perform session tracking is to leverage the information that comes with**

User authorization. When a web server restricts access to some of its resources to only those clients that log in using a recognized username and password. After the client logs in, the username is available to a servlet through getRemoteUser ()

Wean use the username to track the session. Once a user has logged in, the browser remembers her username and resends the name and password as the user views new pages on the site. A servlet can identify the user through her username and they’re byTrack her session.The biggest advantage of using user authorization to perform session tracking is that it’s easy to implement. Simply tell the protect a set of pages, and use getRemoteUser() to identify each client.Another advantage is that the technique works even when the user accesses your site form or exists her browser before coming back.

The biggest disadvantage of user authrization is that it requires each user to register for an account and then log in in each time the starts visiting your site. Most users will tolerate registering and lagging in as a necessary evil when they are accessing sensitive information, but its all overkill for simple session tracking.Other problem with user authorization is that a user cannot simultaneously maintain more than one session at the same site.

**Hidden Form Fields:**

One way to support anonymous session tracking is to use hidden from fields. As the name implies, these are fields added to an HTML, form that are not displayed in the client’s browser, They are sent back to the server when the form that contains them is submitted.

In a sense, hidden form fields define constant variables for a form. To a servlet receiving a submitted form, there is no difference between a hidden fields and a visible filed.

As more and more information is associated with a clients session . It can become burdensome to pass it all using hidden form fields. In these situations it’s possible to pass on just a unique session ID that identifies as particular clients session.

That session ID can be associated with complete information about its session that is stored on the server.

The advantage of hidden form fields is their ubiquity and support for anonymity. Hidden fields are supported in all the popular browsers, they demand on special server requirements, and they can be used with clients that haven’t registered or logged in.

The major disadvantage with this technique, however is that works only for a sequence of dynamically generated forms, The technique breaks down immediately with static documents, emailed documents book marked documents and browser shutdowns.

**URL Rewriting:**

URL rewriting is another way to support anonymous session tracking, With URL rewriting every local URL the user might click on is dynamically modified. or rewritten, to include extra, information. The extra information can be in the deform of extra path information, added parameters, or some custom, server-specific.URL change. Due to the limited space available in rewriting a URL, the extra information is usually limited to a unique session.

Each rewriting technique has its own advantage and disadvantage

Using extra path information works on all servers, and it works as a target for forms that use both the Get and Post methods. It does not work well if the servlet has to use the extra path information as true path information

The advantages and disadvantages of URL.rewriting closely match those of hidden form fileds,The major difference is that URL rewriting works for all dynamically created documents, such as the Help servlet, not just forms. With the right server support, custom URL rewriting can even work for static documents.

**Persistent Cookies:**

A fourth technique to perform session tracking involves persistent cookies. A cookie is a bit of information. sent by a web server to a browser that can later be read back form that browser. When a browser receives a cookie, it saves the cookie and there after sends the cookie back to the server each time it accesses a page on that server, subject to certain rules. Because a cookie’s value can uniquely identify a client, cookies are often used for session tracking.

Persistent cookies offer an elegant, efficient easy way to implement session tracking. Cookies provide as automatic an introduction for each request as we could hope for. For each request, a cookie can automatically provide a client’s session ID or perhaps a list of clients performance. The ability to customize cookies gives them extra power and versatility.

The biggest problem with cookies is that browsers don’t always accept cookies sometimes this is because the browser doesn’t support cookies. More often its because

The browser doesn’t support cookies. More often its because the user has specifically configured the browser to refuse cookies.

The power of serves:

The power of servlets is nothing but the advantages of servlets over other approaches, which include portability, power, efficiency, endurance, safety elegance,integration,extensibility and flexibility.

**Portability:**

As servlets are written in java and conform to a well defined and widely accepted API.they are highly portable across operating systems and across server implementation

We can develop a servlet on a windows NT machine running the java web server and later deploy it effortlessly on a high-end Unix server running apache. With servlets we can really “write once, serve every where”

Servlet portability is not the stumbling block it so often is with applets, for two reasons

First,Servlet portability is not mandatory i.e. servlets has to work only on server machines that we are using for development and deployment

Second, servlets avoid the most error-prone and inconstancy implemented portion of the java languages.

**Power:**

Servlets can harness the full power of the core java. API’s: such as Networking and Url access, multithreading, image manipulation, data compression, data base connectivity, internationalization, remote method invocation(RMI) CORBA connectivity, and object serialization, among others,

**Efficiency And Endurance:**

Servlet invocation is highly efficient, Once a servlet is loaded it generally remains in the server’s memory as a single object instance, There after the server invokes the servelt to handle a request using a simple, light weighted method invocation .Unlike the CGI, there’s no process to spawn or interpreter to invoke, so the servlet can begin handling the request almost immediately, Multiple, concurrent requests are handled the request almost immediately. Multiple, concurrent requests are handled by separate threads, so servlets are highly scalable.

Servlets in general are enduring objects. Because a servlets stays in the server’s memory as a single object instance. it automatically maintains its state and can hold onto external resources, such as database connections.

**Safety:**

Servlets support safe programming practices on a number of levels.

As they are written in java,servlets inherit the strong type safety of the java language. In addition the servlet API is implemented to be type safe. Java’s automatic garbage collection and lack of pointers mean that servlets are generally safe from memory management problems like dangling pointers invalid pointer references and memory leaks.

Servlets can handle errors safely, due to java’s exception – handling mechanism. If a servlet divides by zero or performs some illegal operations, it throws an exception that can be safely caught and handled by the server.

A server can further protect itself from servlets through the use of java security manager.A server can execute its servlets under the watch of a strict security manager.

**Elegance:**

The elegance of the servlet code is striking .Servlet code is clean, object oriented modular and amazingly simple one reason for this simplicity is the served API itself. Which includes methods and classes to handle many of the routine chores of servlet development. Even advanced to operations like cookie handling and session tracking tracking are abstracted int convenient classes.

**Integration:**

Servlets are tightly integrated with the server. This integration allows a servlet to cooperate with the server in two ways . for e.g.: a servlet can use the server to translate file paths, perform logging, check authorization, perform MIME type mapping and in some cases even add users to the server’s user database.

**Extensibility and Flexibility:**

The servlet API is designed to be easily extensible. As it stands today the API includes classes that are optimized for HTTP servlets.But later it can be extended and optimized for another type of servlets.It is also possible that its support for HTTP servlets could be further enhanced.

Servlets are also quite flexible, Sun also introduced java server pages. which offer a way to write snippets of servlet code directly with in a static HTML page using syntax similar to Microsoft’s Active server pages(ASP)

**6. 3 JDBC**

**What is JDBC?**

any relational database. One can write a single program using the JDBC API,and the JDBC is a Java Api for executing SQL,Statements(As a point of interest JDBC is trademarked name and is not an acronym; nevertheless,Jdbc is often thought of as standing for Java Database Connectivity. It consists of a set of classes and interfaces written in the Java Programming language.JDBC provides a standard API for tool/database developers and makes it possible to write database applications using a pure Java API

**What Does JDBC Do?**

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

* Establish a connection with a database
* Send SQL statements
* Process the results
* JDBC Driver Types
* The JDBC drivers that we are aware of this time fit into one of four categories
* JDBC-ODBC Bridge plus ODBC driver
* Native-API party-java driver
* JDBC-Net pure java driver
* Native-protocol pure Java driver

An individual database system is accessed via a specific JDBC driver that implements the java.sql.Driver interface. Drivers exist for nearly all-popular RDBMS systems, through few are available for free. Sun bundles a free JDBC-ODBC bridge driver with the JDK to allow access to a standard ODBC,data sources, such as a Microsoft Access database, Sun advises against using the bridge driver for anything other than development and very limited development.

**Type 01-JDBC-ODBC Bridge Driver**

Type 01 drivers use a bridge technology to connect a java client to an ODBC database service. Sun’s JDBC-ODBC bridge is the most common type 01 driver. These drivers implemented using native code.

**Type 02-Native-API party-java Driver**

Type 02 drivers wrap a thin layer of java around database-specific native code libraries for Oracle databases, the native code libraries might be based on the OCI(Oracle call Interface) libraries, which were originally designed for **c/c++** programmers, Because type-02 drivers are implemented using native code. in some cases they have better performance than their all-java counter parts. They add an element of risk, however, because a defect in a driver’s native code section can crash the entire server

**Type 03-Net-Protocol All-Java Driver**

Type 03 drivers communicate via a generic network protocol to a piece of custom middleware. The middleware component might use any type of driver to provide the actual database access. These drivers are all java, which makes them useful for applet deployment and safe for servlet deployment

**Type-04-native-protocol All-java Driver**

Type o4 drivers are the most direct of the lot. Written entirely in java, Type 04 drivers understand database-specific networking. protocols and can access the database directly without any additional software

**JDBC-ODBC Bridge**

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

**WHAT IS The JDBC-ODBE Bridge ?**

The JDBC-ODBC Bridge is a Jdbc driver, which implements JDBC operations by translating them into ODBC operations. To ODBC it appears as a normal application program. The Bridge is implemented as the sun.jdbc.odbc Java package and contains a native library used to access ODBC.The Bridge is joint development of Intersolv and Java Soft

**6. 4 MySQL**

The database has become an integral part of almost every human's life. Without it, many things we do would become very tedious, perhaps impossible tasks. Banks, universities, and libraries are three examples of organizations that depend heavily on some sort of database system. On the Internet, search engines, online shopping[http://images.intellitxt.com/ast/adTypes/mag-glass_10x10.gif](http://www.devshed.com/), and even the website naming convention (http://www...) would be impossible without the use of a database. A database that is implemented and interfaced on a computer is often termed a database server.  
  
One of the fastest SQL (Structured Query Language) database servers currently on the market is the MySQL server.MySQL, available for download, offers the database programmer with an array of options and capabilities rarely seen in other database servers. What's more, MySQL is free of charge for those wishing to use it for private and commercial use. Those wishing to develop applications specifically using MySQL should consult MySQL's licensing section, as there is a charge for licensing the product.  
  
These capabilities range across a number of topics, including the following:

* Ability to handle an unlimited number of simultaneous users.
* Capacity to handle 50,000,000+ records.
* Very fast command execution, perhaps the fastest to be found on the market.
* Easy and efficient user privilege system.

A database is really nothing more than a hierarchy of increasingly complex data structures. In MySQL, the acknowledged structure for holding blocks (or **records**) of information is called the **table**.

These records, in turn, are made up of the smallest object that can be manipulated by the user, known as the **data type**. Together, one or more of these data types form a record. A table holds the collection of records that make up part of the database. We can consider the hierarchy of a database to be that of the following:

Database < Table < Record < Datatype

Datatypes come in several forms and sizes, allowing the programmer to create tables suited for the scope of the project. The decisions made in choosing proper datatypes greatly influence the performance of a database, so it is wise to have a detailed understanding of these concepts.

