**e-EXAM**

*A Industry Oriented Mini Project report submitted*

*in partial fulfillment of requirements*

*for the award of degree of*

**Bachelor of Technology**

**In**

**Information Technology**

By

**Kanaparthi Harshini (Reg No: 12131A1244)**

**Maddali Susmitha (Reg No: 12131A1260)**

**Koppusetti Hareesh (Reg No: 12131A1256)**

**Kakarlamudi Prudhvi Prasanth (Reg No: 12131A1243)**

****

Under the esteemed guidance of

**Mr. S. Kanthi Kiran**

**Associate Professor**

Department of Information Technology

**Gayatri Vidya Parishad College Of ENGINEERING (AUTONOMOUS)**

(Affiliated to JNTU-K, Kakinada)

**Visakhapatnam**

**2014 - 2015**

Gayatri VidyaParishad College of Engineering (Autonomous) Visakhapatnam



CERTIFICATE

This report on **“e-Exam*”*** is a bonafide record

of the mini project work submitted

**By**

**Kanaparthi Harshini (Reg No: 12131A1244)**

**Maddali Susmitha (Reg No: 12131A1260)**

**Koppusetti Hareesh (Reg No: 12131A1256)**

Kakarlamudi Prudhvi Prasanth (Reg No: 12131A1243)

in their VII semester in partial fulfillment of the requirements for the Award of Degree of

Bachelor of Technology

In

Information Technology

During the academic year 2015

S. Kanthi Kiran Dr. K. B. Madhuri

Name of the Guide Head of the Department

External Examiner

**ACKNOWLEDGEMENT**

We would like to express our deep sense of gratitude to our esteemed institute “Gayatri Vidya Parishad College of Engineering (Autonomous)”, which has provided us an opportunity to fulfill our cherished desire.

We express our sincere thanks to our Principal **Dr**. **A. B. KOTESWARA RAO** for his encouragement to us during the course of this project.

We express our heartfelt thanks and acknowledge our indebtedness to **Prof. Dr. K. B. MADHURI**, Head of the Department, Department of Information Technology.

We express our profound gratitude and our deep indebtedness to our guide **Mr. S. KANTHI KIRAN**, Associate Professor. Whose valuable suggestions, guidance and comprehensive assistance helped us a lot in realizing our present project “ e-Exam”.

We also thank **Mr. SAGAR SATHULURI**, Assistant Prof. and project coordinator, for guiding us throughout the project and lead us in completing our project efficiently.

We would also like to thank all the members of teaching and non-teaching staff of the Information Technology Department for all their support in completion of our project.

**Project Members**:

Kanaparthi Harshini 12131A1244

Maddali Susmitha 12131A1260

Koppusetti Hareesh 12131A1256

Kakarlamudi Prudhvi Prasanth 12131A1243

**ABSTRACT**

Secure Login module enables the user login and other forms to be submitted securely via HTTPs, thus preventing passwords and other private user data from being transmitted in clear text. Secure Login module enforces secure authenticated session cookies, thus preventing session hijacking. This module enables the customer to feel safe and convenient to make the use of transaction in a simple way. It locks down not just the user/login page but also any page containing the user login block. This module keeps the data safer and confidential.

INDEX

1. Introduction
   1. Functionality

2.1 Existing System

2.2 Proposed System

3. Requirements

3.1 Software Requirements

3.2 Hardware Requirements

3.3 SRS Document

3.3.1. Objective

3.3.2 Functionality

4. Design

4.1 System Design

4.2 Database Design

4.3 UML diagrams

4.3.1 Usecase diagram

4.3.2 Class diagram

4.3.3 Interaction Diagram

4.3.4 Sequence Diagram

5. Development

6. Implementation

6.1 Code

6.2 Screenshots

7. Testing

7.1 Introduction

7.2 Types of Testing

7.3 Testcases

8. Conclusion

9. References

1. **INTRODUCTION**

Secure login is very important to the user for maintaining his/her data safer. Now a days online banking has become very popular and user friendly. So that many people are willing to use online banking rather than going to the bank and stand in the queue.But keeping data safer has become a major problem. Many methods were brought out as a solution for solving this problem. One such good method is providing extra security details. Secure Login module keeps data of the user safer and confidential. This makes user to feel comfortable to use the particular site providing the best customer service. With this secure login we have provided, an unauthenticated user cannot access the website even though he knows both the username and password of an authenticated user.

* 1. **FUNCTIONALITY:**

“Secure Login” is designed to make only the registered users can access their accounts in a safer and easier manner.

* 1. **EXISTING SYSTEM**

Currently there are large number of financial institutions providing secure logging namely SBI,ICICI etc., All of these provide username and password for the user to login to their respective institutions. If any unauthenticated user tries to access some registered user’s account by knowing his username and password, he can be easily using the money in that account. Keeping data safer has become a major problem now-a-days.

* 1. **PROPOSED SYSTEM**

In our approach we will provide more security with extra security details of the user. The tool J2SE used is a java based application.

Basically the method used involves some steps:

Initially user should provide his/her username. After providing username, user must provide his security details which were updated during his registration. And the final step to login is entering the confidential password. Then user will be redirected to the required page.

In the first step, server checks the database whether the username provided is correct or not. If it is correct, then the user will be redirected to security details page. In the second step, user will be asked to provide his security details. Then server checks the database whether the details of particular user are correct or not. If they got matched, then the user will be redirected to another step of logging. In the last step, user must provide his confidential password to login into his account. And then the user will be allowed to access his account safely.

.

To solve problems in an industry, a software engineer or a team of engineers must incorporate a development strategy that encompasses the process, methods and tools and layers. This strategy is often referred to as a process model. Process model is chosen based on the nature of the project and application, the methods and tools to be used.

The various process models are:

1. Waterfall Model
2. Prototype Model
3. RAD Model
4. Incremental Model
5. Spiral Model

**Waterfall Model:**

It is also called as classic life cycle or the linear sequential model. It suggests systematic, sequential approach to software development that begins at the system level and progresses through analysis, design, coding, testing and deployment. The initial requirements should be specified at the beginning of the project. This is the main disadvantage of this model as it is difficult for the customer to state all requirements explicitly. And Real projects rarely follow the sequential flow that the model proposes and working version of the program will not be available until late in the project time span.

**Prototype Mode:**

Prototyping is a process that enables the developer to create a model of the software that must be built. It begins with requirements gathering. Developer and customer meet and define the overall objectives for the software, identify whatever requirements are known, and outline areas where further definition is mandatory. A “quick design” then occurs. The quick design focuses on a representation of those aspects of the software that will be visible to the user. The quick design leads to the construction of a prototype. The prototype is evaluated by the customer and is used to refine requirements for the software to be developed.

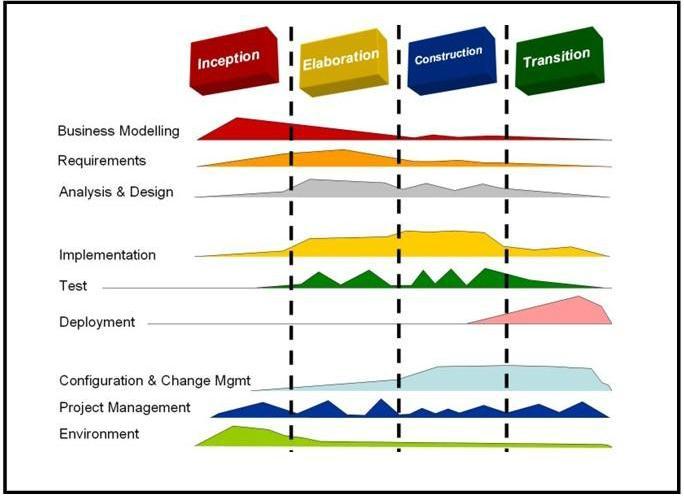
**RAD Model:**

Rapid Application Development (RAD) is an incremental software development process model that emphasizes an extremely short development cycle. The RAD model is a high-speed adaptation of the linear sequential model in which rapid development is achieved by using component based construction. If requirements are well understood and project scope is constrained, the RAD process enables a development team to create a “fully functional system” within very short time periods. RAD approach encompasses the following phases-Business modeling, Data modeling, Process modeling, Application generation, Testing and turn over.

**Incremental Model:**

The incremental model combines elements of the linear sequential model with the iterative philosophy of prototyping. When an incremental model is used, the first increment is often a core product. That is, basic requirements are addressed, but many supplementary features (some known, others unknown) remain undelivered.

