P.A.S.S

A WEB-BASED ONLINE EXAMINATION SYSTEM

CSE 5306 – DISTRIBUTED SYSTEMS
2015 SPRING – SECTION 1

ARCHANA MOHAN

POOJITHA KARLAPALEM

SRIJA REDDY PATEL

UMA NAGA SAI JYOTHI KOTAMRAJU

The University of Texas at Arlington – April, 2015

1. INTRODUCTION

1.1 BACKGROUND

P.A.S.S is a web based portal which is designed for providing an automated platform for conducting examinations. This reduces the complexity and effort involved in assessing candidates manually. The system can be used by schools, universities, and other private educational institutions to conduct tests, examinations and quizzes on a regular basis.

1.2 OBJECTIVE

- The most important goal of our system is to reduce manual effort involved in conducting an examination
- Provide separate views of the same system for both students and examiners, and also for Graduate students and Undergraduate students
- Replicate all the important features of a manually-assisted examination:
    - Allow multiple students to take the examination at the same time
    - Allow timed examinations
    - Facilitate the examiner to create topics and set questions
    - Allow examiner to upload any other course material for the examination
- Advantages over a manual examination:
    - Fully automated grading system
    - Reduced paper-work and manual calculations
    - Can be used anywhere, at any time
    - Result will be displayed in less time.

1.3 HIGH LEVEL DESCRIPTION

1.3.1 DESIGN

The system consists of a database of questions, a web server to serve the user requests, replicated servlet containers to handle server failures, and users of the system. This system has been designed to facilitate two kinds of end users: Faculty, and Students.
There is an Administrator that manages these end users, their account details and manages the overall accessibility of the system.

1.3.2 USER FUNCTIONALITY

The Administrator is allowed to perform the following tasks:

- Creating and managing accounts for students and faculty
- Making sure that the account information is secure and protected from unauthorized users
- Monitoring the user requests and the server functionality

The Faculty can perform the following operations on the system:

- Adding a new main topic
- Adding a new subtopic
- Uploading course related files
- Creating a new exam
- Updating(adding, deleting, modifying) questions in the database
- Adding any description/details/comments about the course and exam

The Candidate is provided with the following features:

- Viewing the main topics and subtopics
- Viewing available examinations to attend
- Take an examination
- Download any available course files
- Viewing exam results

1.4 USE CASE

TYPICAL EXAMINATION SCENARIO:

For each examination, there is a set of specific questions and a time period set by the examiner during which the student has to complete the exam. One question is displayed at a time, and the student is given an option to move forward to the next question. The remaining time is always displayed on the top for the student’s reference. Once all the questions have been answered, the student will be able to submit their exam.

The result is automatically evaluated by the system and displayed to the student at the end of the exam (suited best for Multiple-Choice Questions).

1.5 ORGANIZATION OF THIS REPORT

1. Introduction: This section gives a brief description of the whole system. It consists of Background, Objective, High-level Description, Use cases and organization of this report. The high-level Description section describes the Design and User functionality of the system.
2. Related Work: The previously existing system and the real life use of it has been described under this section.
3. System Overview: The system overview give description about Components, Architecture and Communication of the system.
4. Detailed Design: The components, Challenges and solutions for the challenges are given in this section.
5. Implementation: The environment in which the system is implemented is discussed in this section which includes the server-side scripting, RDBMS & MySQL, Web Server replication (Apache Tomcat), Virtual hosting, Client-side programming and work division.

6. Theoretical/ Simulation Study: Theoretical values of overheads for communication, storage and computation are discussed in this section.
7. Future Work: The additional features that can be implemented in future work of this system are listed in this part of the document.
8. References: The references used for this document are listed.

2. RELATED WORK

The abundant world-wide availability of computer systems and the extensive use of internet has led to the development of online systems for examinations ranging from institution-specific examinations to state, national, and even international level examinations. Some of the most significant American standardized tests such as the SAT, GRE, USMLE, TOEFL, etc. are all web based examination systems that are taken by thousands of candidates every year.

Most of these examinations can be taken at any scheduled time in many different countries. Many of them implement completely or partially automated grading systems, multiple subject-based test sections, smart selection of question sets based on previous answers of the candidate, and many other advanced technical features that a traditional paper examination does not support.

EXAM-SOFTWARE TEST ENGINE:

One of the popular online examination software is the **Exam-software Test Engine**.

Exam Software is a Test Management software to create and conduct computer based online examination. Useful for school, university, and private institutions teachers for managing and conducting examinations. Some recruiting agencies and private firms also use it for assessing candidates based on the online screening tests.

3. SYSTEM OVERVIEW

3.1 COMPONENTS

The system follows a simple client-server architecture which consists of a database directory, two session-replicated web servers, and multiple parallel clients.

1. Database Directory: To store all the databases and data for all the users. The databases consist of user information (login, passwords), main topics, sub topics, examination questions, student answers, grades, etc.

2. Web Server: A web server is used to communicate between the clients and the database. The communication is done through HTTP pages. It is used to process, store and deliver pages to the clients. Our system also consists of a replicated server to handle server failures.

3. Clients:
    - Administrator: Administrator maintains the details of faculty and student accounts. Also restricts database access to unprivileged users.
    - Faculty: Faculty has the privilege to add, modify and delete the main topics, sub-topics, exam papers and other course related content. They also manage registered students, and create new examinations.
    - Students: Students can get information about their registered courses, download available course content, and take any associated exams.

3.2 SYSTEM ARCHITECTURE

Web-based Online Examination System is designed based on a two-tiered architecture. It consists of the following two main divisions:

- Client-side
- Server-side

The client side of the architecture consists of the following three clients - Administrator, Faculty and Students.