**MySQL Datatypes**

MySQL is capable of many of the datatypes that even the novice programmer has probably already been exposed to. Some of the more commonly used include:

**CHAR (M)**  
CHAR's are used to represent fixed length strings. A CHAR string can range from 1-255 characters. In later table creation, an example CHAR datatype would be declared as follows:

ex.  
car\_model CHAR(10);

**VARCHAR (M)**  
VARCHAR is a more flexible form of the CHAR data type. It also represents data of type String, yet stores this data in variable length format. Again, VARCHAR can hold 1-255 characters. VARCHAR is usually a wiser choice than CHAR, due to it's variable length format characteristic. Although, keep in mind that CHAR is much faster than VARCHAR, sometimes up to 50%.  
(A CHAR stores the whole length of the declared variable, regardless of the size of the data contained within, whereas a VARCHAR only stores the length of the data, thus reducing size of the database file.)

ex.  
car\_model VARCHAR(10);

**INT (M) [Unsigned]**  
The INT datatype stores integers ranging from -2147483648 to 2147483647. An optional "unsigned" can be denoted with the declaration, modifying the range to be 0 to 4294967295

ex.  
light\_years INT;  
Valid integer: '-24567'.    Invalid integer: '3000000000'.

ex.  
light\_years INT unsigned;  
Valid integer: '3000000000'.    Invalid integer: '-24567'.

**FLOAT [(M,D)]**  
A FLOAT represents small decimal numbers, used when a somewhat more precise representation of a number is required.

ex.  
rainfall FLOAT (4,2);  
This could be used to represent rainfall average in centimeters per year, which could be a decimal value. More specifically, FLOAT (4,2) states the fact that rainfall can hold up to four characters and two decimal places. Thus,

42.35 is valid, accurately represented.  
324.45 is invalid, rounded to 324.5.  
2.2 is valid, accurately represented.  
34.542 is invalid, rounded to 34.54.

*Note: Due to the fact that FLOAT is rounded, those wishing to represent money values would find it wise to use* ***DECIMAL****, a datatype found within MySQL that does not round values. Consult the documentation for a complete explanation.*

**DATE**   
Stores date related information. The default format is 'YYYY-MM-DD', and ranges from '0000-00-00' to '9999-12-31'. MySQL provides a powerful set of date formatting and manipulation commands, too numerous to be covered within this article. However, one can find these functions covered in detail within the MySQL documentation.

the\_date DATE;

**TEXT / BLOB**  
The text and blob datatypes are used when a string of 255 - 65535 characters is required to be stored. This is useful when one would need to store an article such as the one you are reading. However, there is no end space truncation as with VARCHAR AND CHAR. The only difference between BLOB and TEXT is that TEXT is compared case insensitively, while BLOB is compared case sensitively.

**SET**  
A datatype of type string that allows one to choose from a designated set of values, be it one value or several values. One can designate up to 64 values.

ex.  
transport SET ("truck", "wagon") NOT NULL;

From the above declaration, the following values can be held by transport:

""  
"truck"  
"wagon"  
"truck,wagon"

**ENUM**  
A datatype of type string that has the same characteristics as the SET datatype, but only one set of allowed values may be chosen. Usually only takes up one byte of space, thus saving time and space within a table.

ex.  
transport ENUM ("truck", "wagon") NOT NULL;

From the above declaration, the following values can be held by transport:

""  
"truck"  
"wagon"

**Records**

Together, a group of declared datatypes form what is known as a record. A record can be as small as one data variable, or as many as deemed needed. One or more records form the structure of a table.

**The Bigger Picture: Tables**

Before we can execute commands on the database, we must first create a table in which data can be stored. This is accomplished in the following manner:

mysql> CREATE TABLE test (  
> name VARCHAR (15),  
> email VARCHAR (25),  
> phone\_number INT,  
> ID INT NOT NULL AUTO\_INCREMENT,  
> PRIMARY KEY (ID));

Ensuing output:

Query OK, 0 rows affected (0.10 sec)  
mysql>

The first table in your database has now been created. *Note: no two tables can have the same name.*  
*Note(2): Each dataspace is more often referred to as a* ***column****.*

**Column Characteristics:**

* A name may not be made up of strictly numbers.
* A name may start with a number.
* A name may be up to 64 characters.

**Other table options:**

The following options can be placed after any datatype, adding other characteristics and capabilities to them.

* Primary Key. Used to differentiate one record from another. No two records can have the same primary key. This is obviously useful when it is imperative that no two records are mistaken to be the other.
* Auto\_Increment. A column with this function is automatically incremented one value (previous + 1) when an insertion is made into the record. The datatype is automatically incremented when 'NULL' is inserted into the column.
* NOT NULL. Signifies that the column can never be assigned a NULL value.

ex.  
soc\_sec\_number INT PRIMARY KEY;  
No two soc\_sec\_number records can hold the same value.

ID\_NUMBER INT AUTO\_INCREMENT;

**Insertion of records**

**Note:** The originally created table, test, created in the last section will be used to illustrate the examples in this section. Here it is again, for quick reference:

mysql> CREATE TABLE test (

> name VARCHAR (15),

> email VARCHAR (25),

> phone\_number INT,

> ID INT NOT NULL AUTO\_INCREMENT,

> PRIMARY KEY (ID));

Insertion of data into the table is accomplished, logically enough, using the INSERT command.

mysql> INSERT INTO test VALUES

mysql> ('Bugs Bunny', 'carrots@devshed.com',

mysql> 5554321, NULL);

Result, assuming the command was correctly entered:

Query OK, 1 row affected (0.02 sec)

mysql>

**Selection**

A database would not be much use if one was not able to search and extract data from it. In MySql terms, this is accomplished through the SELECT statement.

mysql> SELECT \* FROM test

mysql> WHERE (name = "Bugs Bunny");

Result:

|  |  |  |  |
| --- | --- | --- | --- |
| **name** | **email** | **phone** | **ID** |
| Bugs Bunny | carrots@devshed.com | 5554321 | 1 |

Let's assume we have inserted four differing records, all bearing the same name of "Bugs Bunny", yet having different email addresses and phone numbers. The table test, would look somewhat like the following:

|  |  |  |  |
| --- | --- | --- | --- |
| **name** | **email** | **phone** | **ID** |
| Bugs Bunny | carrots@devshed.com | 5554321 | 1 |
| Bugs Bunny | peppers@devshed.com | 5554331 | 2 |
| Bugs Bunny | lettuce@devshed.com | 5554341 | 3 |
| Bugs Bunny | celery@devshed.com | 5554351 | 4 |

**Deletion**

One can also delete records inserted into the table. This is accomplished through the DELETE command.

mysql> DELETE FROM test

mysql> WHERE (name = "Bugs Bunny");

Result:  
This would result in the deletion of all records within the table test containing name "Bugs Bunny".

Another example:

mysql> DELETE FROM test

mysql> WHERE (phone\_number = 5554321);

Result: (Using the previously illustrated example)

|  |  |  |  |
| --- | --- | --- | --- |
| **name** | **email** | **phone** | **ID** |
| Bugs Bunny | peppers@devshed.com | 5554331 | 2 |
| Bugs Bunny | lettuce@devshed.com | 5554341 | 3 |
| Bugs Bunny | celery@devshed.com | 5554351 | 4 |

**Modification**

MySQL also has the capability of modifying data already entered into the table. This is accomplished through the UPDATE command.

mysql> UPDATE test SET name = 'Daffy Duck'

mysql> WHERE name = "Bugs Bunny";

|  |  |  |  |
| --- | --- | --- | --- |
| **name** | **email** | **phone** | **ID** |
| Daffy Duck | peppers@devshed.com | 5554331 | 2 |
| Daffy Duck | lettuce@devshed.com | 5554341 | 3 |
| Daffy Duck | celery@devshed.com | 5554351 | 4 |

This section, we covered the core [MySQL database[http://images.intellitxt.com/ast/adTypes/mag-glass_10x10.gif](http://www.devshed.com/)](http://www.devshed.com/) manipulation functions, basic insertion, deletion, modification, and search. The next section will elaborate on these capabilities, providing extended functioning and flexibility when manipulating the database.

What we have covered so far is but a small part of what MySQL is capable of. Let's delve a little deeper into the language, exploring some of the more advanced commands of the language.

**Logical Operations**

MySQL includes full support of all basic logical operations.

**AND (&&)**

mysql> SELECT \* FROM test WHERE

mysql> (name = "Bugs Bunny") AND

mysql> (phone\_number = 5554321);

Result:  
All records containing the name "Bugs Bunny" AND the phone number '5554321' will be displayed to the screen.

**OR ( || )**

mysql> SELECT \* FROM test WHERE

mysql> (name = "Bugs Bunny") OR

mysql> (phone\_number = 5554321);

Result:  
All records containing the name "Bugs Bunny" OR the phone number '5554321' will be displayed to the screen.

**NOT ( ! )**

mysql> SELECT \* FROM test WHERE

mysql> (name != "Bugs Bunny");

Result:  
All records NOT containing the name "Bugs Bunny" will be displayed to the screen.

**Order By**

mysql> SELECT \* FROM test WHERE

mysql> (name = "Bugs Bunny") ORDER BY

mysql> phone\_number;

Result:  
All records containing the name "Bugs Bunny" will be displayed to the screen, ordered in respect to the phone\_number.

**Search functions**

MySQL offers the user the ability to perform both general and specific searches on data.

mysql> SELECT \* FROM test WHERE

mysql> (name LIKE "%gs Bunny");

Result:  
All records containing the partial string "gs Bunny" will be displayed to the screen. This would include such names as: "Bugs Bunny", "ags Bunny", "gs Bunny", and "234rtgs Bunny".

Notice that "LIKE" has been used instead of the equals sign (=). "LIKE" signifies that one is searching for an estimate of the data requested, and not necessarily an exact copy.

The '%' sign could be placed anywhere within the string. The method in which the server searches for a string is dependent upon where one places the '%' sign.

mysql> SELECT \* FROM test WHERE

mysql> (name LIKE "Bugs Bunny%");

Result:  
All records containing the partial string "Bugs Bunny" will be displayed to the screen. This would include such names as: "Bugs Bunnys", "Bugs Bunnyyyy453", "Bugs Bunnytrtrtrtrtr", but not "gs Bunny".

**Focused Search Results**

One can also perform searches and display only certain columns.

mysql> SELECT name FROM test WHERE

mysql> (name = "Bugs Bunny");

Result:

|  |
| --- |
| **name** |
| Bugs Bunny |

**Alter table**

Another very important function of MySQL is the ability to modify previously created tables. This is accomplished via the ALTER statement. This function allows one to add, modify, and delete columns, as well as rename the table, among other functions.