The core product is used by the customer. As a result of use and/or valuation, a plan is developed for the next increment. The plan addresses the modification of the core product to be better meeting the needs of the customer and the delivery of additional features and functionality. This process is repeated following the delivery of each increment, until the complete product is produced.



**3. REQUIREMENTS:**

**3.1SOFTWARE REQUIREMENTS:**

* Operating System : Windows Xp, 7, 8.
* Database : ORACLE.
* Browser Capability : IE7,Google Chrome, Mozilla Firefox
* Programming Language : JAVA
* SOFTWARE : APACHE TOMCAT
* Scripting Language : JAVA SCRIPT
* Web Components : JSP

**3.2HARDWARE REQUIREMENTS:**

* PROCESSOR : core i3
* PROCESSOR SPEED : 2.4GHZ
* MONITOR : COLORMONITOR
* HARD DISK : 40GB
* RAM : 512MB

**REQUIREMENT STUDY**

The origin of most software systems is in the need of a client, who either wants to automate and existing manual system or desires a new software system. The software system itself is created by the developer finally the completed system will be used by the end user. Thus, there are three major parties interested in a new system: the client, the users, and the developer. The requirements for the system that will satisfy the need of the clients and the concerns of the user have to communicate to the developer.

The problem is that the client usually does not understand software or the software development process, and the developer often does not understand the clients problem and application area. This causes a communication gap between the parties involved in the development project. A basic purpose of software requirement specification is to bridge this communication gap. SRS is the medium through which the client and the user need are accurately specified; indeed SRS forms the basic of software development. A good SRS should satisfy all the parties-something very hard to achieve and involves trade-offs and persuasion.

**Iterative Incremental Model:**

During the first requirements analysis phase, customers and developers specify as many requirements as possible and prepare a SRS document. This model delivers an operationalquality product at each release, but one that satisfy only a subset of the customer’s requirements. The complete product is divided into releases and the developer delivers the product release by the release.

**REQUIREMENT PROCESS**:

The main reason of modeling generally focuses on the problem structure, not its external behaviors. Consequently, thing’s like user interfaces are rarely modeled, whereas thy frequently from a major components of the SRS.

Similarly performance constraints, design constraints, standards compliance, recovery, etc. are specified clearly in the SRS because the designer must know about there to properly design the system.

To properly satisfy the basic goals, an SRS should have certain properties and should contain diff type of req. A good SRS is [iEE87, IEE94]: complete if everything the software is supposed to and responses to the software to all classes of input data are specified in the SRS.

Correctness and completeness go hand in hand; an SRS in unambiguous if and only if every requirement stated has one and only one interpretation, requirements often written in natural language.

An SRS is verifiable if and only if every stated requirement is verifiable. A requirement is verifiable if there exists of some cost-effective process that can check whether the final software meets those requirements. An SRS is consistent if there is no requirements that with another.

Writing and SRS is an iterative process. Even when requirement of system are specified they are later modified as the need of the client change. Hence an SRS should be easy to modify. An SRS is traceable if the origin of each of its requirements is clear and if it facilitates the referencing of each requirement in future development [EEE87].

One of the most common problems in requirement specification is when some of the requirements of the client are not specified. This necessitates addition and modifications to the requirements later in the development cycle, which are often expensive to incorporate.

**Project Schedule Study phase:**

In the study phase we do the preliminary investigation and determine the system requirements. We study the system and collect the data to draw the dataflow diagrams. We follow the methods like questions and observation to find the facts that are involved in the process. This is an important because if the specification study is not done properly then design phase etc will go wrongly.

**Design Phase:**

In this design phase we design the system making use of study phase and the data flow diagrams. We make use the general access for designing.

We consider the top down approach. In the design phase we determine the entities and their attributes and the relationships between the entities. We do both logical and physical design of the system.

**Development Phase:**

In the development phase we mostly do the coding part following the design of the system. We follow modular programming for development and after development and after developing each and every module we do the unit testing followed by the integration testing.

**Implementation Phase:**

The last phase of the project is the implementation phase. Quality assurance is the primary motive in this phase. The quality assurance is the review of software products and related documentation for completeness, correctness, reliability and maintainability. The philosophy behind the testing is it finds errors. The testing strategies are of tow types, the code testing and the specifications testing. In the code testing we are examining the logic of the program. On the surface, code testing seems to be ideal methods for testing software, but no tall software errors are uncovered.

**Feasibility Study**

Feasibility is an important phase in software development process. It enables the developers to have an assessment of the product being developed. It refers to the feasibility study of product in terms of outcomes of the product, operational use and technical support required for implementation it.

Feasibility study should be performed on the basis of various criteria and parameters. The various feasibility studies are:

1. Economic Feasibility

2. Operational Feasibility

3. Technical Feasibility

**Economic Feasibility:**

It refers to the benefits or outcomes we are deriving from the product as compared to the total cost we are spending for developing the product.

In the present system, the development of new product greatly enhances the accuracy of the system and reduces the delay in the processing of applications and generating the reports. The errors can be greatly reduced and at the same time providing the great level of security. Here we don’t need additional equipment except memory of required capacity. No need for spending money on client for maintenance because the database used is web enabled database.

**Operational Feasibility:**

It refers to the feasibility of the product to be operational. Some products may work very well at design and implementation but may fail in the real time environment. It includes the study of human required and their technical expertise.

In the present system, the entering the details, updating the details and reports generations are perfect and quick in operations.

**Technical Feasibility:**

It refers to whether the software that is available in the market fully supports the present application .It studies the pros and cons of using particular software for the development and its feasibility. It also studies the additional time needed to be given to people to make the application work.

In the present system the user interface is user friendly and does not require much expertise and training .It just needs mouse click to do operations and to generate reports. The software that is used for developing is highly suitable for the present applications since the users require fast access to the web pages with a high degree of security. This is achieved through integration of web server and database server in the same environment.

* 1. **SRS DOCUMENT**

**3.3.1 OBJECTIVE**:

The objective of this work is to make the online banking simple and user friendly to the registered users by providing them the best service. Users can be able to use their accounts without any fear of losing their money. Users should provide their security credentials while accessing their accounts, so that the data can be kept confidential.

* + 1. **FUNCTIONALITY:**

**Technologies used are:**

**Java:**

Java is a network-friendly programming language invented by Sun Microsystems. Java is a

complete software ecosystem that represents different values to different types of users. Java technology is the first software technology that simply works without a struggle. Users are delighted to see applications run reliably and compatibly on such an incredible array of network products - from PCs, game players, and mobile phones to home appliances and automotive electronics.

Java technology is an object-oriented, platform-independent, multithreaded programming environment. It is the foundation for Web and networked services, applications, platform-independent desktops, robotics, and other embedded devices. Java is often used to build large, complex systems that involve several different computers interacting across networks, for example transaction processing systems. Java is also used to create software with graphical user interfaces such as editors, audio players, web browsers, etc.

**Java Script:**

JavaScriptis a script-based programming language which was developed by Netscape Communication Corporation. JavaScript was originally called Live Script and renamed as JavaScript to indicate its relationship with Java.

JavaScriptsupports the development of both client and server components of Web-based applications. On the client side, it can be used to write programs that are executed by a Web browser within the context of a Web page. On the server side, it can be used to write web server programs that can process information submitted by a Web browser and then updates the browser’s accordingly.

Even though JavaScriptsupports both client and server Web programming, we prefer JavaScript at Client side programming since most of the browsers supports it.

JavaScript is almost as easy to learn as HTML, and JavaScript statements can be included in HTML documents by enclosing the statements between a pair of scripting tags <SCRIPT>..</SCRIPT>.

<SCRIPT LANGUAGE=”JavaScript”>

JavaScript statements

</SCRIPT>

Here are a few things we can do with JavaScript:

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

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

**JavaScript vs. Java**

JavaScript and Java are entirely different languages. A few of the most glaring differences are:

* Java applets are generally displayed in a box within the web document; JavaScript can affect any part of the Web document itself.
* While JavaScript is best suited to simple applications and adding interactive features to Web pages; Java can be used for incredibly complex applications. There are many other differences but the important thing to remember is that JavaScript and Java are separate languages. They are both useful for different things; in fact they can be used together to combine their advantages.

**Advantages:**

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

**Java Server Pages (JSP)**

A JSP page is a text-based document that describes how to process a *request* to create a *response*. The description intermixes template data with some dynamic actions and leverages on the Java Platform.

**The Java Server Pages specification includes**:

**·** Standard directives

· Standard actions

· Script language declarations, script lets and expressions

· A portable tag extension mechanism.