The server side consists of a web server, two session-replicated servlet containers and a Database. All user requests to access the database will have to pass through the web server, which provides access based on the type of the user. The server then fetches data from the database and presents it to the client.

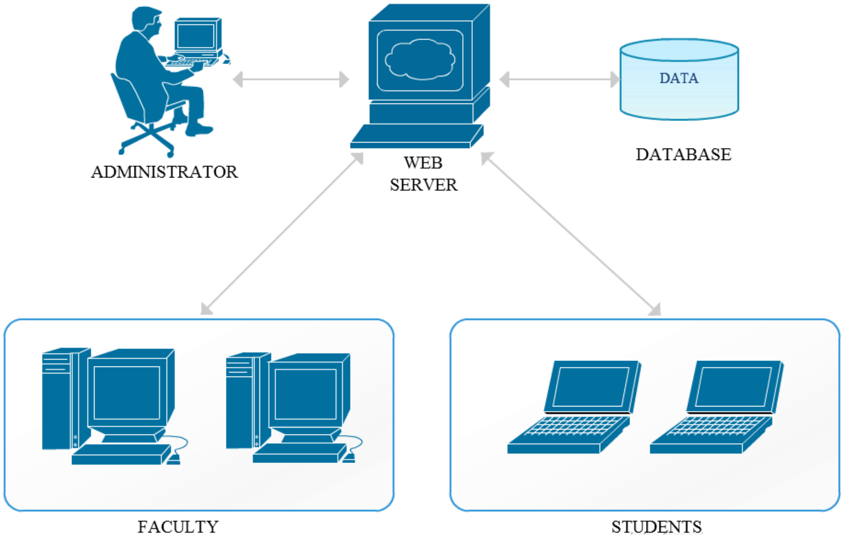


Figure 3.1 System Architecture

3.2.1 REPLICATED SERVER ARCHITECTURE

Server replication is done by clustering. The server system consists of an Apache HTTP server and multiple nodes of Apache Tomcat (servlet containers) which replicate session. The HTTP server sits in the front acting as a load balancer for the Tomcat nodes.

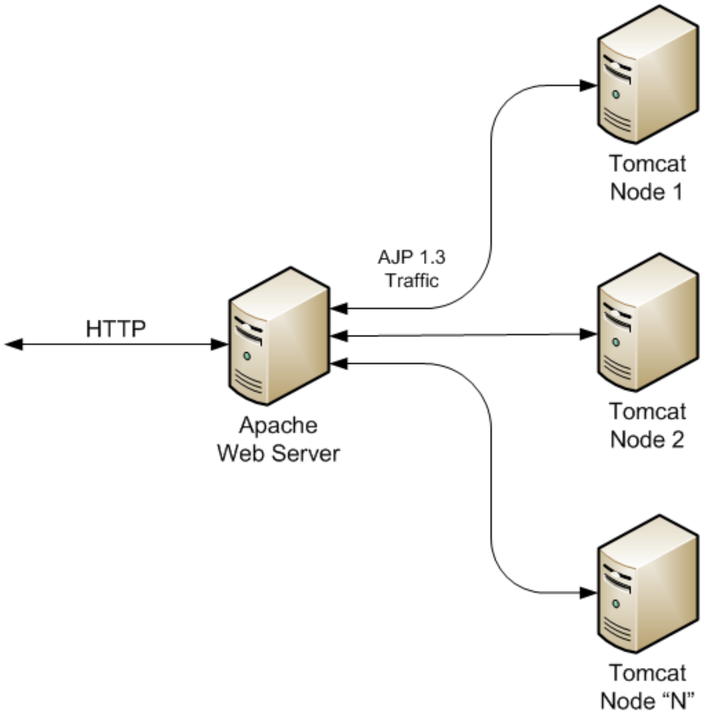


Figure 3.2 Tomcat Cluster Architecture
Source: http://www.richardnichols.net

3.3 COMMUNICATION

1. All the components communicate with each other through wired/ wireless networks, and internet, which holds this distributed network together to route data. We have used a free-version of the virtual router called Connectify, which wirelessly connects two computers.
2. Administrators, faculty, and students all get to interact directly with the web server, which then lets one of the servlet containers, through APJ 1.3 protocol, handle these requests to access data from the database.
3. Client is typically a web browser that uses the HTTP protocol to transfer web pages to and from the server. Protocols at each TCP/IP stack standardize the flow of data.
4. The faculty access the database through the web server for posting questions.
5. The students take the exams through the web server, the questions are stored in the database.

4. DETAILED DESIGN

4.1 COMPONENTS

1. **Database Directory:** Consists of all the data created for and used by the examination system. A Database Management System is used for creating, manipulating, maintaining, and sharing these databases among multiple users. We will maintain three kinds of databases:
   - Faculty database: Each course will maintain its own database in which teachers can add, delete, or modify their respective course examination questions.
   - Student database: For students to store information about the courses they're enrolled in, upcoming exams, past grades, etc.
   - Administrator databases: To store all the administrator related content such as usernames, passwords, and other user account information which is accessible only to the administrator.

2. **Web Server** acts a mediator between the clients and the database. It is responsible for implementing the entire logic of the system. The web server receives requests from clients, and passes them to the servlet container that is first available to it. We have a servlet cluster, i.e, an HTTP server and multiple servlet containers (Tomcat nodes) to handle failures and to balance heavy loads. The apache HTTP server acts as the load balancer here.