Example: Rename the table

mysql> ALTER table test RENAME mytest;

Example: Add a column

mysql> ALTER table mytest ADD birthday DATE;

Example: Modify a column

mysql> ALTER table mytest CHANGE

mysql> name newname VARCHAR (25);

Example: Delete a column

mysql> ALTER table mytest DROP newname;

Executing the above four functions would modify test, creating the following table:

mysql> TABLE mytest (

> email VARCHAR (25),

> phone\_number INT,

> ID INT AUTO\_INCREMENT,

> birthday DATE );

**5. 5 HTML**

Html is a language which is used to create web pages with html marking up a page to indicate its format, telling the web browser where you want a new line to begin or how you want text or images aligned and more are possible.

We used the following tags in our project.

**TABLE:**

Tables are so popular with web page authors is that they let you arrange the elements of a web page in such a way that the browser won’t rearrange them web page authors frequently use tables to structure web pages.

**<TR>:**

**<**TR**>** is used to create a row in a table encloses <TH> and <TD> elements. <TR> contain many attributes. Some of them are,

* ALIGN: specifies the horizontal alignment of the text in the table row.
* BGCOLOR: Specifies the background color for the row.
* BORDERCOLOR: Sets the external border color for the row.
* VALIGN: Sets the vertical alignment of the data in this row.

**<TH>:**

<TH> is used to create table heading.

* ALIGN: Sets the horizontal alignment of the content in the table cell. Sets LEFT, RIGHT, CENTER.
* BACKGROUND: Species the back ground image for the table cell.
* BGCOLOR: Specifies the background color of the table cell
* VALIGN: Sets the vertical alignment of the data. Sets to TOP, MIDDLE, BOTTOM or BASELINE.
* WIDTH: Specifies the width of the cell. Set to a pixel width or a percentage of the display area.

**<TD>:**

<TD> is used to create table data that appears in the cells of a table.

* ALIGN: Species the horizontal alignment of content in the table cell. Sets to LEFT, CENTER, RIGHT.
* BGCOLOR: Specifies the background image for the table cell.
* BGCOLOR: sets the background color of the table cells.
* WIDTH: Species the width of the cell

**FRAMES:**

Frames are used for either run off the page or display only small slices of what are supposed to be shown and to configure the frame we can use <FRAMESET>there are two important points to consider when working with <FRAMESET>.

* <FRAMESET> element actually takes the place of the <BODY> element in a document.
* Specifying actual pixel dimensions for frames.

<FRAME> Elements are used to create actual frames.

From the frameset point of view dividing the browser into tow vertical frames means creating two columns using the <FRAMESET> elements COLS attribute.

The syntax for vertical fragmentation is,

<FRAMESET COLS =”50%, 50%”>

</FRAMESET>

Similarly if we replace COLS with ROWS then we get horizontal fragmentation.

The syntax for horizontal fragmentation is,

<FRAMESET ROWS=”50%, 50%”>

</FRAMESET>

**FORM:**

The purpose of FORM is to create an HTML form; used to enclose HTML controls, like buttons and text fields.

**ATTRIBUTES:**

* ACTION: Gives the URL that will handle the form data.
* NAME: Gives the name to the form so you can reference it in code set to an alphanumeric string.
* METHOD: method or protocol is used to sending data to the target action URL. The GET method is the default, it is used to send all form name/value pair information in an URL. Using the POST method, the content of the form are encoded as with the GET method, but are sent in environment variables.

## CONTROLS IN HTM

**<**INPUT TYPE =BUTTON>:

Creates an html button in a form.

ATTRIBUTES:

* NAME: gives the element a name. Set to alphanumeric characters.
* SIZE: sets the size.
* VALUE: sets the caption of the element.

**<**INPUT TYPE = PASSWORD>:

Creates a password text field, which makes typed input.

ATTRIBUTES:

* NAME: gives the element a name, set to alphanumeric characters.
* VALUE: sets the default content of the element.

<INPUT TYPE=RADIO>:

**C**reates a radio button in a form.

ATTRIBUTE:

* NAME: Gives the element a name. Set to alphanumeric character.
* VALUE: Sets the default content of the element.

<INPUT TYPE=SUBMIT>:

Creates a submit button that the user can click to send data in the form back to the web server.

ATTRIBUTES:

NAME: Gives the element a name. Set to alphanumeric characters.

VALUE: Gives this button another label besides the default, Submit Query. Set to alphanumeric characters.

**<**INPUT TYPE=TEXT>:

Creates a text field that the user can enter or edit text in.

ATTRIBUTES:

NAME: Gives the element a name. Set to alphanumeric characters.

VALUE: Holds the initial text in the text field. Set to alphanumeric characters.

**6.6 JAVA SCRIPT**

Java script originally supported by Netscape navigator is the most popular web scripting language today. Java script lets you embedded programs right in your web pages and run these programs using the web browser. You place these programs in a <SCRIPT> element, usually with in the <HEAD> element. If you want the script to write directly to the web page, place it in the <BODY> element.

**JAVASCRIPT METHODS:**

**Writeln:**

Document.writeln() is a method, which is used to write some text to the current web page.

**onClick:**

Occurs when an element is clicked.

**onLoad:**

Occurs when the page loads.

**onMouseDown:**

Occurs when a mouse button goes down.

**onMouseMove:**

Occurs when the mouse moves.

**onUnload:**

Occurs when a page is unloaded.

**7. Code**

**7.1 Login Code**

<%@page import="java.sql.\*,java.lang.\*,dbconnection.\*"%>

<%

String uname=request.getParameter("uname");

String pass=request.getParameter("pass");

try{

Connection con=DBConnection.dBCon();

Statement st=con.createStatement();

ResultSet rs=st.executeQuery("select \* from instructor where uname='"+uname+"' and pass='"+pass+"'");

if(rs.next())

{

session.setAttribute("iname", uname);

session.setAttribute("pass", pass);

response.sendRedirect("instructorhome.jsp");

}else

{

response.sendRedirect("ilogin.jsp?message=Invalid Login Details");

}

}catch(Exception e){System.out.println(e);}

%>

**7.2 Registration Code:**

<%@ include file="mainhead.jsp" %>

<div style="position:absolute; top:200px; left:600px">

<!DOCTYPE html>

<html>

<body>

<script>

function myFunction() {

window.open("register.jsp","\_self");

}

function myFunction1() {

window.open("reginst.jsp","\_self");

}

</script>

<script>

function validation()

{

var a=document.s.sname.value;

if(a=="")

{

alert("Enter Student Name");

document.s.sname.focus();

return false;

}

var b=document.s.sid.value;

if(b=="")

{

alert("Enter Student ID");

document.s.sid.focus();

return false;

}

var c=document.s.email.value;

if(c=="")

{

alert("Enter Student Email");

document.s.email.focus();

return false;

}

var d=document.s.dob.value;

if(d=="")

{

alert("Enter Student Date Of Birth");

document.s.dob.focus();

return false;

}

var e=document.s.uname.value;

if(e=="")

{

alert("Enter Student Full Name");

document.s.uname.focus();

return false;

}

var f=document.s.pass.value;

if(f=="")

{

alert("Enter Student Password");

document.s.pass.focus();

return false;

}

var g=document.s.pass2.value;

if(g=="")

{

alert("Enter Student Conform Passwprd");

document.s.pass2.focus();

return false;

}

var h=document.s.pic.value;

if(h=="")

{

alert("Select Profile Picture");

document.s.pic.focus();

return false;

}

var i=document.s.pass.value;

var j=document.s.pass2.value;

if(i!=j)

{

alert("Password And Conform Password Are Not Same");

document.s.pass.focus();

return false;

}

}

</script>

<input type="checkbox" checked = "checked" onclick="myFunction()">Student

<input type="checkbox" onclick="myFunction1()">Instructor

<br><br>

<form name="s" action="register1.jsp" onsubmit="return validation();" enctype="multipart/form-data" method=post>

Name<input type=text name="sname"><br><br>

ID<input type=text name="sid"><br><br>

Department<select name=dept><br><br>

<option>ECE</option>

<option>CSE</option>

<option>EEE</option>

<option>IT</option>

<option>Mech</option>

<option>Civil</option>

</select><br><br>

Email <input type=text name="email"><br><br>

DOB <input type=date name="dob"><br><br>

Full Name <input type=text name="uname"><br><br>

Password <input type=password name="pass"><br><br>

Conform Password <input type=password name="pass2"><br><br>

Select Pic<input type=file name=pic><br><br>

<input type=submit value="Register"></table>

</form>

</body>

</html>

</div>

**7.3 Quiz Code:**

<%@ include file="shead.jsp" %>

<div style="position:absolute; top:250px; left:600px">

<script>

function myFunction() {

var a=document.getElementById("anjum").value;

window.open("squiz2.jsp?sub="+a,"\_self");

}

</script>

<%@page import="java.sql.\*,java.lang.\*,dbconnection.\*"%>

<%

try{

Connection con=DBConnection.dBCon();

Statement st=con.createStatement();

ResultSet rs=st.executeQuery("select \* from sub");%>

Select Subject <select name=sub onChange="myFunction()" id="anjum">

<option>select</option>

<% while(rs.next())

{

%>

<option value="<%=rs.getString(2)%>"><%=rs.getString(2)%></option>

<%}%>

</select><br><br>

<%}catch(Exception e){System.out.println(e);}

%>

**7.4 Result Code:**

<%@ include file="shead.jsp" %>

<div style="position:absolute; top:250px; left:600px">

<%@page import="java.sql.\*,java.lang.\*,dbconnection.\*"%>

<%Statement st=null; %>

<%

int qid2=Integer.parseInt(request.getParameter("qid2"));

int qid=Integer.parseInt(request.getParameter("qid"));

int que=Integer.parseInt(request.getParameter("que"));

int eid2=Integer.parseInt(request.getParameter("eid2"));

int eid=Integer.parseInt(request.getParameter("eid"));

String sid=(String)session.getAttribute("sid");

int qnum=0,sc=0;

String a=null,b=null,url=null,status=null;

try{

Connection con=DBConnection.dBCon();

st=con.createStatement();

ResultSet rs=st.executeQuery("select count(qnum) from exam where sid='"+sid+"' and eid2='"+eid2+"' and opt!='null'");

while(rs.next())

{

qnum=rs.getInt(1);

}

ResultSet rs2=st.executeQuery("select \* from exam where sid='"+sid+"' and eid2='"+eid2+"'");

while(rs2.next())

{

if(rs2.getString(8).compareToIgnoreCase(rs2.getString(9))==0)

{

sc++;

}

}

url="result1.jsp?qid="+qid+"&qid2="+qid2+"&que="+que+"&eid="+eid+"&sid="+sid+"&eid2="+eid2;

%>

<h2>Total Number of Questions : <%=que%></h2>

<h2>Number of Questions Answered : <%=qnum%></h2>

<h2>Score : <%=sc%></h2>

<%

ResultSet rs3=st.executeQuery("select \* from exam where sid='"+sid+"' and eid2='"+eid2+"'");

System.out.println("select \* from exam where sid='"+sid+"' and eid2='"+eid2+"'");

%>

<table border=1><tr><th>Question Number</th><th>Question</th><th>Answer</th><th>Your Answer</th><th>Status</th></tr>