**DIRECTIVES AND ACTIONS**

There may be two types of *elements* in a JSP page: ***directives*** or ***actions*.** *Directives* provide global information that is conceptually valid independent of any specific request received by the JSP page. For example, a directive can be used to indicate the scripting language to use in a JSP page. *Actions* may, and often will, depend on the details of the specific request received by the JSP page. If a JSP container uses a compiler or translator, the directives can be seen as providing information for the compilation/translation phase, while actions are information for the subsequent request processing phase.

An action may create some *objects* and may make them available to the scripting elements through some *scripting-specific variables.*

Directive elements: Syntax:

<%@ directive ...%>

Action elements: tag:<mytag attr1=”attribute value” ...>

body

</mytag>

or an empty tag

<mytab attr1=”attribute value” .../>

An element type abstracts some functionality by defining a specialized (sub)language that allows more natural expression of the tasks desired, can be read and written more easily by tools and also can even contribute specialized yet portable tool support to create them. The JSP specification provides a Tag Extension mechanism that enables the addition of new actions, thus allowing the JSP page “language” to be easily extended in a portablefashion. A typical example would be elements to support embedded database queries. Tag libraries can be used by JSP page authoring tools and can be distributed along with JSP pages to any JSP container like Web and Application servers.

The Tag Extension mechanism can be used from JSP pages written using any valid scripting language, although the mechanism itself only assumes a Java run time environment. Custom actions provide access to the attribute values and to their body; they can be nested and their bodies can include scripting elements.

**Execution:**

A *JSP page* is executed in a *JSP container*, which is installed on a Web server, or on a Web enabled application server. The JSP container delivers *requests* from a client to a JSP page and *responses* from the JSP page to the client. All JSP containers must support HTTP as a protocol for requests and responses, but a container may also support additional request/response protocols. The default request and response objects are of type HttpServletRequest and HttpServletResponse, respectively.

**Compilation:**

JSP pages may be *compiled* into its JSP page implementation class plus some deployment information. This enables the use of JSP page authoring tools and JSP tag libraries to author a Servlet.This have several benefits:

Removal of the start-up lag that occurs when a JSP page delivered as source receives the first request & Reduction of the footprint needed to run a JSP container, as the java compiler is not needed.

**Objects and Scopes:**

JSP page can create and/or access some Java objects when processing a request. The JSP specification indicates that some objects are created implicitly, perhaps as a result of a directive other objects are created explicitly through actions; objects can also be created directly using scripting code, although this is less common. The created objects have a *scope attribute* defining *where* there is a reference to the object and *when* that reference is removed

Objects are always created within some JSP page instance that is responding to some *request* object.

**There are several scopes:**

* **page :**

Objects with *page* scope are accessible only within the page where they are created. All references to such an object shall be released after the response is sent back to the client from the JSP page or the request is forwarded somewhere else. References to objects with *page* scope are stored in the page Context object.

* **Request:**

Objects with *request* scope are accessible from pages processing the same request where they were created. All references to the object shall be released after the request is processed; in particular, if the request is forwarded to a resource in the same runtime, the object is still reachable. References to objects with *request* scope are stored in the request object.

* **Session:**

Objects with *session* scope are accessible from pages processing requests that are in the same session as the one in which they were created. It is not legal to define an object with session scope from within a page that is not session-aware. All references to the object shall be released after the associated session ends. References to objects with *session* scope are stored in the session object associated with the page activation.

* **Application:**

Objects with *application* scope are accessible from pages processing requests that are in the same application as they one in which they were created. All references to the object shall be released when the runtime environment reclaims the ServletContext. Objects with application scope can be defined (and reached) from pages that are not session-aware.

**HTML**

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

**HTML** is not a programming language but it is an application of ISO Standard 8879, **SGML (Standard Generalized Markup Language),** but specialized to hypertext and adapted to the Web. The idea behind Hypertext is that instead of reading text in rigid linear structure, we can easily jump from one point to another point. We can navigate through the information based on our interest and preferences.

A markup language is simply a series of elements, each delimited with special characters that define how text or other items enclosed within the elements should be displayed. Hyperlinks are underlined or emphasized works that load to other documents or some portions of the same documents.

**HTML** can be used to display any type of document on the host computer, which can be geographically at a different location. It is a versatile language and can be used on any platform or desktop.

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

**Basic HTML Tags**

<! --- --> Specifies comments

<A>…………</A> Creates hypertext links

<B>…………. </B> Formats text as bold

<BIG>…………</BIG> Formats text in large font

<BODY>……. </BODY> Contains all tags and text in the HTML document

<CENTER>…. </CENTER> Creates text

<DD>…………..</DD> Definitions of a term

<DL>……………</DL> Creates definition list

<FONT>……. </FONT> Formats text with a particular font

<FORM>…..</FORM> Encloses a fill-out form

<FRAME>…. </FRAME> Defines a particular frame in a set of frames

<H#>………..</H#> Creates headings of different levels

<HEAD>……..</HEAD> Contains tags that specify information about a document

<HR>……..</HR> Creates a horizontal rule

<HTML>……</HTML> Contains all other HTML tags

<META>……</META> Provides meta-information about a document

<SCRIPT>……</SCRIPT> Contains client-side or server-side script

<TABLE>……</TABLE> Creates a table

<TD>……</TD> Indicates table data in a table

<TR>……</TR> Designates a table row

<TH>……</TH> Creates a heading in a table.

**ADVANTAGES**

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

**ORACLE:**

An Oracle database is a collection of data treated as a unit. The purpose of a database is to store and retrieve related information. A database server is the key to solving the problems of information management. In general, a server reliably manages a large amount of data in a multiuser environment so that many users can concurrently access the same data. All this is accomplished while delivering high performance. A database server also prevents unauthorized access and provides efficient solutions for failure recovery.

Oracle Database is the first database designed for enterprise grid computing, the most flexible and cost effective way to manage information and applications. Enterprise grid computing creates large pools of industry-standard, modular storage and servers. With this architecture, each new system can be rapidly provisioned from the pool of components. There is no need for peak workloads, because capacity can be easily added or reallocated from the resource pools as needed.

The database has logical structures and physical structures. Because the physical and logical structures are separate, the physical storage of data can be managed without affecting the access to logical storage structures.

The Oracle Database 10g family of software products implements much of the core grid technology to get companies started. And Oracle delivers this grid computing functionality in the context of holistic enterprise architecture, providing a robust security infrastructure, centralized management, intuitive, powerful development tools, and universal access. Oracle Database 10g includes:

* Oracle Database 10g
* Oracle Application Server 10g
* Oracle Enterprise Manager 10g
* Oracle Collaboration Suite 10g

Centralized Data Management. Oracle Database 10g manages all types of structured, semi-structured and unstructured information, representing, maintaining and querying each in its own optimal way while providing common access to all via SQL and XML Query. Along with traditional relational database structures, Oracle natively implements OLAP cubes, standard XML structures, geographic spatial data and unlimited sized file management, thus virtualizing information representation. Combining these information types enables connections between disparate types of information to be made as readily as new connections are made with traditional relational data.

Each Oracle database has a data dictionary. An Oracle data dictionary is a set of tables and views that are used as a read-only reference about the database. For example, a data dictionary stores information about both the logical and physical structure of the database. A data dictionary also stores the following information:

* The valid users of an Oracle database
* Information about integrity constraints defined for tables in the database
* The amount of space allocated for a schema object and how much of it is in use

A data dictionary is created when a database is created. To accurately reflect the status of the database at all times, the data dictionary is automatically updated by Oracle in response to specific actions, such as when the structure of the database is altered. The database relies on the data dictionary to record, verify, and conduct ongoing work. For example, during database operation, Oracle reads the data dictionary to verify that schema objects exist and that users have proper access to them.

**APACHE TOMCAT:**

Apache Tomcat is an open source software implementation of the Java Servlet and JavaServer Pages technologies. The Java Servlet and JavaServer Pages specifications are developed under the Java Community Process

Apache Tomcat is developed in an open and participatory environment and released under the Apache License Version 2. Apache Tomcat is intended to be a collaboration of the best-of-breed developers from around the world

 Described as a "reference implementation" of the Java Servlet and the Java Server Page specifications, Tomcat is the result of an open collaboration of developers and is available from the Apache Web site in both binary and source versions. Tomcat can be used as either a standalone product with its own internal Web server or together with other Web servers, including Apache, Netscape Enterprise Server, Microsoft Internet Information Server (IIS), and Microsoft Personal Web Server. Tomcat requires a Java Runtime Enterprise Environment that conforms to JRE 1.1 or later.

Tomcat is one of several open source collaborations that are collectively known as Jakarta.