   We have implemented a request level clustering along with session replication, where each request may go to a different node - this is ideal since the traffic would be balanced across all nodes, and if a node goes down, the user will have no idea about it. This method of load balancing across nodes allows us to scale our application out horizontally across many machines.

3. **User Interface**, as a web browser, allows clients to interact with the system easily from any web enabled device. When clients get registered to the system he/ she will be permitted to access the content. User interface displays the desired results to clients.

   In this project we have 3 kinds of users- Administrator, Instructor, and Student.
   - Administrator handles the database containing account details of instructors and students.
   - Instructors register, login, add their main topics, subtopics, upload course related files and exam question sets.
   - Students register, login to the system with their id/password, take the exam, and view the results.

4. **Connection** among the devices: We have use a WiFi HotSpot called Connectify to connect the devices wirelessly. *Connectify is a PC Virtual Router which creates a virtual User interface, as a web browser, allows clients to interact with the system easily from any web enabled device*. (Source: www.connectify.me)

4.2 CHALLENGES FACED

1. Parallel server requests by multiple clients
2. Database isolation. Data should be accessible to only their respective users, i.e,
    - Faculty and students should not be able to access stored account information about other users
    - Students should not be able to access faculty databases. eg: examination questions, modification of grades, etc.
    - Administrator should not be able to access any of the course related faculty and student information
3. Server crash during an examination, or abrupt shutdown of a server machine.

4.3 SOLUTIONS PROVIDED

1. Synchronization using threads, mutexes.
2. The database is accessible according to the role of the user. Access rights have been given to the users by the administrator.
3. Session is maintained through server clustering and replication. While one server node crashes and recovers, another active node handles the subsequent user requests while maintaining the same session state, and the user will not have any idea about failure of the initial server node.

5. IMPLEMENTATION

Some of the key features that have been implemented in our system are:

- Separate systems for Graduate and Undergraduate students.
- Faculty can add multiple Main Topics, Subtopics, upload course related files, and upload Examination question sets. Students can view these main topics, subtopics, download the course related files, view available exams, and take an examination.
- Each student is allowed to take the same examination only up to 3 times, and the most recent grade is recorded.
- Server cluster can be scaled to contain up to 40 servers. Hence, the system will work seamlessly even after 39 servers fail, and **the users will have no idea about it**.
- The website can be accessed from any type of client machine: PC’s, Tablets, Mobile phones. Basically, any device that contains a web browser.

5.1 SERVER-SIDE SCRIPTING USING SERVLETS

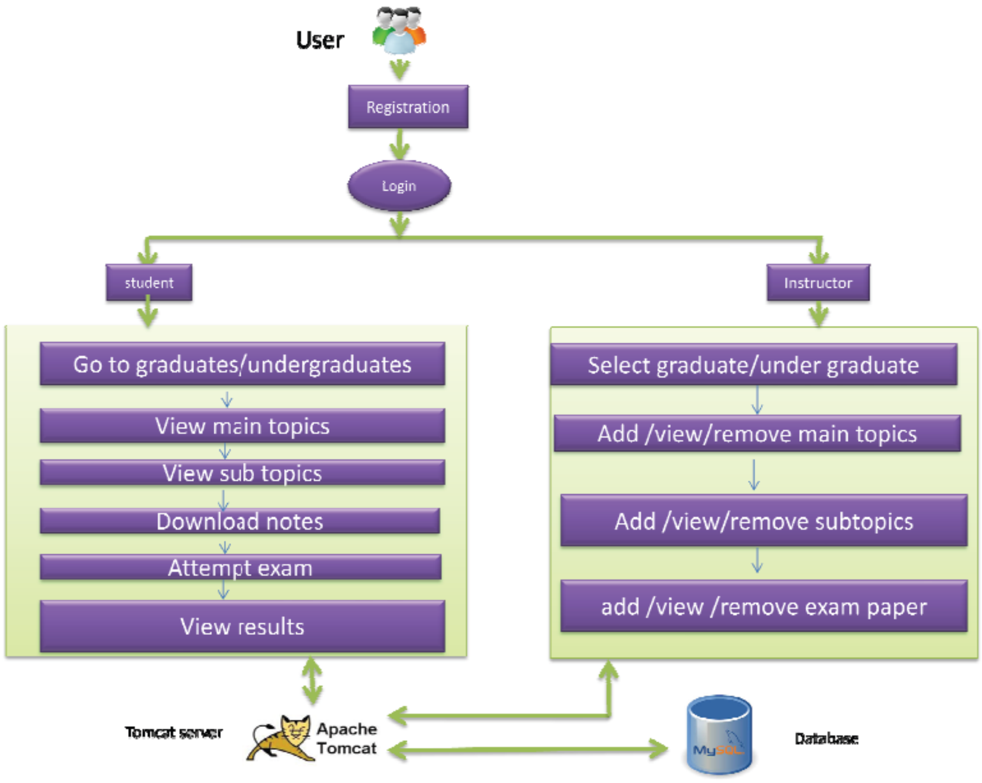


Fig 5.1 Work flow of the system

The entire project is divided into four broad Modules:

- Module 1:
  Authentication and authorization required for the students and professors who want to get connected.

- Module 2:
  The courseware is divided on the basis of graduates and undergraduates.
  Depending on which user is logged in the respective syllabus and downloads is displayed.

- Module 3:
  Users can login into the website they can write the test automatically the timer will start after completion of the test he/she can see the result.

- Module 4:
  Users can view/download the materials based on their interest.
  Note: Users can take exam only 3 times on his/her account.