<%while(rs3.next())

{

if(rs3.getString(8).compareToIgnoreCase(rs3.getString(9))==0)

{

status="Correct";

}

else

{

status="Wrong";

}%>

<tr><td><%=rs3.getInt(6)%></td><td><%=rs3.getString(7)%></td><td><%=rs3.getString(8)%></td><td><%=rs3.getString(9)%></td><td><%if(status=="Correct"){%><font color=green><% }else{%><font color=red><%}%><%=status%></font></td></tr>

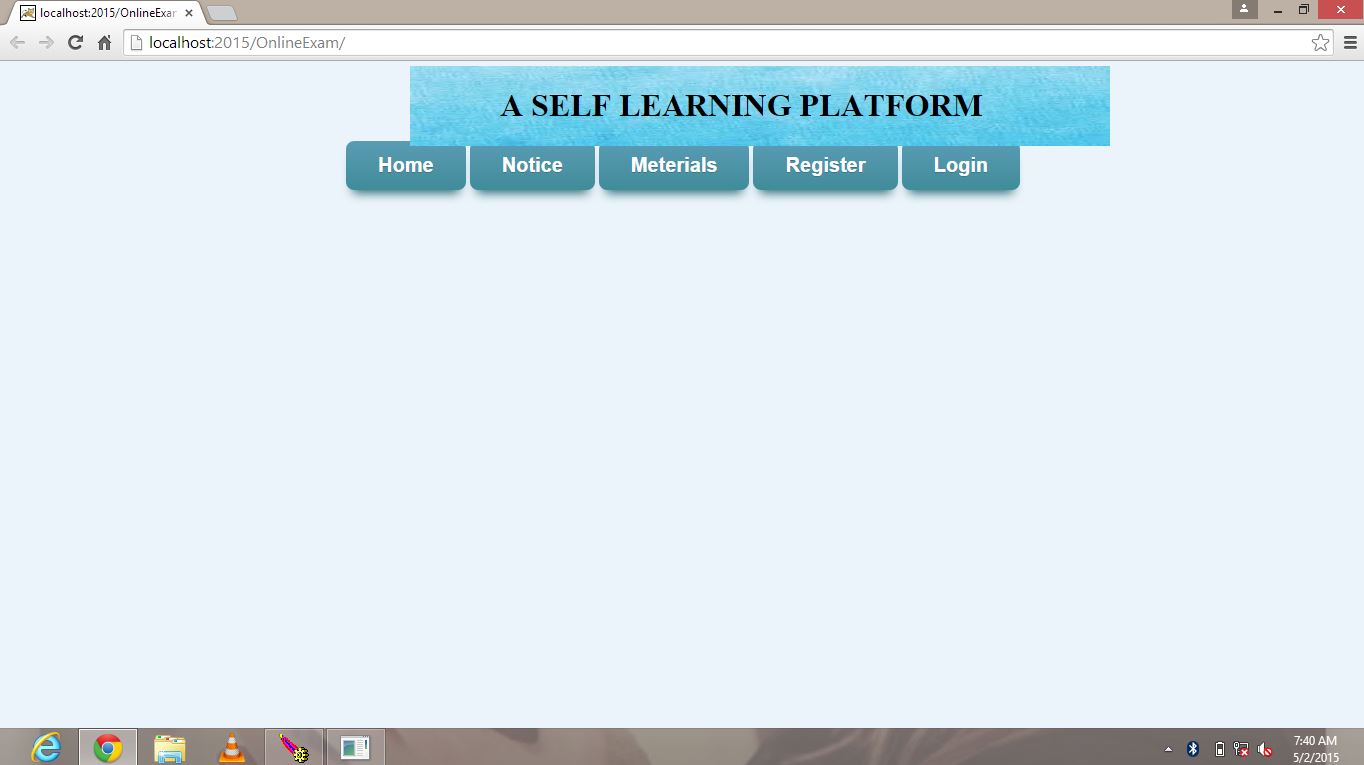
<% }%>

</table>

<%}catch(Exception e){System.out.println(e);}

%>

**8. IMPLEMENTATION AND SCREENSHOTS**

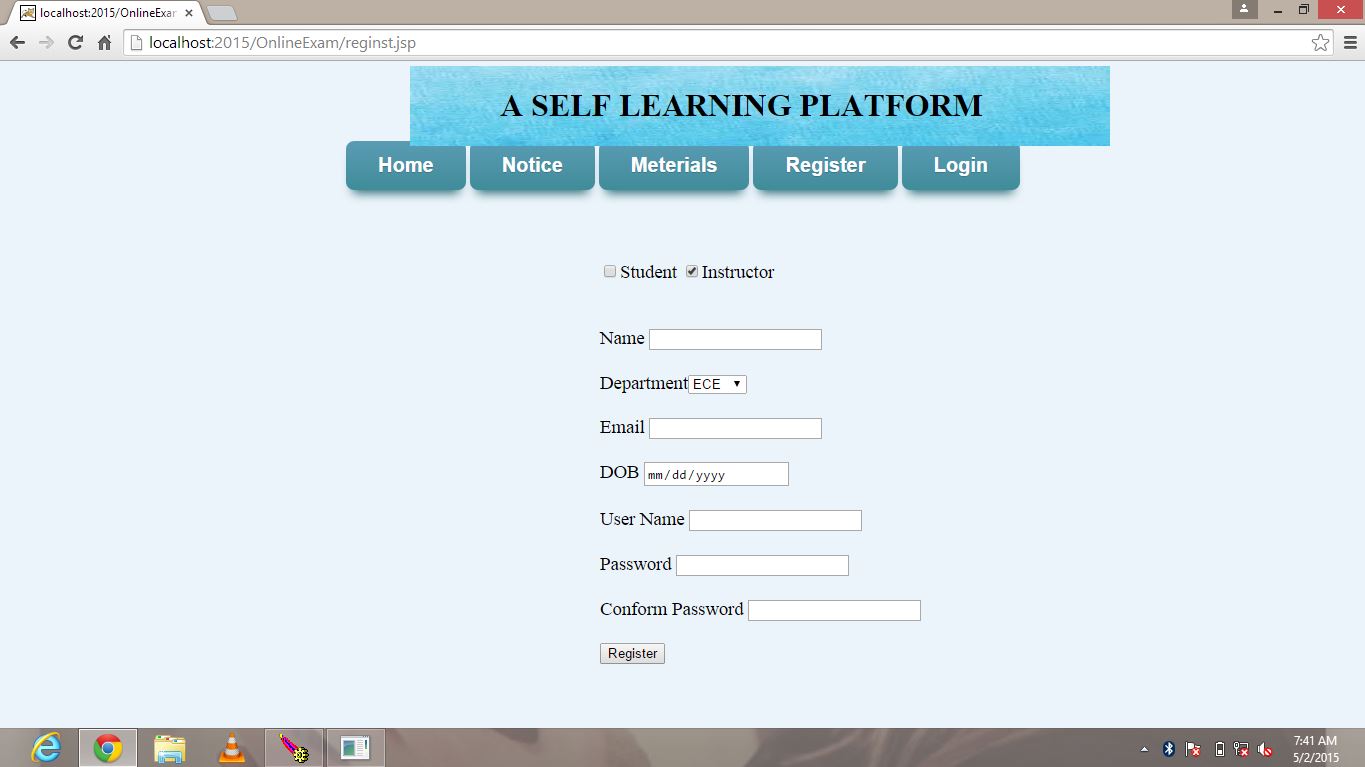
WELCOME PAGE

Steps:

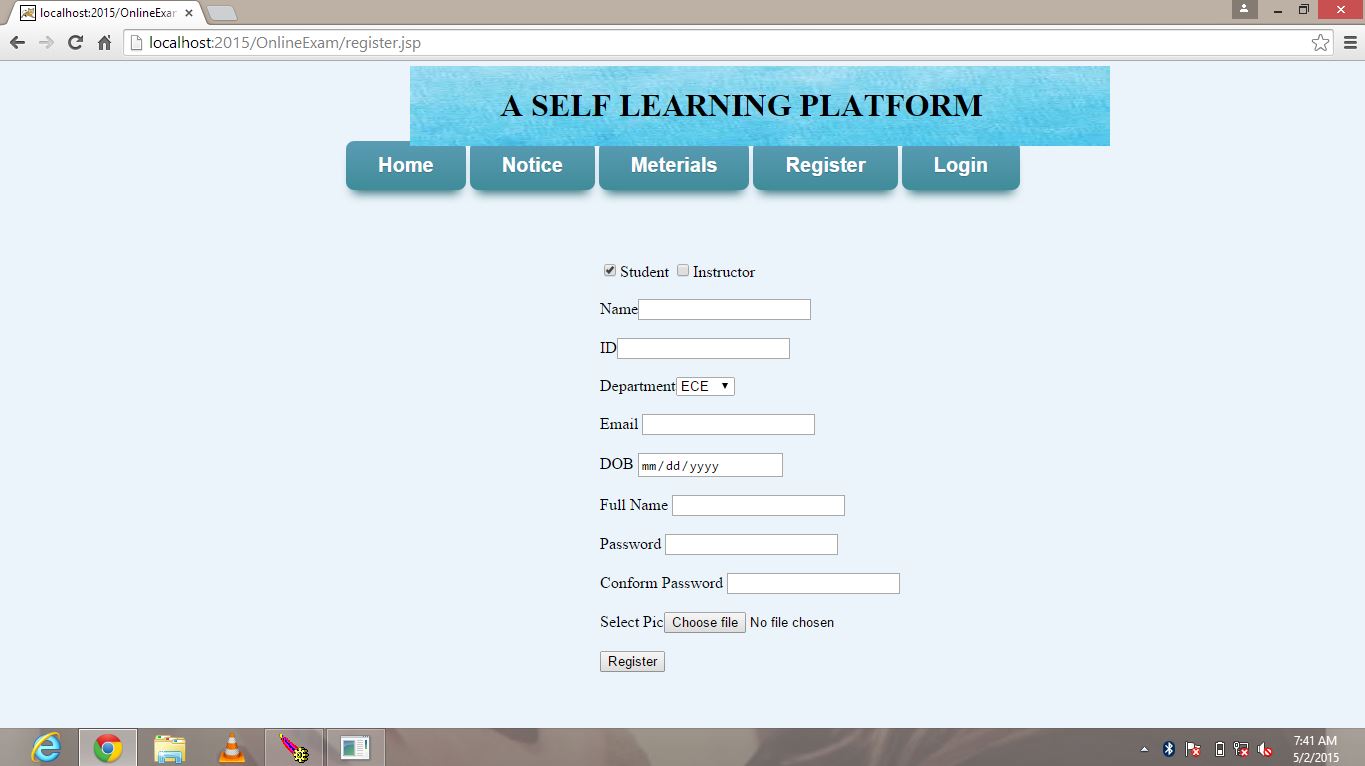
1. To login through this form you have to register first. If you are already registered please login.