**4. DESIGN:**

**4.1 SYSTEM DESIGN:**

System design is the process, which involves conceiving planning and carrying out the plan by generating the necessary reports and inputs. In other words design phase acts as bridge between the software requirement specification and implementation phase, which satisfies those requirements. System design is the transformation of the analysis model into a system design model.

The design of the system is correct if a system built precisely according to the requirements of that system. Design should be clearly verifiable, complete and traceable. The goal is to divide the problem into manageably small modules that can be solving separately. The different modules have to cooperate and communicate together to solve the problem. The complete project is broken down into different identifiable modules. Each module can be understood separately. All the modules at last are combined to get the solution of the complete system.

4.2 DATABASE DESIGN:

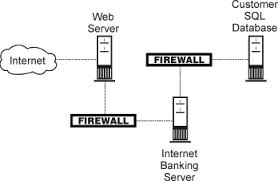
A database is an organized collection of related information used for a specific purpose, such as keeping track of ongoing work order activities or maintaining a library. When we use a computerized database management system such as oracle, the database is called ‘relational’. The information is divided into separate stacks of logically related data, each of which is stored in a separate table in database. ‘Tables’ are the fundamental objects at the heart of relational database.

Once the information is arranged in separate tables, we can view, edit, and delete the information with online forms, search for and retrieve some of or all information with queries. Rather than storing data in large two-dimensional table, called ‘flat file’, we can distribute data among individual tables.

Tables can be related in three different ways: one-to-one, one-to-many, many-to-many. The type we define depends on how many records in each table are likely to have the same value. To relate these tables, one of them must include a field that contains a unique value in every record. This is called a ‘primary key’. The most kind of relationship used is one-to-many.

Many popular DBMSs are used today. Among these popular databases are – Microsoft Access, Microsoft SQL Server, Mysql and Oracle etc.

The Block Diagram of the Architecture is as follows:



**4.3 UNIFIED MODELING LANGUAGE DIAGRAMS**

* 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.

1. This view represents the system from the user’s perspective.
2. The analysis represents 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.

**Implementation Model View**

* In the structural and behavioral as parts of the system are represented as they are to be built.

**Environmental Model View**

* In this structural and behavioral aspects of the environment in which the system is to be implemented are represented.

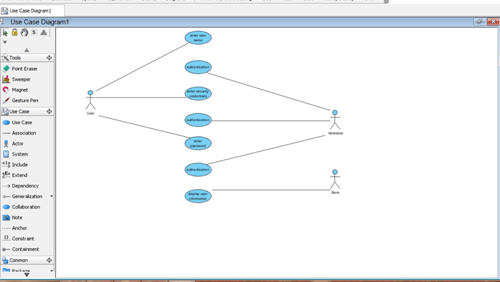
UML is specifically constructed through two different domains they are

* UML Analysis modeling, this focuses on the user model and structural model views of the system.
* UML design modeling, which focuses on the behavioral modeling, implementation modeling and environmental model views.

**4.3.1. Use case Diagrams:**

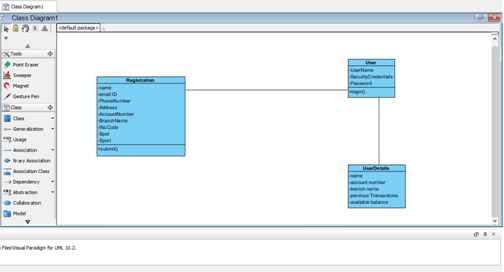
Use case Diagrams represent the functionality of the system from a user’s point of view. Use cases are used during requirements elicitation and analysis to represent the functionality of the system. Use cases focus on the behavior of the system from external point of view.

Actors are external entities that interact with the system. Examples of actors include users like administrator, bank customer …etc., or another system like central database

****

**4.3.2. Class Diagrams:**

Class diagrams ar widely used to describe the types of objects in a system and their relationships. These model the class structure and contents using elements such as classes, packages and objects. Class diagrams describe three different perspectives when designing a system. Conceptual, specification and implementation.

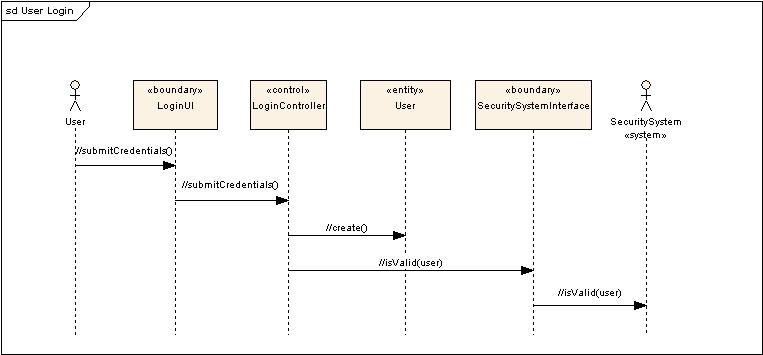
****

* + 1. **Interaction Diagrams:**

Sequence diagrams and Collaboration diagrams together called interaction diagrams. These are two of the five diagrams used in the UML for modeling the dynamic aspects of the systems. An interaction diagrams shows an interaction, consisting of a set of objects and their relationships, including the messages that may be dispatched among them.

**4.3.4. Sequence Diagrams:**

A sequence diagram shows, as parallel vertical lines(“life lines”), different processes or objects that live simultaneously and horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.



## 

## 5. DEVELOPMENT:

**About Software Development:**

Software development is the set of activities that results in software products. Software development may include research, new development, modification, reuse, re-engineering, maintenance, or any other activities that result in software products. Especially the first phase in the software development process may involve many departments, including marketing, engineering, research and development and general management.

Software development process include following steps-

* *Requirement Analysis:* The most important task in creating the software product is extracting the requirements or requirement analysis. Frequently demonstrating live code may help reduce the risk that the requirements are incorrect. Once the general requirements are gleaned from the client, an analysis of scope of the development should be determined and clearly stated.
* *Specification:* It is the task of precisely describing the software to be written. In practice, most successful specifications are written to understand and fine-tune applications that are already developed. These are most important for external interfaces that must remain stable.
* *Architecture:* The architecture of the system refers to an abstract representation of the system. It is concerned with making sure the software system will meet the requirements of the product.
* *Design, implementation and testing:* Implementation is the part of process where software engineers actually program the code for project. Software testing is integral and important part of the software development process. This part of the process ensures that bugs are recognized as early as possible.
* *Deployment and maintenance:* Deployment starts after the code is appropriately tested, is approved for release and sold. Maintenance and enhancing software to cope with newly discovered problems or new requirements can take far more time than the initial development of software.

Mainly this application is developed to reduce the physical work of maintaining through log books thus saving paper and making retrieval and access easy.

## 6.IMPLEMENTATION

## 6.1. CODE:

**Register.html**

Register.html

<html>

<head>

<title>registration page</title>

</head>

<body>

<form name=”f” method=”post” action=”register.java”>

<label>USERNAME:<input type=”text” name=”un”/></label>

<label>PASSWORD:<input type=”hidden” name=”pw”/></label>

<label>FULLNAME:<input type=”text” name=”n”/></label>

<label>CONTACT NUMBER:<input type=”text” name=”c”/></label>

<label>EMAIL ID:<input type=”text” name=”id”/></label>

<label>ADDRESS:<input type=”text” name=”a”/></label>

<label>ACCOUNT NUMBER:<input type=”text” name=”an”/></label>

<label>BRANCH NAME:<input type=”text” name=”bn”/></label>

<label>SPORT:<input type=”text” name=”g”/></label>

<label>SPOT:<input type=”text” name=”s”/></label>

</form>

</body>

</html>

**Register.java**

importjavax.servlet.http.\*;

importjavax.servlet.\*;

import java.io.\*;

importjava.sql.\*;

public class register extends HttpServlet

{

public void doPost(HttpServletRequestreq,HttpServletResponse res) throws ServletException,IOException{

res.setContentType("text/html");

PrintWriter pw=res.getWriter();

String username=req.getParameter(“un”);

String password=req.getParameter(“pw”);

String fullname=req.getParameter(“n”);

String contact number=req.getParameter(“c”);

String emailid=req.getParameter(“id”);

String address=req.getParameter(“a”);

String account number=req.getParameter(“an”);

String branchname=req.getParameter(“bn”);

String sport=req.getParameter(“g”);

String spot=req.getParameter(“s”);

try{

Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");

Connection con=DriverManager.getConnection("jdbc:odbc:suresh","system","tiger");

Statement st=con.createStatement();