The server-side scripting of this project is developed in java using Eclipse IDE. JRE 1.6 or higher version is installed in order to run the program.

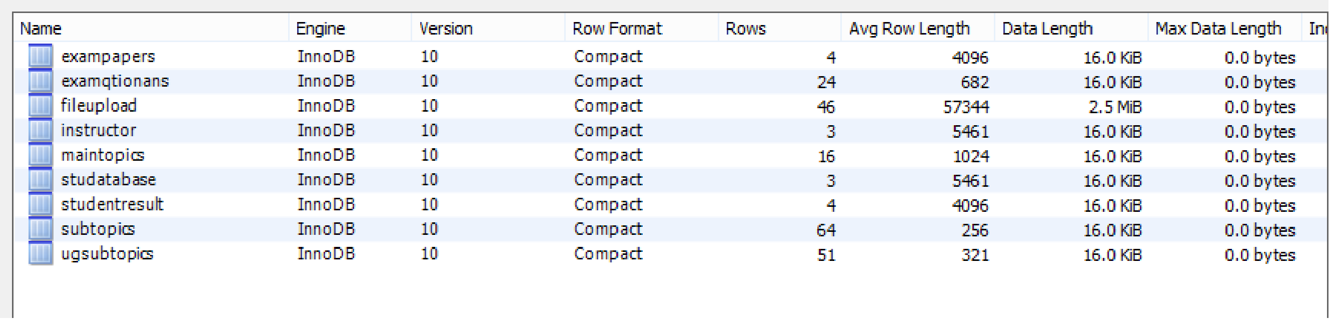
Reasons for choosing Java:

- Java helps us create real world applications and is used to improve the application without rewriting the entire code. Unlike C and C++, Java is platform independent as we can run a code anywhere on any operating system. This is particularly advantageous in a distributed system like ours.
- Java has been used for its best known features: distributed and multi-threading. Distributed services enables multiple servers to run at a time that provides the services required by multiple clients. This can be developed using J2EE application.

System Requirements:

1. Operating system: Any OS that supports JVM
2. Software: JDK, J2SDK.
3. Integrated Development Environment: Any Java IDE like Eclipse.

5.2 RDBMS & MySQL



Name	Engine	Version	Row Format	Rows	Avg Row Length	Data Length	Max Data Length	In
exampapers	InnoDB	10	Compact	4	4096	16.0 KiB	0.0 bytes	
examqtionans	InnoDB	10	Compact	24	682	16.0 KiB	0.0 bytes	
fileupload	InnoDB	10	Compact	46	57344	2.5 MiB	0.0 bytes	
instructor	InnoDB	10	Compact	3	5461	16.0 KiB	0.0 bytes	
maintopics	InnoDB	10	Compact	16	1024	16.0 KiB	0.0 bytes	
studatabase	InnoDB	10	Compact	3	5461	16.0 KiB	0.0 bytes	
studentresult	InnoDB	10	Compact	4	4096	16.0 KiB	0.0 bytes	
subtopics	InnoDB	10	Compact	64	256	16.0 KiB	0.0 bytes	
ugsubtopics	InnoDB	10	Compact	51	321	16.0 KiB	0.0 bytes	

Fig 5.2 Relations in the Database

MySQLWorkbench has been used in our project to create the entire database.
The Structure of the database is shown in above figure.
MySQL is Relational Database Management System which provides all the tools required to create, update, maintain, and share databases among multiple users.
MySQL Cluster is a technology which provides additional useful functionalities such as **Replication, Auto-Sharding, Hybrid Storage,** and **Shared Nothing** to the MySQL DBMS.

The following tables have been used:

- **ExamPapers**: Exam papers table stores all the values of the chapter name, test path uploaded by the Instructor.
- **Examqtioans**: This table stores the question number followed by the questions and correct Answer.

- **Fileupload**: Fileupload table stores the values of the files being uploaded and its extension.
- **Instructor**: Instructor tables consists of instructor Login details such as Name, email address and Password.
- **Maintopics**: This table consists of subject code, the name of the topic and the email of the instructor who uploaded it.
- **Studatabase**: The student details like first name, last name, graduate or undergraduate, email address and password are stored.
- **Studentresult**: The details of the results like email of the student, right answers and number of attempts are stored in this table.
- **Subtopics:** Code, subtopics and email of the instructor are stored in this table.
- **Ugsubtopics**: This table stores the code, subtopics and email of the instructor related to undergraduates.