2. REGISTER:

INSTRUCTOR REGISTRATION

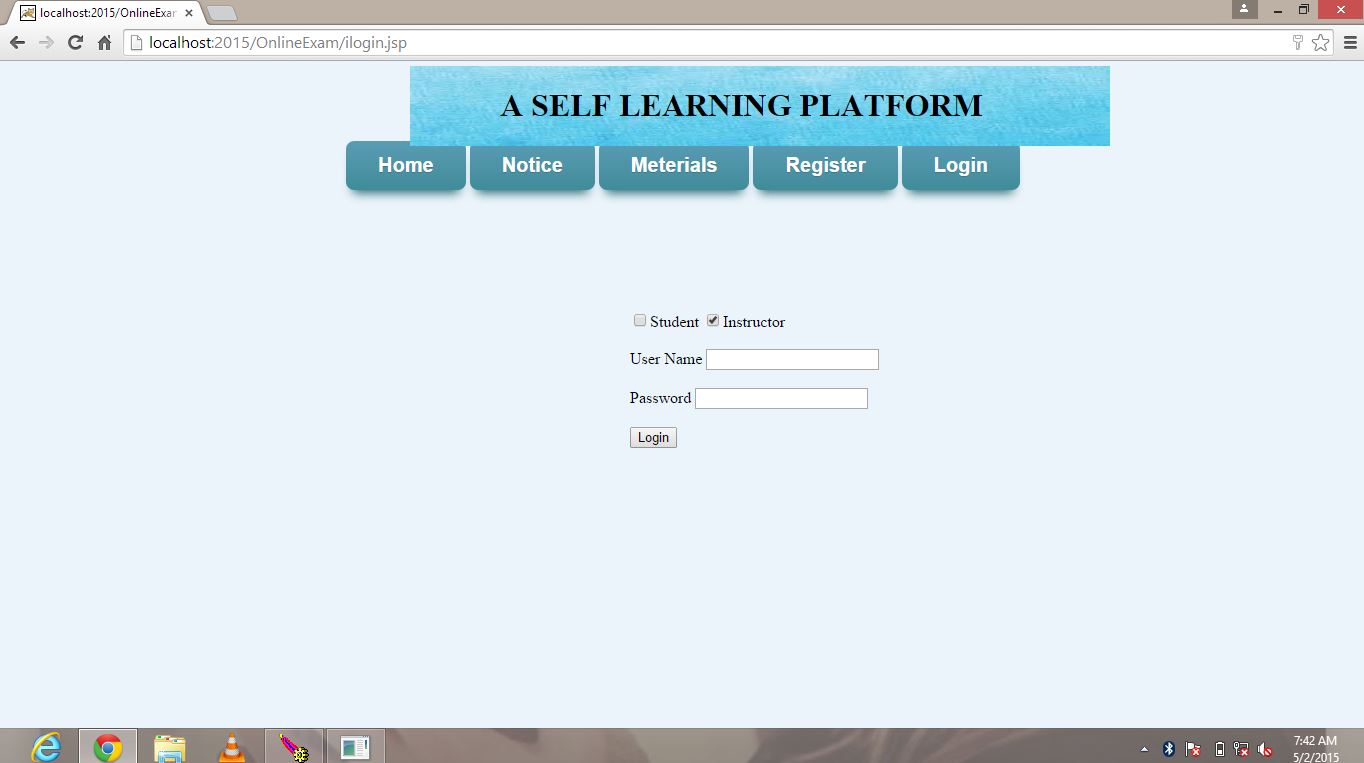


STUDENT REGISTRATION



Fill all the details and do login.

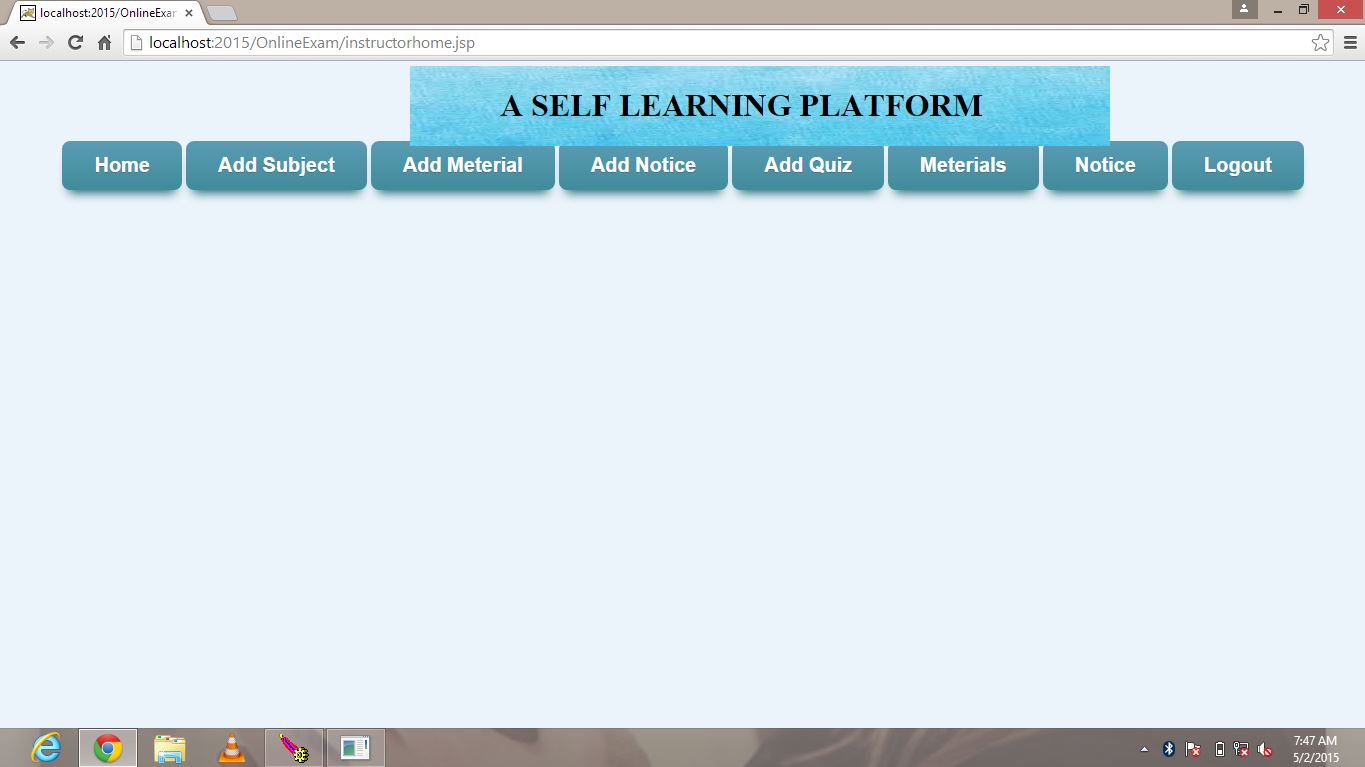
3. LOGIN PAGE



Depending upon your category you can perform action accordingly.

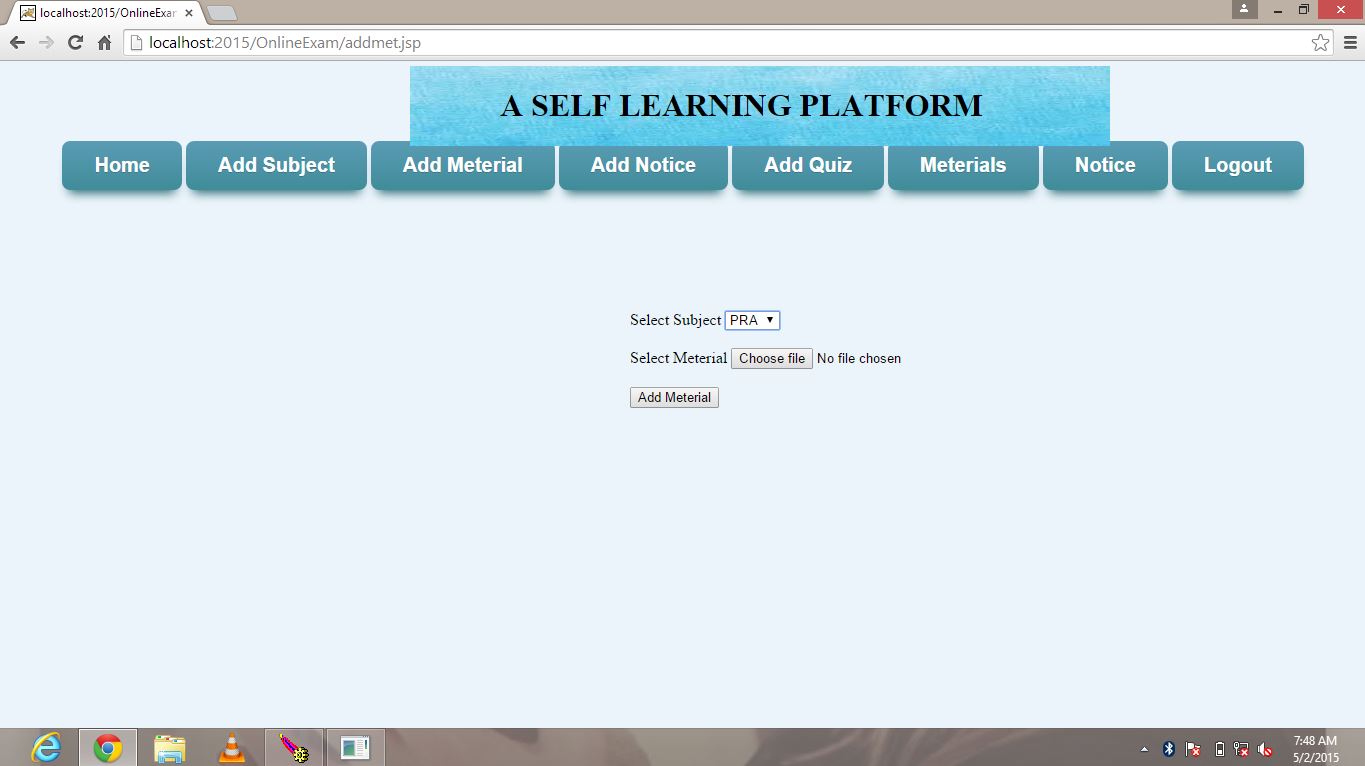
4. When category as Instructor: can add material, notices, quizzes