String query="insert into emp1 values('"+un+"','"+pw+"','"+n+"','"+c+"','"+id+"','"+a+"','"+an+"','"+bn+"','"+g+"','"+s+"')";

int s=st.executeUpdate(query);

if(s>0){

pw.println("you have been successfully registered");

}

else{

pw.println("registration failed");

}

}

catch(Exception e){

pw.println(e.getMessage());

}

}

}

**Web.xml**

<?xml version="1.0" encoding="ISO-8859-1"?>

<web-app>

<servlet>

<servlet-name>register</servlet-name>

<servlet-class>register</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>register</servlet-name>

<url-pattern>/register</url-pattern>

</servlet-mapping>

</web-app>

**Login.html*:***

<html>

<head>

<title>login page</title>

</head>

<body>

<form method=”get” action=”auth.java”>

<input type=”text” name=”username”>

<input type=”submit” value=”continue”>

</form>

</body>

</html>

**Auth.java:**

importjavax.servlet.\*;

importjavax.servlet.http.\*;

import java.io.\*;

importjava.sql.\*;

public class authen1 extends HttpServlet

{

public void doPost(HttpServletRequestreq,HttpServletResponse res) throws ServletException,IOException{

int flag=0;

res.setContentType("text/html");

PrintWriter pw=res.getWriter();

String user=req.getParameter(“un”);

try{

Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");

Connection con=DriverManager.getConnection("jdbc:odbc:suresh","system","tiger");

Statement st=con.createStatement();

String query="select \* from emp1 where username='"+user+"' ";

ResultSetrs=st.executeQuery(query);

if(rs.next()){

RequestDispatcherrd=req.getRequestDispatcher(“/1.html”);

}

else{

pw.println("unauthenticated user");

}

}

catch(Exception e){

pw.println(e.getMessage());

}

}

}

**secdetails.html**

<html>

<head>

<title>additional information</title>

</head>

<body>

<form method=”post” action=”authentication.java”>

<label>choose your favourite sport:

<input type=”checkbox” name=”c”><imgsrc=”cricket.jpg” width=”10%” height=”20%”>&nbsp;

<input type=”checkbox” name=”c” ><imgsrc=”football.jpg” width=”10%” height=”20%”>&nbsp;

<input type=”checkbox” name=”c” ><imgsrc=”carroms.jpg” width=”10%” height=”20%”>&nbsp;

<input type=”checkbox” name=”c” ><imgsrc=”chess.jpg” width=”10%” height=”20%”>&nbsp;

</label>

<label>choose your favourite spot:

<input type=”checkbox” name=”c1” ><imgsrc=”ooty.jpg” width=”10%” height=”20%”>&nbsp;

<input type=”checkbox” name=”c1” ><imgsrc=”kodeikenal.jpg” width=”10%” height=”20%”>&nbsp;

<input type=”checkbox” name=”c1” ><imgsrc=”alipi.jpg” width=”10%” height=”20%”>&nbsp;

<input type=”checkbox” name=”c1” ><imgsrc=”darjiling.jpg” width=”10%” height=”20%”>&nbsp;

</label>

<input type=”submit” value=”proceed”/>

</form>

</body>

</html>

**Authentication.java*:***

importjavax.servlet.\*;

importjavax.servlet.http.\*;

import java.io.\*;

importjava.sql.\*;

public class authen1 extends HttpServlet

{

public void doPost(HttpServletRequestreq,HttpServletResponse res) throws ServletException,IOException{

int flag=0;

res.setContentType("text/html");

PrintWriter pw=res.getWriter();

String sports=req.getParameter(“g”);

String spots=req.getParameter(“s”);

try{

Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");

Connection con=DriverManager.getConnection("jdbc:odbc:suresh","system","tiger");

Statement st=con.createStatement();

String query="select \* from emp1 where username= ‘”+user+”’andsport='"+sports+"' and spot='"+spots+"'";

ResultSetrs=st.executeQuery(query);

if(rs.next()){

RequestDispatcherrd=req.getRequestDispatcher(“/2.html”);

}

else{

pw.println("unauthenticated user");

}

}

catch(Exception e){

pw.println(e.getMessage());

}

}

}

**Password.html**

<html>

<head>

<title>login page</title>

</head>

<body>

<form method=”get” action=”auth2.jsp”>

<input type=”hidden” name=”password”>

<input type=”submit” value=”continue”>

</form>

</body>

</html>

**Auth2.jsp**

<html>

<head>

<title>abc</title>

</head>

<body>

<%@page import=”java.sql.\*”%>

<h1>list of users</h1>

<%

class.forName(“sun.jdbc.odbc.JdbcOdbcDriver”);

Connection con=DriverManager.getConnection("jdbc:odbc:suresh","system","tiger");

Statement st=con.createStatement();

String query="select \* from emp1 where username='"+user+"' and password=’”+pwd+”’ ";

ResultSetrs=st.executeQuery(query);

%>

<table border=”1”>

<tr>

<th>USERNAME</th>

<th>PASSWORD</th>

<th>FULLNAME</th>

<th>CONTACT NUMBER</th>

<th>MAIL ID</th>

<th>ADDRESS</th>

<th>ACCOUNT NUMBER</th>

<th>BRANCHNAME</th>

<th>FAVOURITE SPORT</th>

<th>FAVOURITE SPOT</th>

</tr>

<%while(rs.next())

{

%>

<tr>

<td><%=rs.getString(1)%></td>

<td><%=rs.getString(2)%></td>

<td><%=rs.getString(3)%></td>

<td><%=rs.getInt(4)%></td>

<td><%=rs.getString(5)%></td>

<td><%=rs.getString(6)%></td>

<td><%=rs.getInt(7)%></td>

<td><%=rs.getString(8)%></td>

<td><%=rs.getString(9)%></td>

<td><%=rs.getString(10)%></td>

</tr>

<%

}

rs.close();

st.close();

con.close();