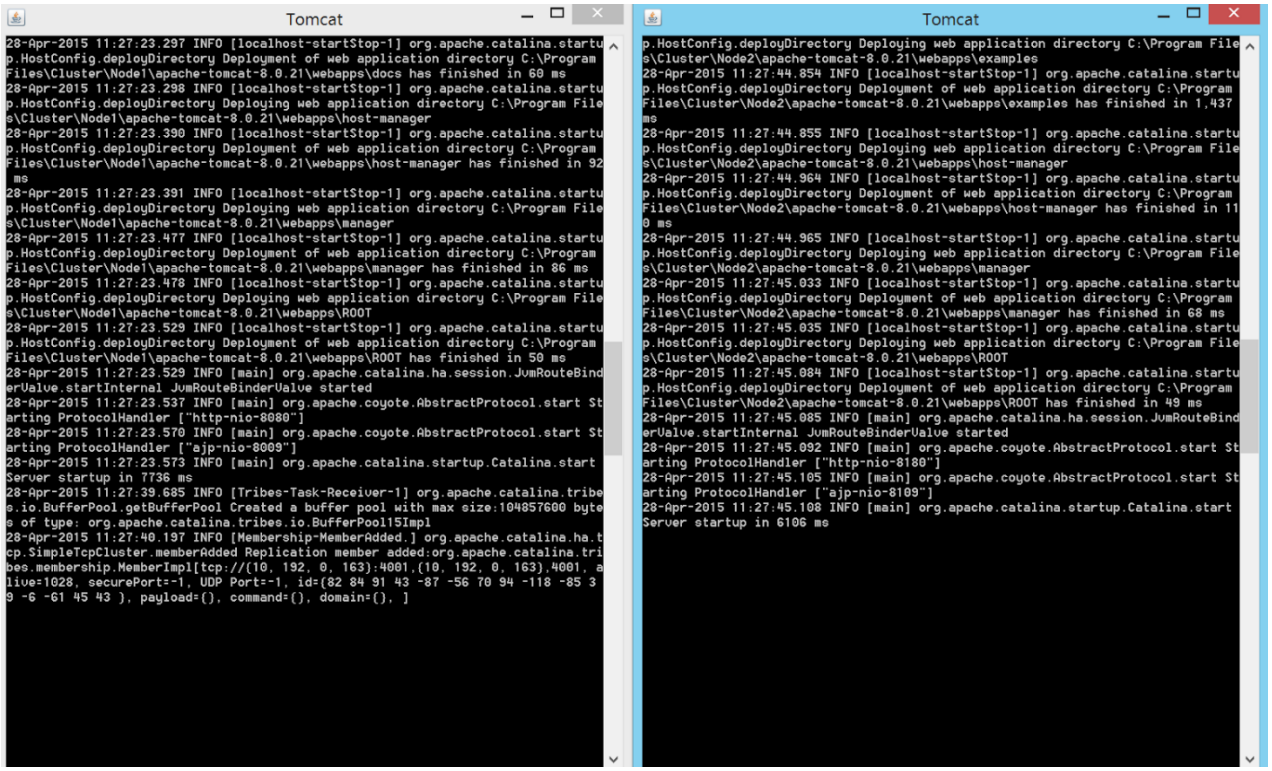
Software Requirements: MySQL DBMS, MySQL Cluster technology
IDE: MySQL command-line client

5.3 WEB SERVER AND SESSION REPLICATION

Apache HTTP server receives requests from clients, and sends them to the available Tomcat node, that runs the java web application, and delivers web pages as a response back to the client. The Apache HTTP server uses the HTTP protocol to transfer web pages to and from the client.

Implementing session replicated server cluster was a two-step process:

1. Setting up Tomcat Nodes on the same or on different machines. In this project, due to limited resources, we have implemented 2 Tomcat nodes. However, it can be scaled to support up to 40 Tomcat nodes in one single cluster. A Load balancer is also set up, which is the Apache HTTP server, and is configured across our two nodes. The Load Balancer's load factor can be modified to send traffic to one node or the other depending on our requirements.
2. The second step is session replication. We have configured our two Tomcat nodes to replicate each other's sessions using "stickysession", which is a Tomcat setting for session management. Our web application is also implemented to be distributable so that it can be distributed across our server cluster nodes. All the Java session attributes are made serializable, so that everything inside our current server node's session can be sent to the other server node for replication over the network.



```
28-Apr-2015 11:27:23.297 INFO [localhost-startStop-1] org.apache.catalina.startu
p.HostConfig.deployDirectory Deployment of web application directory C:\Program
Files\Cluster\Node1\apache-tomcat-8.0.21\webapps\docs has finished in 60 ms
28-Apr-2015 11:27:23.298 INFO [localhost-startStop-1] org.apache.catalina.startu
p.HostConfig.deployDirectory Deploying web application directory C:\Program File
s\Cluster\Node1\apache-tomcat-8.0.21\webapps\host-manager
28-Apr-2015 11:27:23.390 INFO [localhost-startStop-1] org.apache.catalina.startu
p.HostConfig.deployDirectory Deployment of web application directory C:\Program
Files\Cluster\Node1\apache-tomcat-8.0.21\webapps\host-manager has finished in 92
 ms
28-Apr-2015 11:27:23.391 INFO [localhost-startStop-1] org.apache.catalina.startu
p.HostConfig.deployDirectory Deploying web application directory C:\Program File
s\Cluster\Node1\apache-tomcat-8.0.21\webapps\manager
28-Apr-2015 11:27:23.477 INFO [localhost-startStop-1] org.apache.catalina.startu
p.HostConfig.deployDirectory Deployment of web application directory C:\Program
Files\Cluster\Node1\apache-tomcat-8.0.21\webapps\manager has finished in 86 ms
28-Apr-2015 11:27:23.478 INFO [localhost-startStop-1] org.apache.catalina.startu
p.HostConfig.deployDirectory Deploying web application directory C:\Program File
s\Cluster\Node1\apache-tomcat-8.0.21\webapps\ROOT
28-Apr-2015 11:27:23.529 INFO [localhost-startStop-1] org.apache.catalina.startu
p.HostConfig.deployDirectory Deployment of web application directory C:\Program
Files\Cluster\Node1\apache-tomcat-8.0.21\webapps\ROOT has finished in 50 ms
28-Apr-2015 11:27:23.529 INFO [main] org.apache.catalina.ha.session.JvmRouteBind
erValve.startInternal JvmRouteBinderValve started
28-Apr-2015 11:27:23.537 INFO [main] org.apache.coyote.AbstractProtocol.start St
arting ProtocolHandler ["http-nio-8080"]
28-Apr-2015 11:27:23.570 INFO [main] org.apache.coyote.AbstractProtocol.start St
arting ProtocolHandler ["ajp-nio-8009"]
28-Apr-2015 11:27:23.573 INFO [main] org.apache.catalina.startup.Catalina.start
Server startup in 7736 ms
28-Apr-2015 11:27:39.685 INFO [Tribes-Task-Receiver-1] org.apache.catalina.tribe
s.io.BufferPool.getBufferPool Created a buffer pool with max size:104857600 byte
s of type: org.apache.catalina.tribes.io.BufferPool15Impl
28-Apr-2015 11:27:40.197 INFO [Membership-MemberAdded.] org.apache.catalina.ha.t
cp.SimpleTcpCluster.memberAdded Replication member added:org.apache.catalina.tri
bes.membership.MemberImpl[tcp://{10, 192, 0, 163}:4001,{10, 192, 0, 163},4001, a
live=1028, securePort=-1, UDP Port=-1, id={82 84 91 43 -87 -56 70 94 -118 -85 3
9 -6 -61 45 43 }, payload={}, command={}, domain={}, ]
```

```
p.HostConfig.deployDirectory Deploying web application directory C:\Program File
s\Cluster\Node2\apache-tomcat-8.0.21\webapps\examples
28-Apr-2015 11:27:44.854 INFO [localhost-startStop-1] org.apache.catalina.startu
p.HostConfig.deployDirectory Deployment of web application directory C:\Program
Files\Cluster\Node2\apache-tomcat-8.0.21\webapps\examples has finished in 1,437
ms
28-Apr-2015 11:27:44.855 INFO [localhost-startStop-1] org.apache.catalina.startu
p.HostConfig.deployDirectory Deploying web application directory C:\Program File
s\Cluster\Node2\apache-tomcat-8.0.21\webapps\host-manager
28-Apr-2015 11:27:44.964 INFO [localhost-startStop-1] org.apache.catalina.startu
p.HostConfig.deployDirectory Deployment of web application directory C:\Program
Files\Cluster\Node2\apache-tomcat-8.0.21\webapps\host-manager has finished in 11
0 ms
28-Apr-2015 11:27:44.965 INFO [localhost-startStop-1] org.apache.catalina.startu
p.HostConfig.deployDirectory Deploying web application directory C:\Program File
s\Cluster\Node2\apache-tomcat-8.0.21\webapps\manager
28-Apr-2015 11:27:45.033 INFO [localhost-startStop-1] org.apache.catalina.startu
p.HostConfig.deployDirectory Deployment of web application directory C:\Program
Files\Cluster\Node2\apache-tomcat-8.0.21\webapps\manager has finished in 68 ms
28-Apr-2015 11:27:45.035 INFO [localhost-startStop-1] org.apache.catalina.startu
p.HostConfig.deployDirectory Deploying web application directory C:\Program File
s\Cluster\Node2\apache-tomcat-8.0.21\webapps\ROOT
28-Apr-2015 11:27:45.084 INFO [localhost-startStop-1] org.apache.catalina.startu
p.HostConfig.deployDirectory Deployment of web application directory C:\Program
Files\Cluster\Node2\apache-tomcat-8.0.21\webapps\ROOT has finished in 49 ms
28-Apr-2015 11:27:45.085 INFO [main] org.apache.catalina.ha.session.JvmRouteBind
erValve.startInternal JvmRouteBinderValve started
28-Apr-2015 11:27:45.092 INFO [main] org.apache.coyote.AbstractProtocol.start St
arting ProtocolHandler ["http-nio-8180"]
28-Apr-2015 11:27:45.105 INFO [main] org.apache.coyote.AbstractProtocol.start St
arting ProtocolHandler ["ajp-nio-8109"]
28-Apr-2015 11:27:45.108 INFO [main] org.apache.catalina.startup.Catalina.start
Server startup in 6106 ms
```

Fig 5.1 Tomcat cluster nodes - Node 1 (left) and Node 2 (right)

Above, we can see the two Tomcat nodes being in active state.
Node 1 (left) is the first node started, hence it adds the subsequently started node as a replication member, creates a buffer pool, and connects to its ports for session data transmission to achieve replication.
Node 2 (right) starts up and waits for the first node to crash/stop in order to take control over the session.

5.4 NAME-BASED VIRTUAL HOSTING

Our web application is currently deployed on our local server cluster, and for it to be accessed from remote clients, the server machines and the client machines should be connected to the same local area network. In order for the application to support any local network, we have created a name-based virtual host. Once the machines are connected to the same network, and when the server IP's host has been configured on the client machines, the web application can be accessed using the URL http://www.examination.com/PASS