Instructor HOMEPAGE



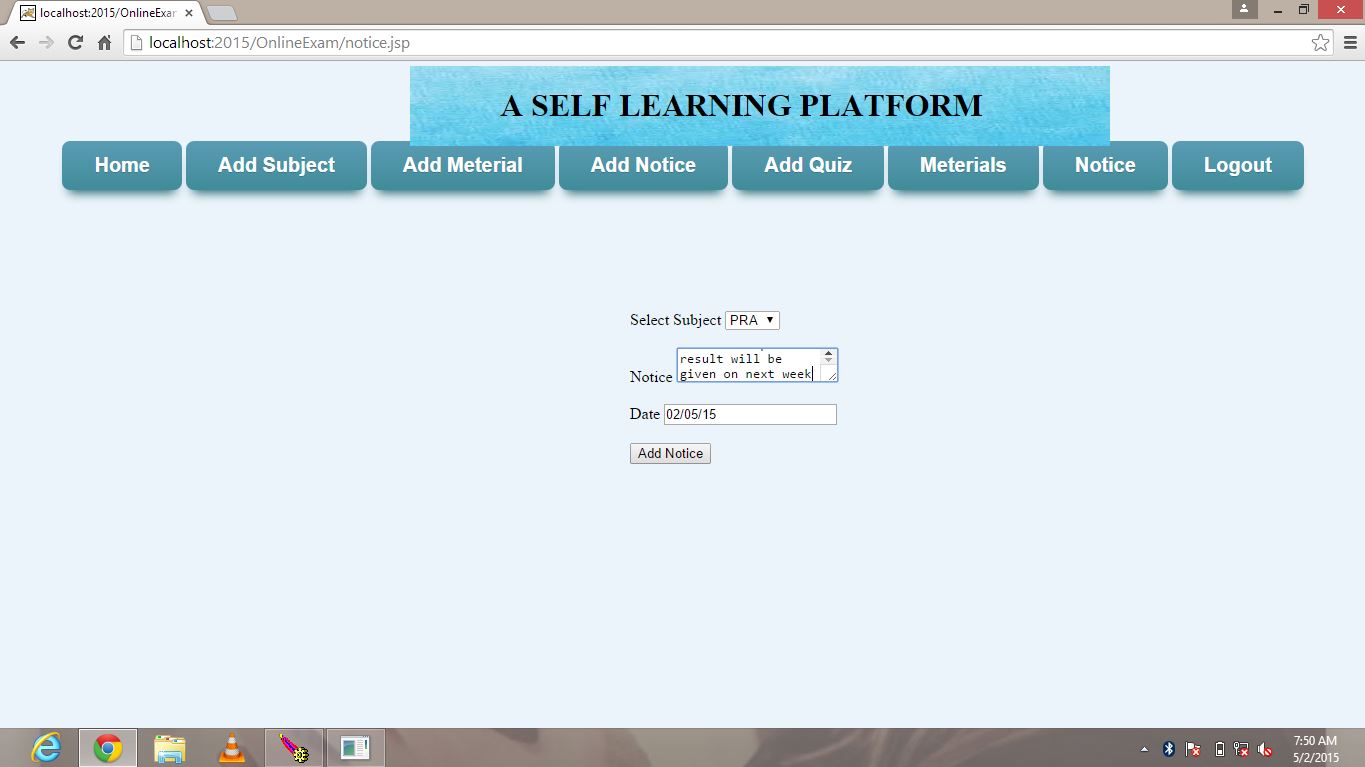
ADD MATERIAL

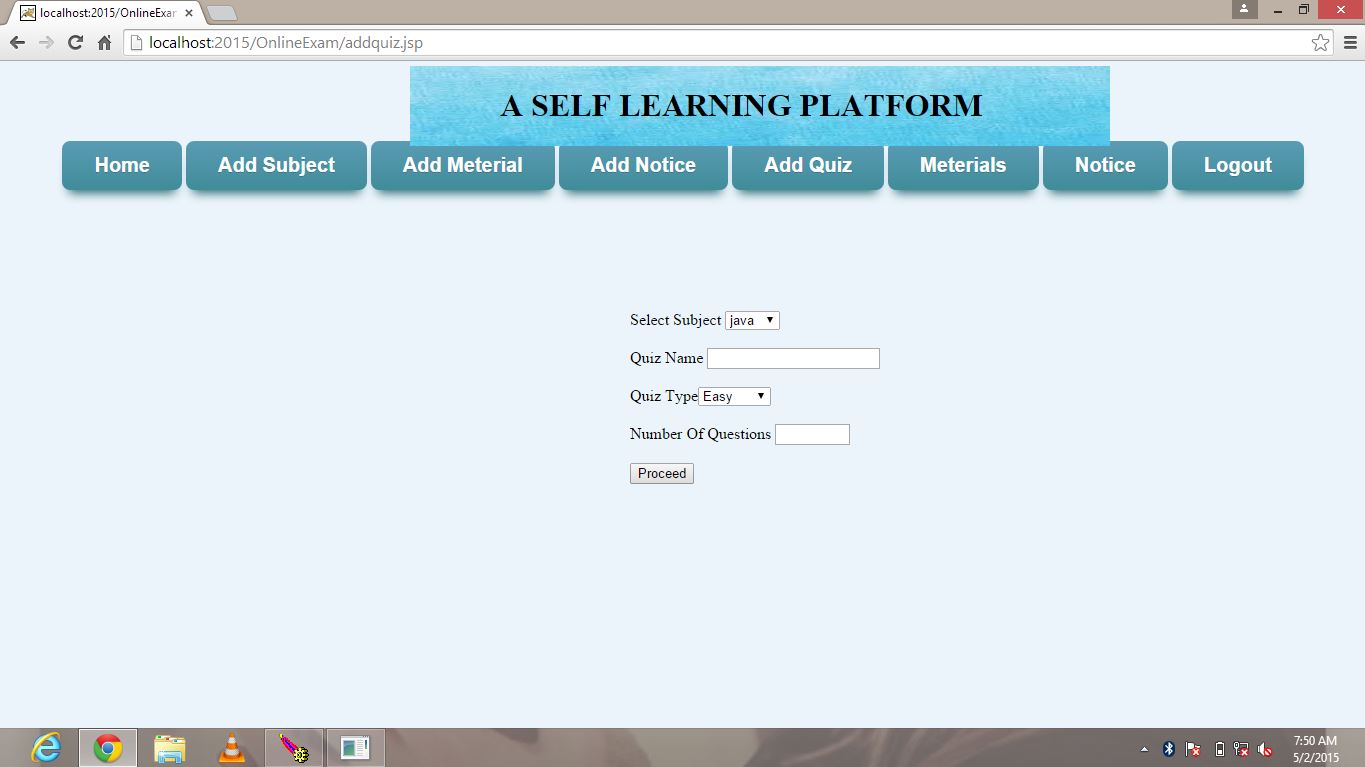
By selecting subject instructor can materials according to choice.



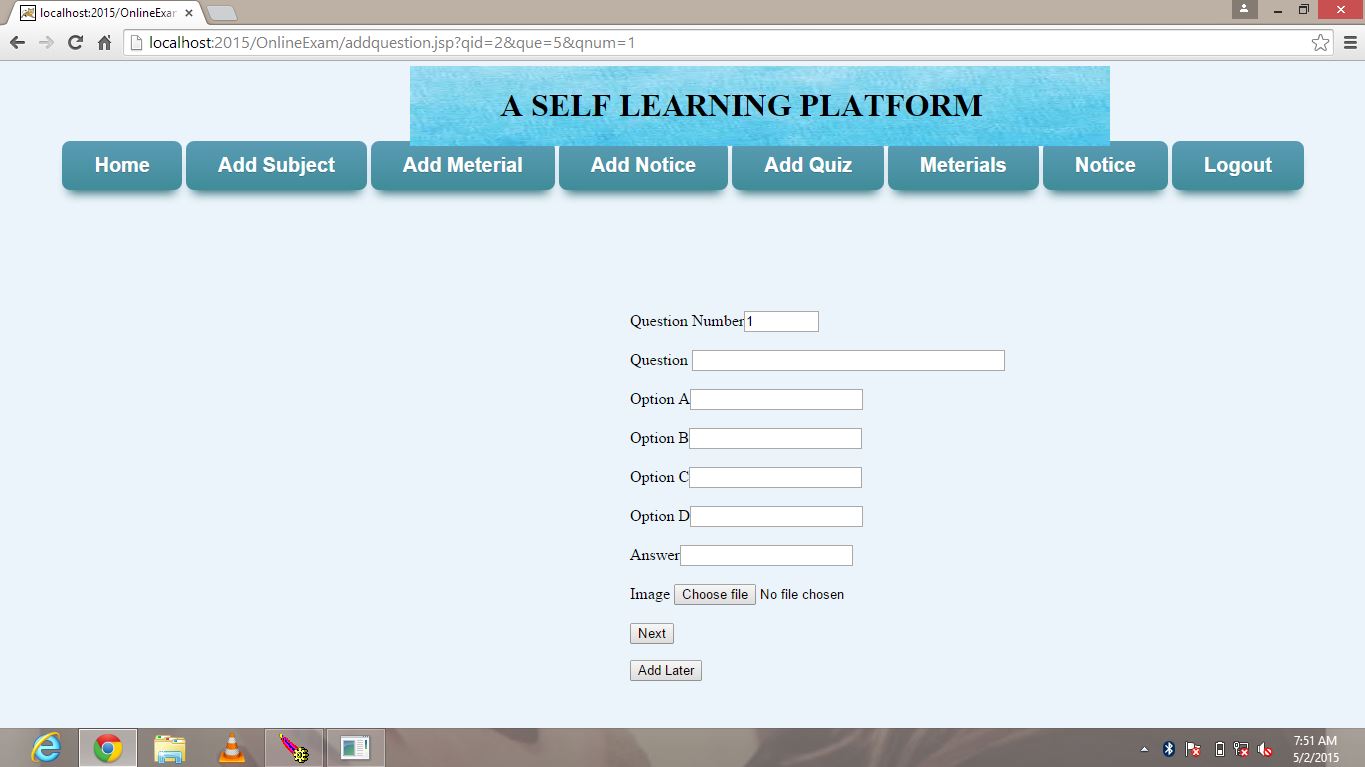
ADD NOTICE

Instructor can add notices if they want to display any information



ADD QUIZ: Instructor has to make the quiz according to the choice of subject given. He has to choose the following options mode of quiz, subject, and number of questions