%>

</body>

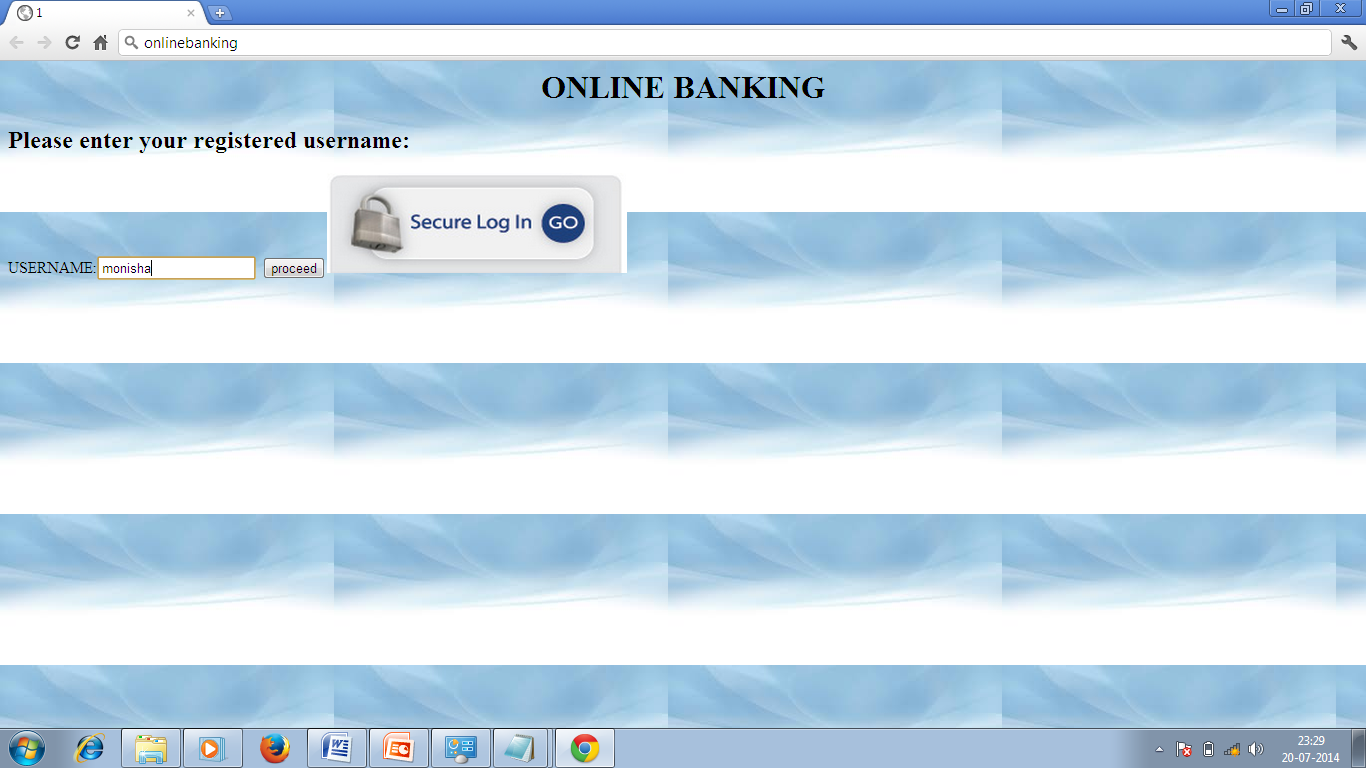
</html>

**6.2 SCREEN SHOTS**

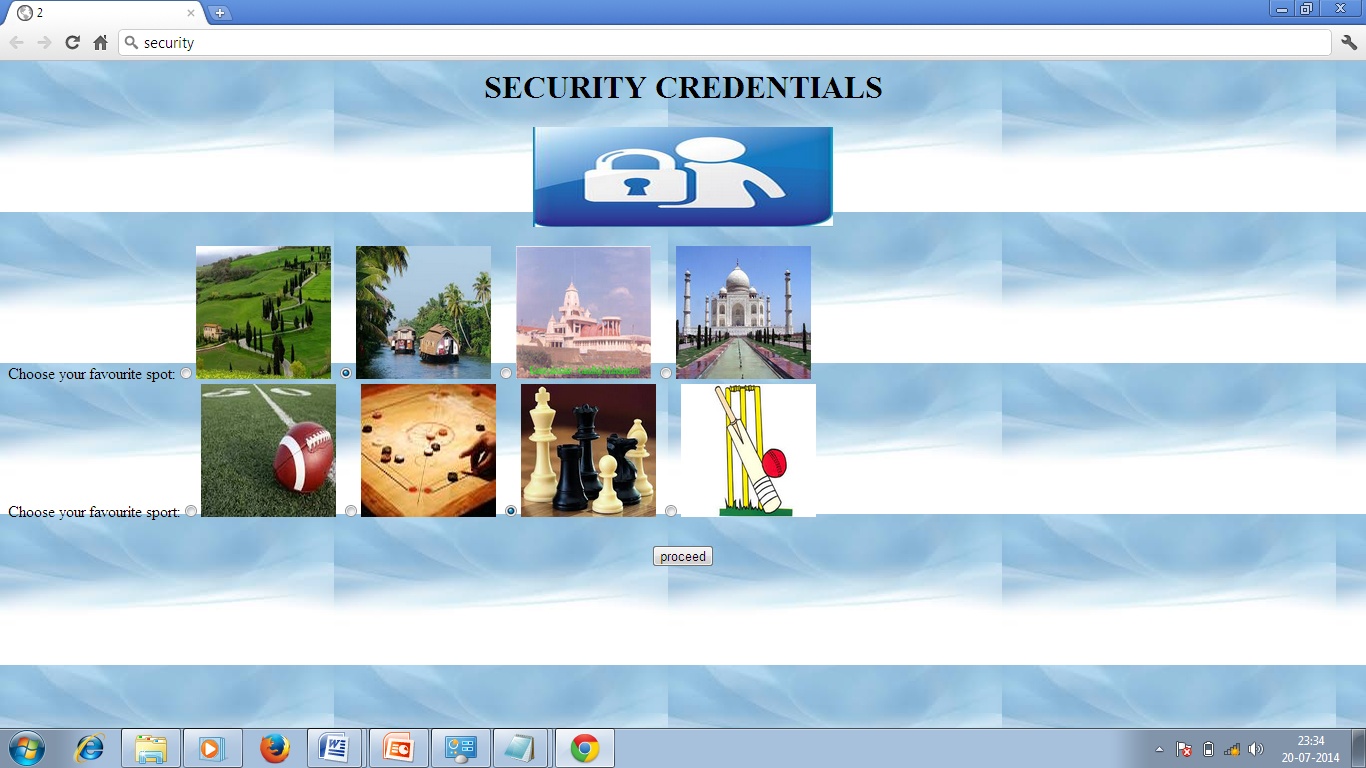
Registration page for unregistered users:

****

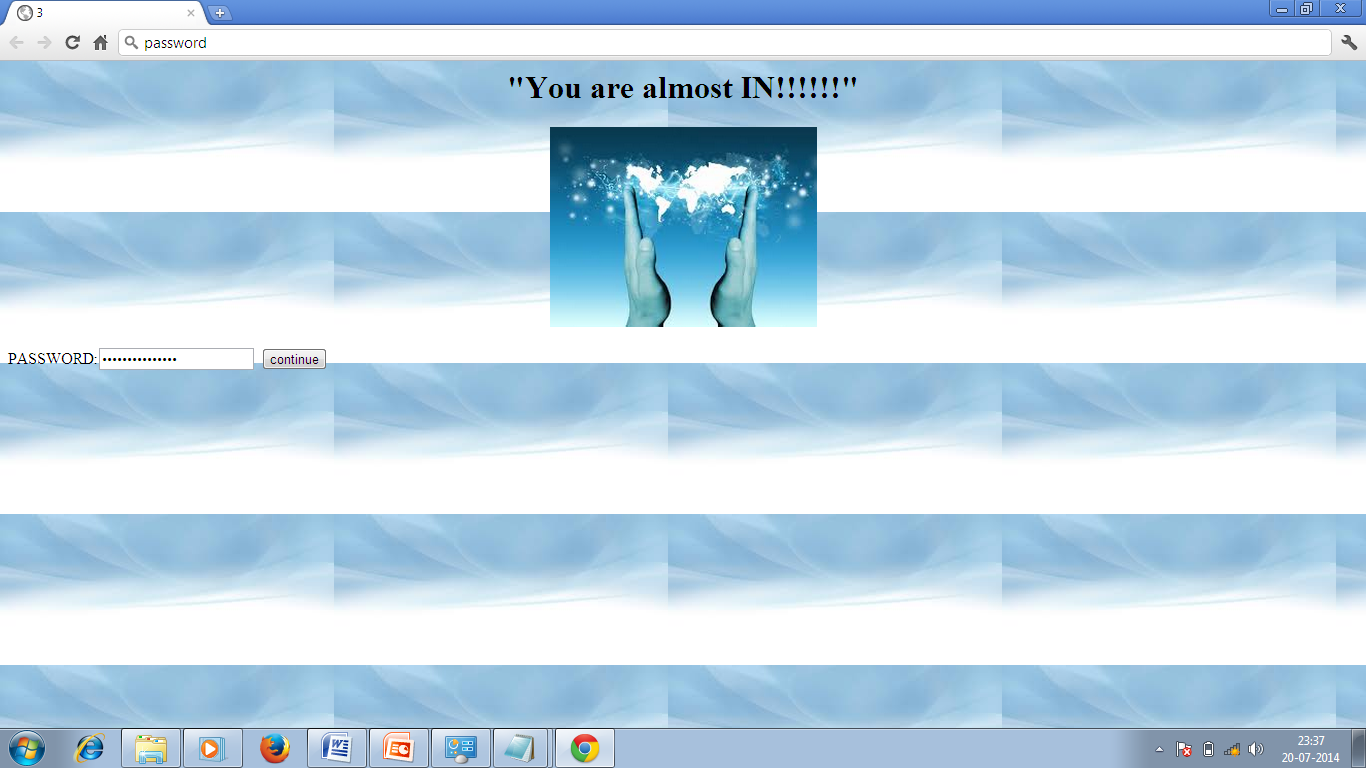
**Login page:**

****

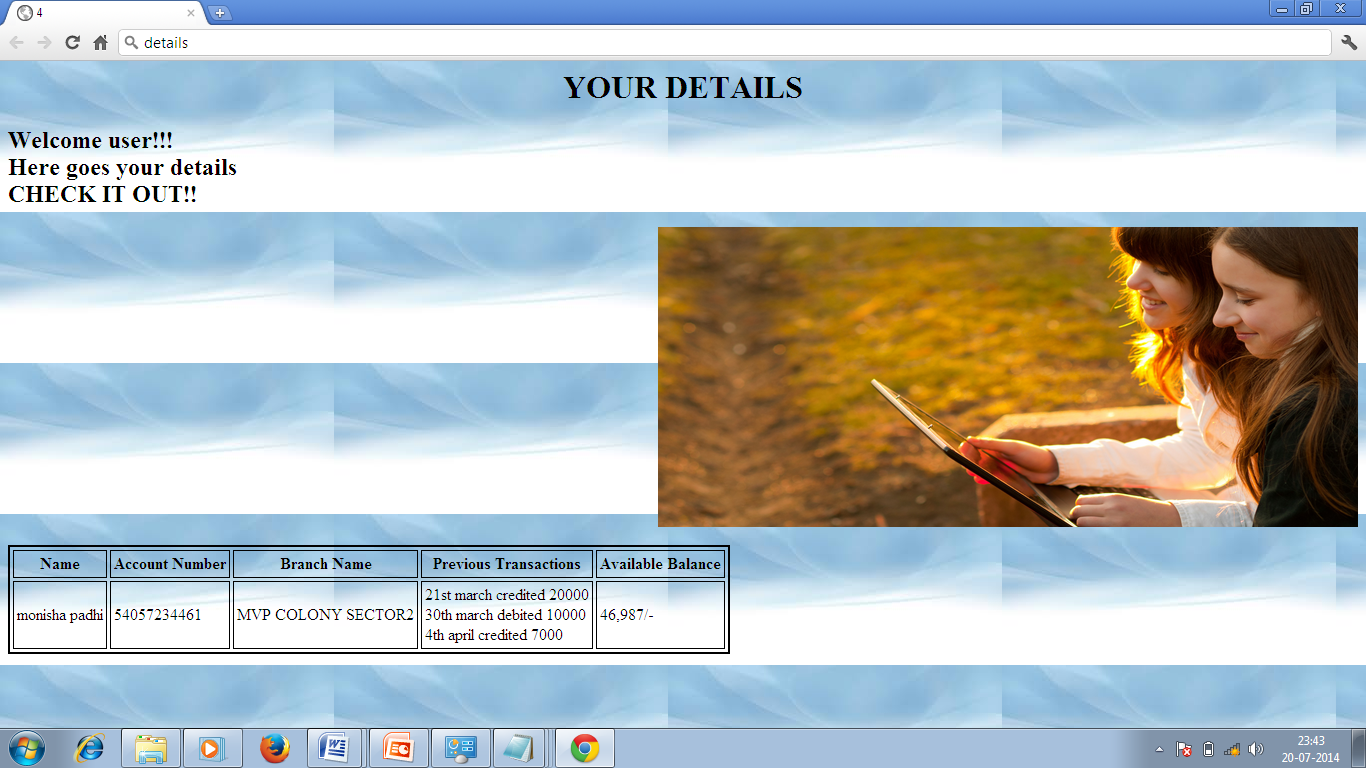
**User must provide his security details to access his account**

****

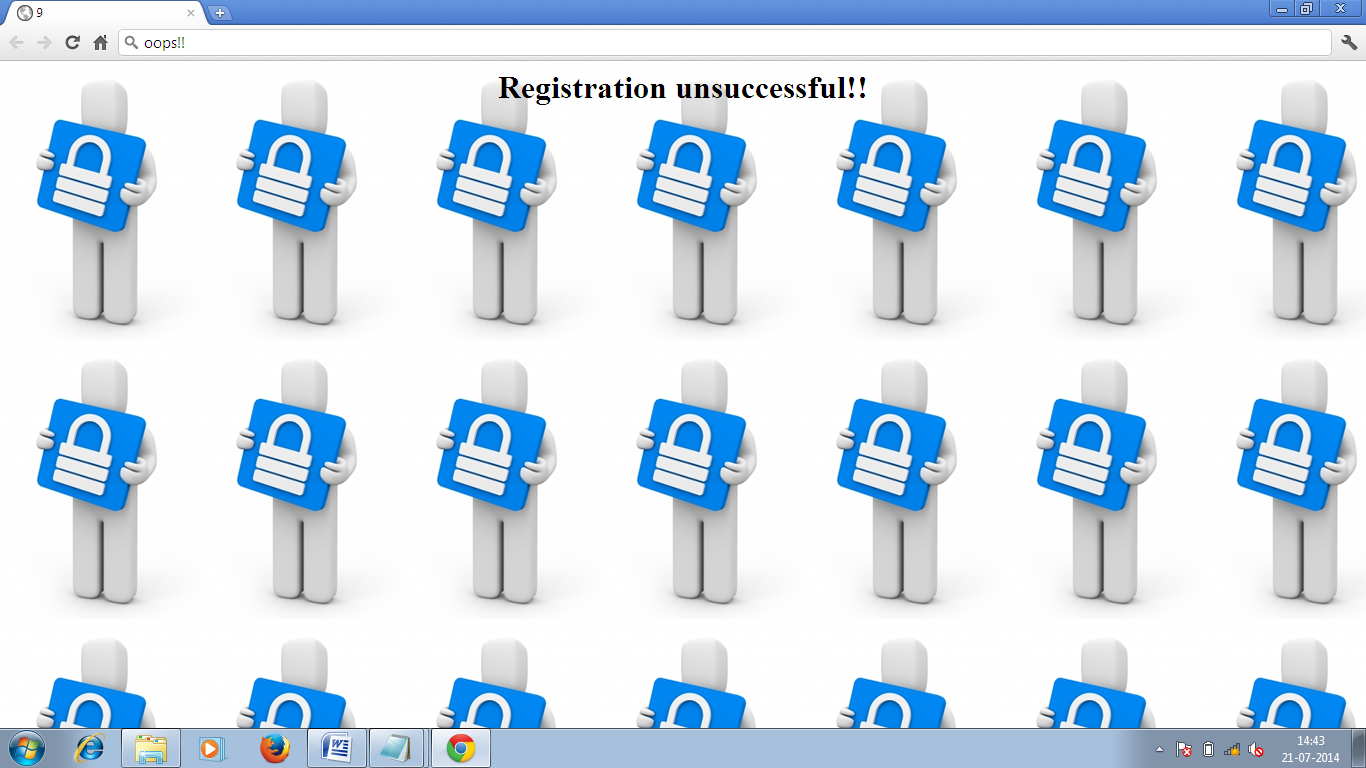
**After providing his security details he will be redirected to the next page:**