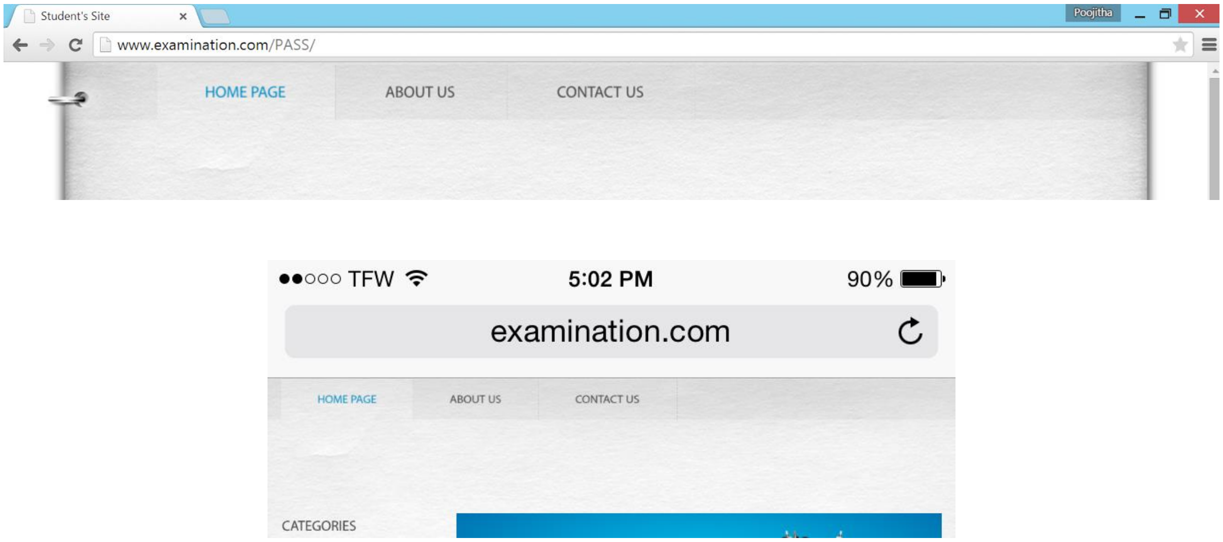


Fig 5.2 Name-Based Virtual Host URL

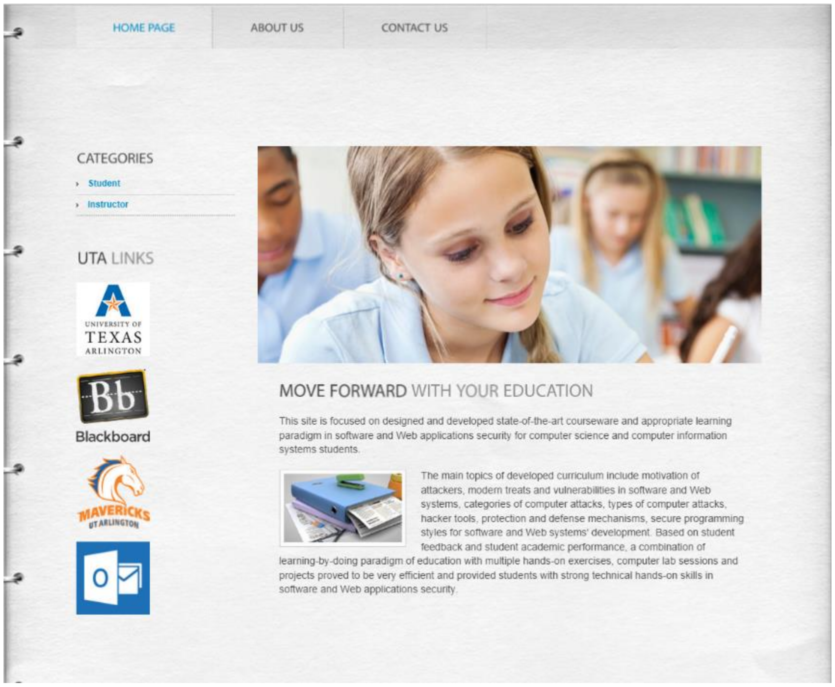
The above images show the system being accessed by multiple, different types of devices by using the same URL at the same time.

5.5 CLIENT-SIDE PROGRAMMING

HTML (Hypertext Markup Language) is used to create web pages. HTML describes the structure of the content on web pages.
CSS is used to provide design to these web pages. This helps in separating content from presentation i.e., design.
JavaScript is used for creating dynamic content, modification and handling content logic on web pages. It provides functionality to a website.
The UI is divided into 4 modules:

1. Login: Authentication and authorization page for the students and Instructors.
2. Instructors: Pages to add/ delete topics, questions, view answers
3. Students: Pages to view topics, take exams. Examination pages contain a timer that will start automatically.
4. Results: Displayed after completion of the exam.

Below are some of the main pages in our system:

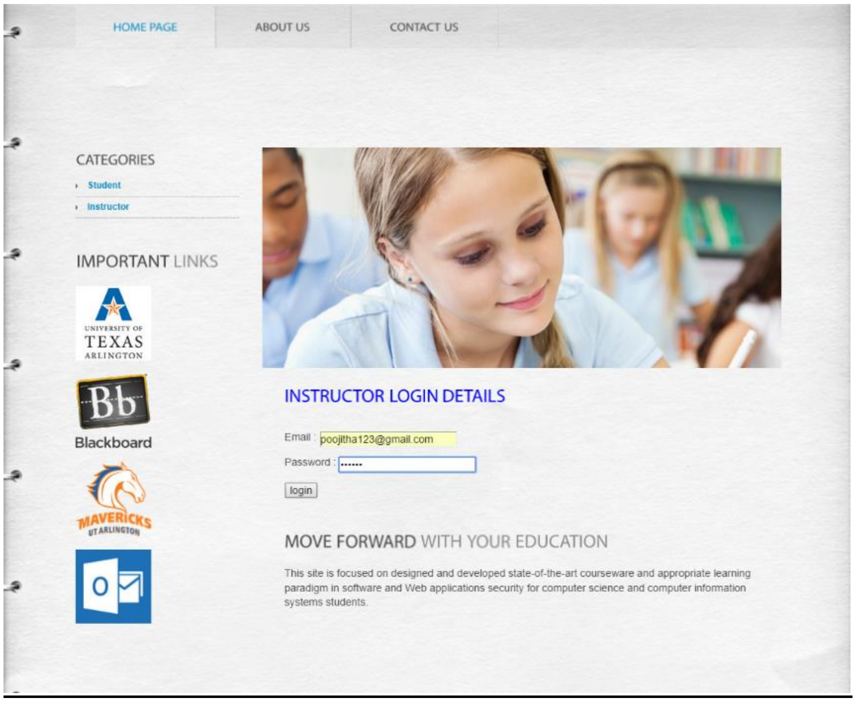


MOVE FORWARD WITH YOUR EDUCATION

This site is focused on designed and developed state-of-the-art courseware and appropriate learning paradigm in software and Web applications security for computer science and computer information systems students.