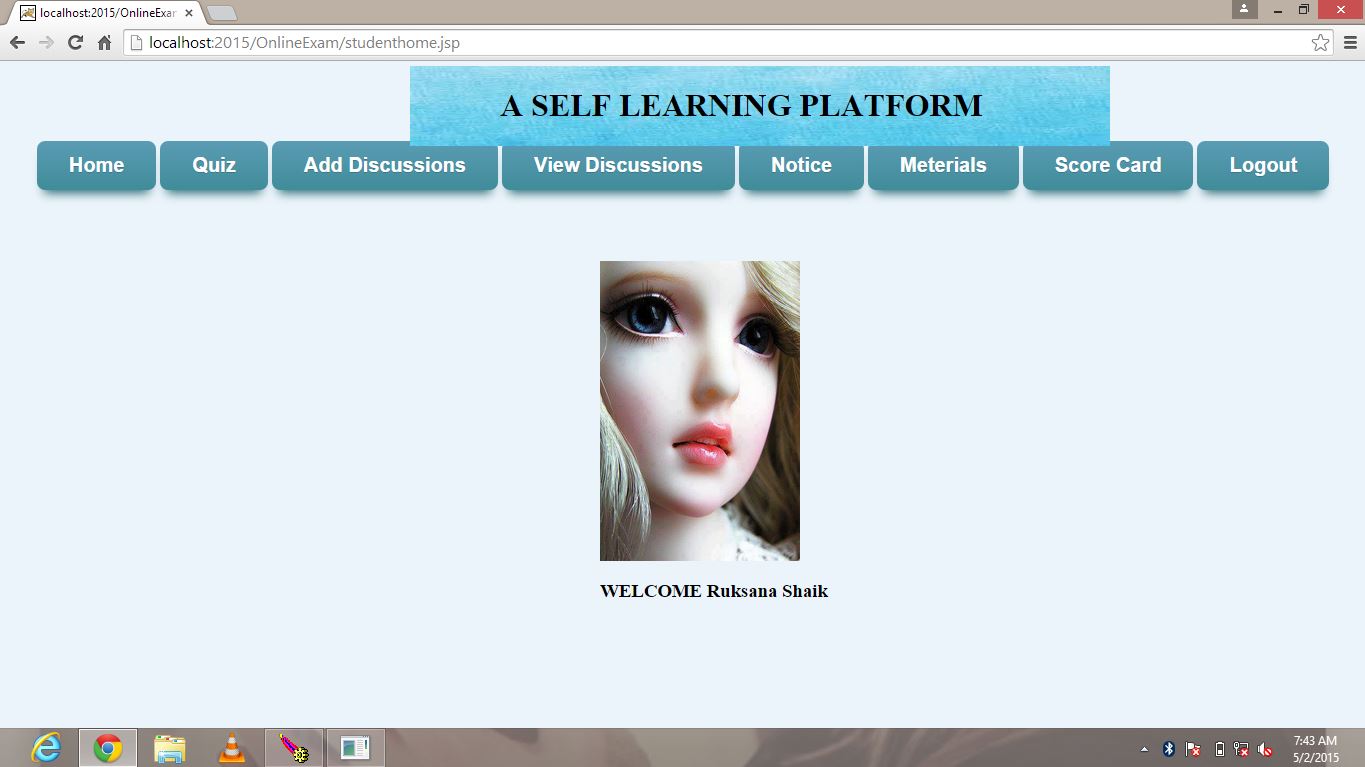
After the choosing the subject, quiz mode he need to add the questions to that quiz



Instructor can view the material, notices to check.

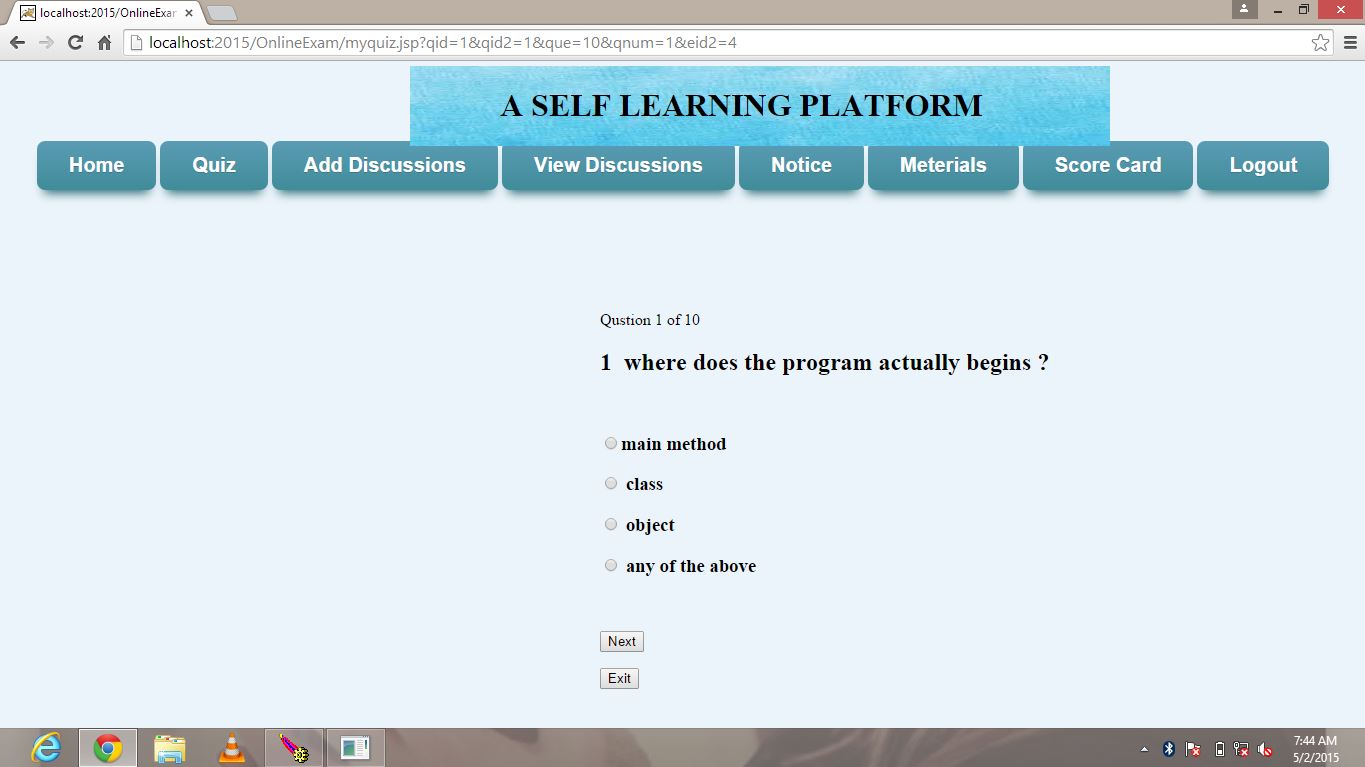
If user login as Student

Student homepage



After login student choose action to perform from the given links

If he chooses quiz by selecting subject, mode of exam questions will display

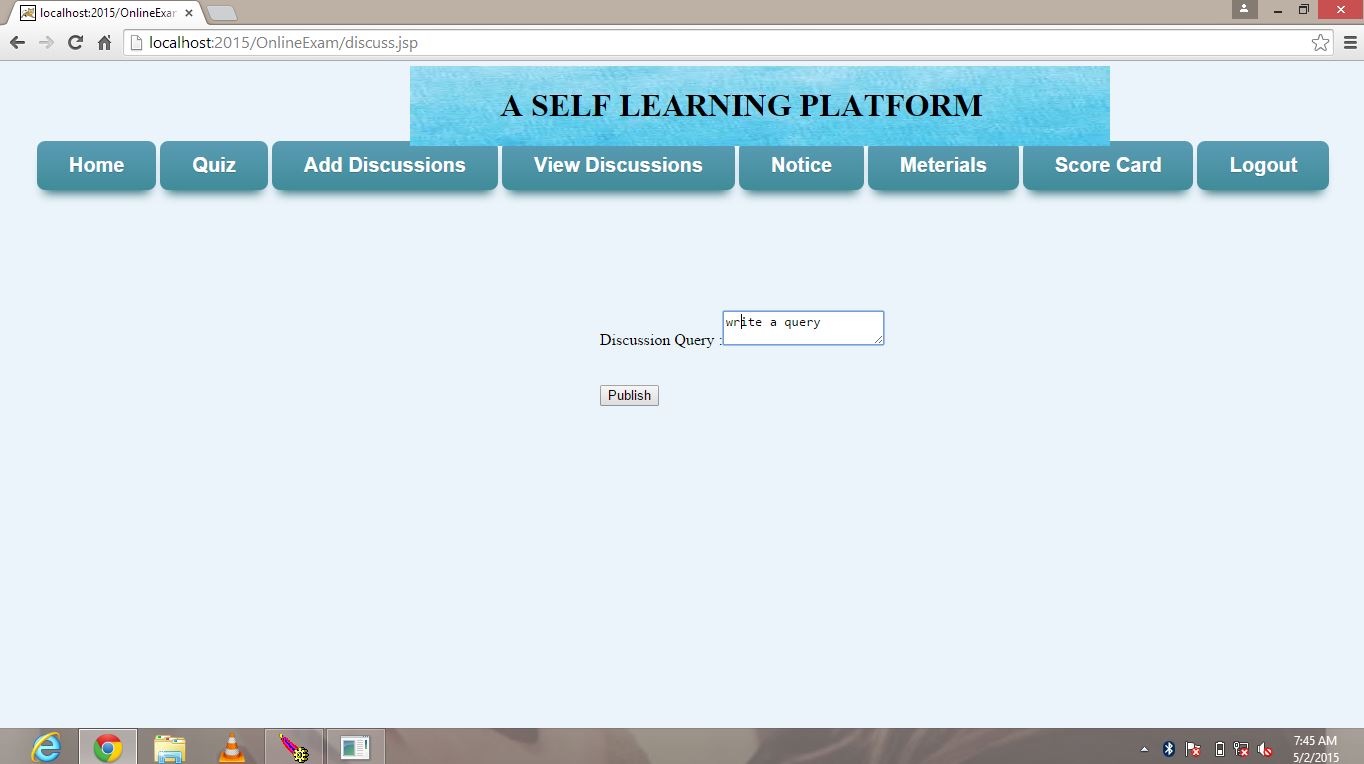


After answering all questions click on submit it will show scoreboard for the quiz finished