****

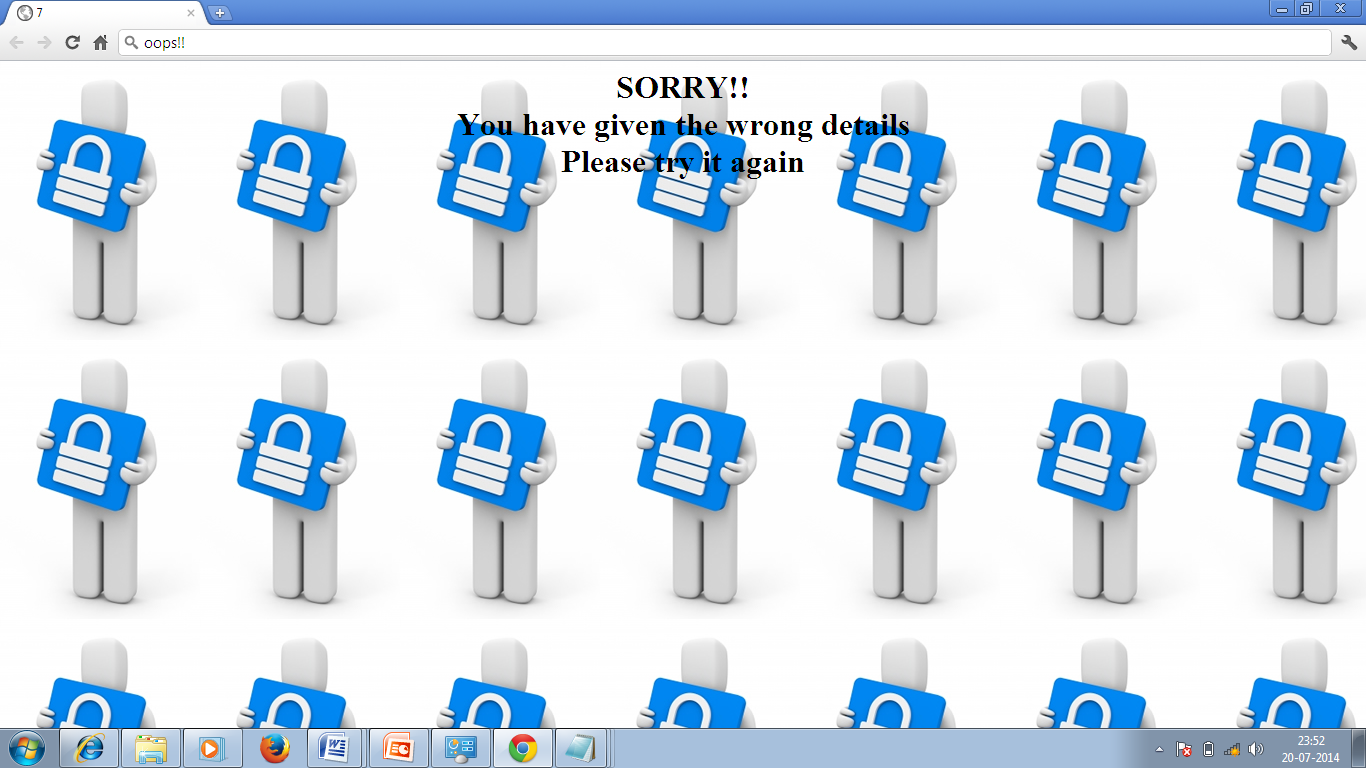
**User details page:**

****

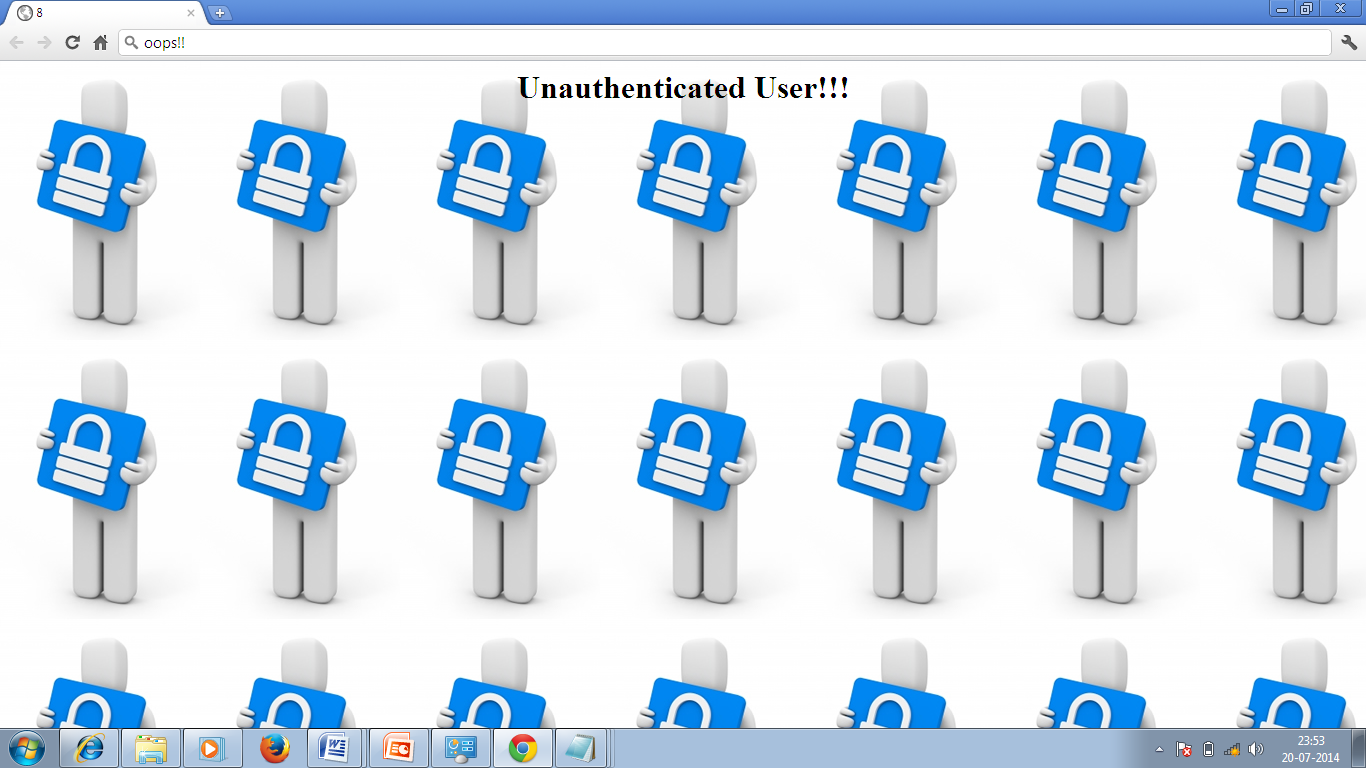
**If any fields are left blank while registration or any error occurs, this page will be displayed:**

****

**If the user gives wrong username, this page will be displayed:**



**If the user did not provide correct security details and password, this page will be displayed:**



**TESTING:**

## 7.1 INTRODUCTION

The development of software involves series of productive activities and testing is an important activity of them. This phase is a critical element of software quality assurance and represents the ultimate review of specification, coding and testing.

The main objectives of testing are as follows:

* Testing is a process of executing a program with the intent of finding an error.
* A good test case is one that has a high probability of finding an undiscovered error.
* A successful test is one uncovers an undiscovered error.

Testing can be done in different ways. Some of the types of testing are mentioned below. The main purpose of any type of test is to systematically uncover different classes of errors and do so with a minimum amount of time and effort.

## 7.2 TYPES OF TESTING

* Unit testing
* Integration testing
* Regression testing
* System testing
* Alpha testing
* Beta testing

Testing can be done manually or by using testing tools. There are several testing tools for different software.

**Unit Testing:**  It is a method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures, are tested to determine if they are fit for use.

**Integration Testing:** It is the phase in software testing in which individual software modules are combined and tested as a group Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates, and delivers as its output the integrated system ready for system testing.

**Regression Testing**: Regression testing is any type of software testing that seeks to uncover new software bugs, or regressions, in existing functional and non-functional areas of a system after changes, such as enhancements, patches or configuration changes, have been made to them.

**System Testing:** System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements.

**Alpha Testing:** Alpha testing is simulated or actual operational testing by potential users/customers or an independent test team at the developers' site. Alpha testing is often employed for off-the-shelf software as a form of internal acceptance testing, before the software goes to beta testing.

**Beta Testing:** Beta testing comes after alpha testing and can be considered a form of external user acceptance testing. Versions of the software, known as beta versions, are released to a limited audience outside of the programming team. The software is released to groups of people so that further testing can ensure the product has few faults or bugs. Sometimes, beta versions are made available to the open public to increase the feedback field to a maximal number of future users.

**Each module can be tested using the following two strategies:**

**Black Box Testing:** In this strategy some test cases are generated as input conditions that fully execute all functional requirements for the program. This testing is used to find errors in the following categories:

* Incorrect or missing functions
* Interface errors
* Errors in data structure or external database access
* Performance errors
* Initialization and termination errors

In this testing, only the output is checked for correctness. The logical flow of the data is not checked.

**White Box Testing:** In this test cases are generated on the logic of each module by drawing flow graphs of that module and logical decisions are tested on all the cases.

**7.3 TEST CASES:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TEST NO | TESTCASE DESCRIPTION/ACTION | INPUT | EXCEPTED  OUTPUT | OBSERVED  OUTPUT |
| 1. | Validate login | Enter the username and password without registering. | User must not get logged in and the same page should appear. | User will not get logged in and the same page will appear. |
| 2. | Validate login | Enter correct username and password. | User must get redirected directly to the security details page. | User got redirected to the security details page. |
| 3. | Validate username | Enter the wrong username. | User must not get logged in and the same page should appear with error message. | User will not get logged in and the same page will appear with error message. |
| 4. | Validate username | Any of the fields left blank. | An error message “please enter username” should appear. | An error message “please enter username” appeared. |
| 5. | Verify security details | Any one of the details did not match. | Error message “please enter the correct details” should appear. | Error message “please enter the correct details” appeared. |
| 6. | Verify security details | All the details provided are correct. | User will be redirected to the password entering page. | User will be redirected to password entering page. |
| 7. | Validate password | Enter the wrong password. | User must not get redirected to the next page and error message will be displayed. | User will not get redirected to the next page and error message will be displayed. |
| 8. | Validate password | Enter the correct password. | User must be redirected to the details of his/her account. | User will be redirected to the details of his/her account. |

**8. CONCLUSION:**

* There are several ways for performing secured authentication.
* Our solution mainly concentrates on providing extra security so that only registered users can enter into their account and have their work done.
* Our solution is useful to the user to use his account safely with complete trust.
* This module provides the best security for online banking.

**9. REFERENCES:**

* http://www.w3schools.com/html/default.asp
* The Complete Reference Java 5th Edition by Herbert Schildt
* http://www.jsptut.com/
* Internet and world wide web by Dietel and Nieto
* Java server pages by Hans Bergsten
* Sql and pl/sql for oracle 10g,black book,Dr.P.S.Deshpande