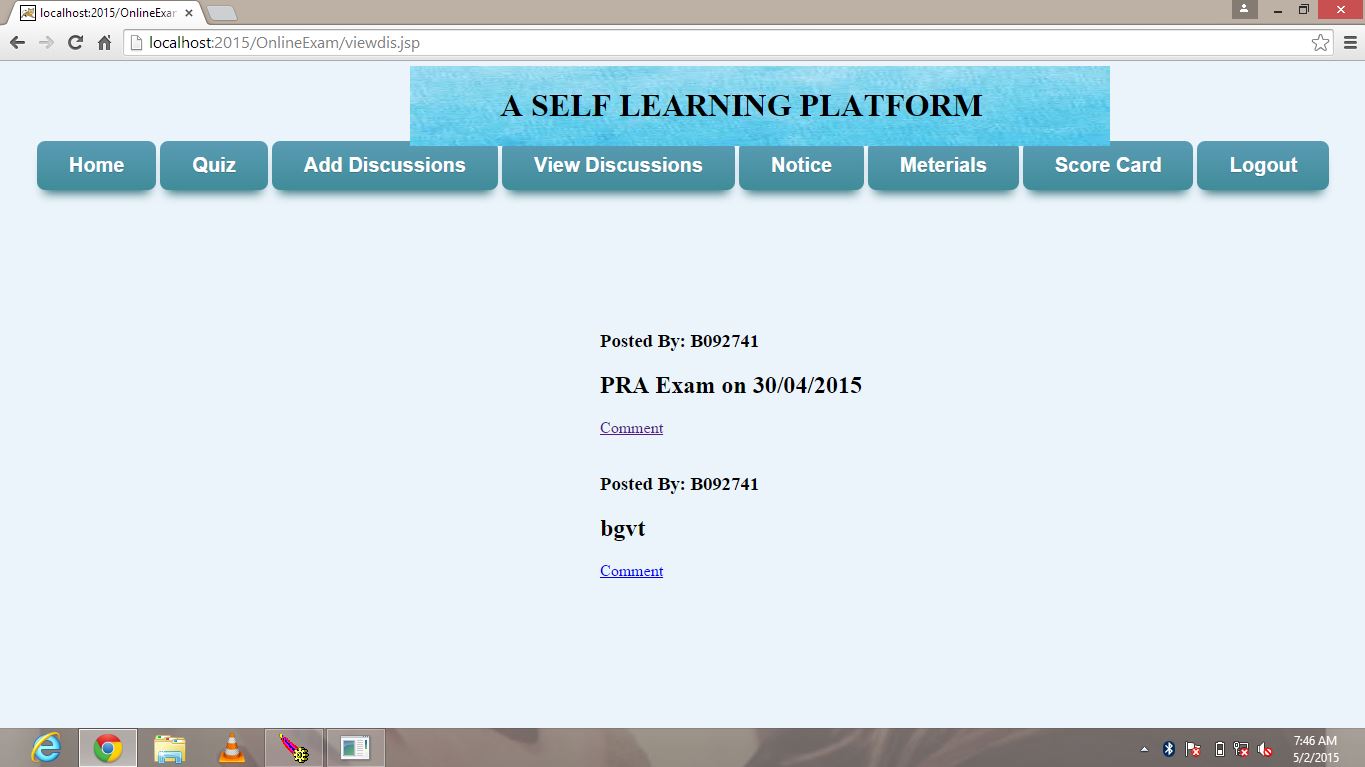


Students can post their queries get clarified by using Discussion forums

Publishing the discussion on a query



And also can comment their views for the queries



Students can also view material and can download by his choice of subject, and also can view notices posted by instructors.

**9. SOFTWARE TEST DOCUMENTATION**

**9.1 UNIT TEST DOCUMENTATION**

**1. Registration module**

**Introduction**

**System Overview:** This module requires the details of users to store in database, and also to take username and password for future login for website. In this page user need to enter the details in textboxes, if these details are correct after clicking on submit button he will be taken to welcome page.

**Test Approach:** The approach is to check valid and invalid user.

**Test Plan:**

Functionality:

After receiving username from the user it checks validation of the username entered by the user follows correct format or not. It also checks whether the user is valid or not.

Features to be tested:

* Check that the username should not be left blank.
* Whether it can detect an invalid user
* A unique password should be set

**TEST CASES**

**Case registration: TestCase1**

Purpose: Its purpose is to test how the registration page responds to the username without any entry i.e. null.

Inputs: No entry in username field.

Expected outputs: It will show an error message “Invalid username, Please enter the username”.

Test Procedure: Don’t enter any value in the username text field of registration page. Then click the register button.

**Case registration: Test Case 2**

Purpose: Its purpose is to test how the registration page responds to the password and confirm password without any entry.

Inputs: No entry in password field.

Expected Outputs It show an error message "Invalid password! Please enter any password”.

Test Procedure: Don’t enter any value in the password text field of registration page. Then click the register button.

**Case registration: Test Case 3**

Purpose: Its purpose is to test how the registration module responds to the email without any entry.

Inputs: No entry in email field.

Expected outputs: It shows an error message “Invalid email! Please enter your email id”.

Test Procedure: Don’t enter any value in the email text field of registration page. Then click the register button.

**Case registration: Test Case 4**

Purpose: Its purpose is to test how the register module responds to the Date of Birth without any entry.

Inputs: No entry in DOB field.

Expected outputs: It shows an error message “Invalid DOB! Please enter your Date of birth”.

Test Procedure: Don’t enter any value in the DOB text field of registration page. Then click the register button.

**Case registration: Test Case 5**

Purpose: Its purpose is to test how the registration module responds to the department without any entry.

Inputs: No entry in department field.

Expected Outputs It show an error message "Invalid! Please select department”.

Test Procedure: Don’t enter any value in the department text field of register page. Then click the register button.

**2. Login module**

**System Overview:** This module requires the details of users to check from database whether the login details exist or not. In this page user need to enter the username or login id and password entered at the time of registration. After login page user can access links.

**Test Approach:** The approach is to check valid and invalid user.

**Test Plan:**

Functionality:

After receiving username from the user it checks validation of the username entered by the user follows correct format or not. It also checks whether the user is exist or not.

Features to be tested:

* Check that the username should not be left blank whether it can detect an invalid user
* Check username and password are valid and exist in database

**TEST CASES**

**Case Login: TestCase1**

Purpose: Its purpose is to test how the login page responds to the username without any entry i.e. null.

Inputs: No entry in username field.

Expected outputs: It will show an error message “Invalid username, Please enter the username”.

Test Procedure: Don’t enter any value in the username text field of login page. Then click the login button.

**Case login: Test Case 2**

Purpose: Its purpose is to test how the login page responds to the password and confirm password without any entry.

Inputs: No entry in password field.

Expected Outputs: It show an error message "Invalid password! Please enter any password”.

Test Procedure: Don’t enter any value in the password text field of login page. Then click the login button.

**3. INSTRUCTOR MODULE**

System Overview: This module requires login from the instructor username and password. Instructor can perform add subject, add quiz, add material, add notice tasks.

**Case Instructor: Test Case1**

Purpose: Case1 checks the add subject page whether subject entered by the instructor valid or not.

Inputs: No entry in subject field

Expected outputs: It shows an error message “enter the subject name”.

Test procedure: Do not enter or select any subject name of subject field of add subject page. Then click on add button.

**Case instructor: Test case 2**

Purpose: It will check the material uploaded by instructor correct or not.

Inputs: Do not upload any file in material field

Expected Outputs: It shows error

Test procedure: Do not select any material to upload in material field then click on submit.

**Case instructor: Test Case 3**

Purpose: It will check the errors in add notice page.

Inputs: Instructor has to choose the subject and enter notice information.

Expected output: if the fields are empty it will error message.

Test Procedure: do not select any subject or do not enter any text in notice field then click on add notice button.

**Case instructor: Test Case 4**

Purpose: it will check the errors in quiz page.

Inputs: select subject, select mode, number of questions

Expected Output: if the mandatory fields are empty it will show error message

Test procedure: leave any text field as empty then click on submit quiz.

**4. STUDENT MODULE**

System Overview: This module requires login from the student username and password. Student can access quiz, material, view notice pages.

**Case Student: Test Case1**

Purpose: It checks the quiz page whether subject entered by student valid or not.

Inputs: No entry in subject field

Expected outputs: It shows an error message “enter the subject name”.

Test procedure: Do not enter or select any subject of subject field. Then click on next button.

**Case Student: Test Case2**

Purpose: It checks the quiz page whether mode of quiz entered by student valid or not.

Inputs: No entry in quiz mode (easy, medium, hard) field

Expected outputs: It shows an error message.

Test procedure: Do not enter or select any choice. Then click on next button.

**9.2. INTEGRATION TEST DOCUMENTATION**

**INTRODUCTION**

**System Overview:**

Self learning platform is a website where students and teachers can do their tasks independently. Student or instructor has to login to the site with valid username and password given at registration then only they can do their tasks. After login to the site instructor can add the material through add material page, can put the notices from add notice page. Quizzes should be added by instructor. Students can view the material and notices and they can discuss their queries from discussion page. Student can assess himself by taking quizzes. After finishing their task they can logout from website.

**Test Approach**

Here we are integrating all the modules registration, login, instructor module, student module to check whether these pages working or not. Our database contains some registration details of students and instructors. We have to check whether all details are correct; if correct will check whether they are working properly or not.

**Test Plan**

Features to be tested

* Correctness of username and password for login
* Check the details are entered without error in registration form
* Check the working with multiple users at a time by accessing form different computers
* Check whether the detailed entered by users are stored in database

**TEST CASES**

**CASE 1: Correctness of username and password for login**

Purpose: Check that the user name entered by the user is in correct format. Username and password should be unique.

Inputs: enter the registered username and password

Chandana 123 (registered)

Abcds 123(not registered)

Expected outputs: for first details it will successfully login to homepage, second input will show error asking valid details

Test procedure: Open login page and enter inputs then click on login button

**CASE 2: Check the details are entered without error in registration form**

Purpose: Check all fields of registration page not empty and details given are unique and valid.

Inputs: enter both valid and not valid inputs, and leave field empty

Expected outputs: for valid details it will successfully login to homepage, if fields are empty will show error asking valid details

Test procedure: Open registration page and enter inputs then click on register button

**CASE 3: Check the working with multiple users at a time by accessing form different computers**

Purpose: Check website can access for multiple users without any errors.

Inputs: connect to server and access from multiple computers

Expected outputs: access should be allowed and will not show errors

Test procedure: arrange more than three systems and access from one server

**CASE 4: Check whether the detailed entered by users are stored in database**

Purpose: Check the details in database whether all details entered by users are stored

Inputs: enter registration details

Expected outputs: if the details are wrong it will not store any details

Test procedure: enter some valid and invalid details and check in database

**10.CONCLUSION**

The “**Self learning platform**” is accessed through Internet. The user connects to Internet by means of a login and password. After those members are going to use this application, discuss and quiz part. In this system the users want to use this system should be registered then he will be allowed to use the system. Every user that they want to use this system must login with their username and password.

**APPENDIX**

**ABBREVATIONS:**

HTML: Hyper text mark up language.

JSCRIPT: java script

# DFD: Data Flow Diagrams

HTTP: Hyper Text Transfer Protocol

JDBC: Java Data Base Connectivity**.**

**FAQ’S**

**What is java script?**

**Java script is a compact, object based scripting language for developing client and server internet applications.**

**Client vs server side java script?**

Client side java script is interpreted only with in the browser that support it, and the code is visible to the user. Server side java script is stored in a pre-compiled state on the server, so it is browser – independent, and only the results of the java script programs are passed to the browser, so that code is never revealed.

**Where can <script> container tags be placed with in an html document?**

In general, the <script> container tags may appear any where with in the html document. It is more viable to have the tags placed with in the <head> container.

**BIBLIOGRAPHY**

**Advanced Java Programming - Dietel and Dietel**

**Mastering JAVA 2 - John Zukowski**

**Java Server Programming - Apress**

**Software Engineering - Roger S Pressman**

**Análysis & Design of InformationSystems – Senn**

Websites

[www.eci.gov.in](http://www.eci.gov.in)

[www.google.com](http://www.google.com)

[www.apeci.com](http://www.apeci.com)

[www.askjeeves.com](http://www.askjeeves.com)