The main topics of developed curriculum include motivation of attackers, modern treats and vulnerabilities in software and Web systems, categories of computer attacks, types of computer attacks, hacker tools, protection and defense mechanisms, secure programming styles for software and Web systems' development. Based on student feedback and student academic performance, a combination of learning-by-doing paradigm of education with multiple hands-on exercises, computer lab sessions and projects proved to be very efficient and provided students with strong technical hands-on skills in software and Web applications security.

PASS Online Examination System Home Page



INSTRUCTOR LOGIN DETAILS

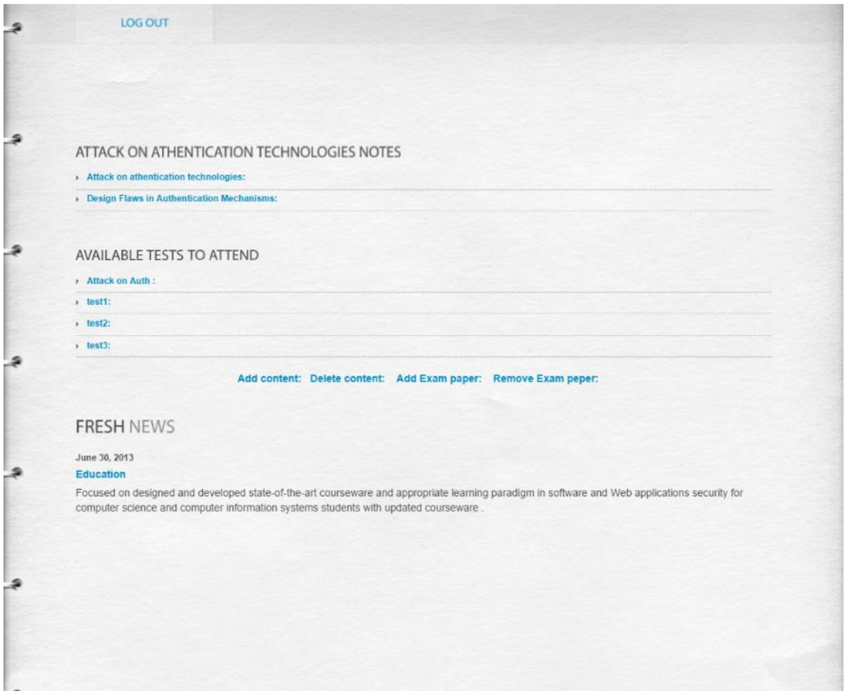
Email : poojitha123@gmail.com

Password : ••••••

MOVE FORWARD WITH YOUR EDUCATION

This site is focused on designed and developed state-of-the-art courseware and appropriate learning paradigm in software and Web applications security for computer science and computer information systems students.

Instructor Login Page



ATTACK ON ATHENTICATION TECHNOLOGIES NOTES

- Attack on athentication technologies:
- Design Flaws in Authentication Mechanisms:

AVAILABLE TESTS TO ATTEND

- Attack on Auth :
- test1:
- test2:
- test3:

Add content: Delete content: Add Exam paper: Remove Exam peper:

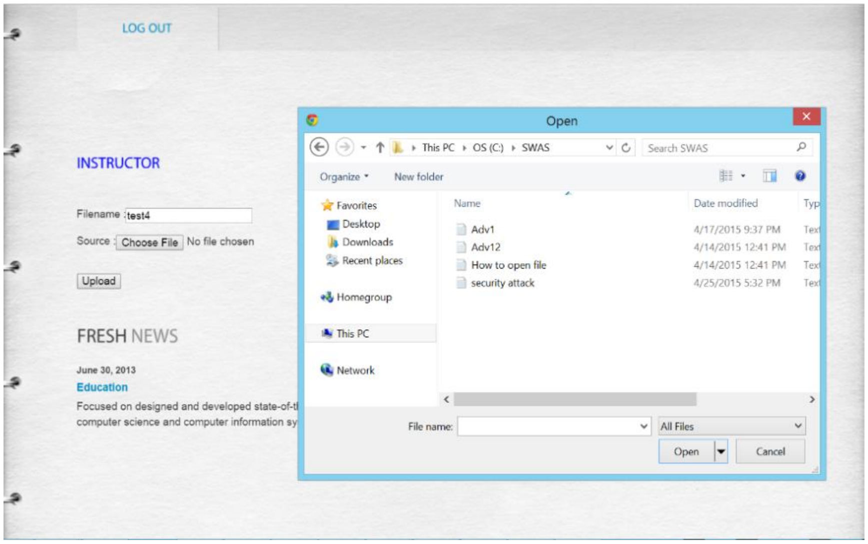
FRESH NEWS

June 30, 2013

Education

Focused on designed and developed state-of-the-art courseware and appropriate learning paradigm in software and Web applications security for computer science and computer information systems students with updated courseware .

<u>Instructor Add/Remove Content Page</u>



INSTRUCTOR

Filename : test4

Source : Choose File No file chosen

Upload

FRESH NEWS

June 30, 2013

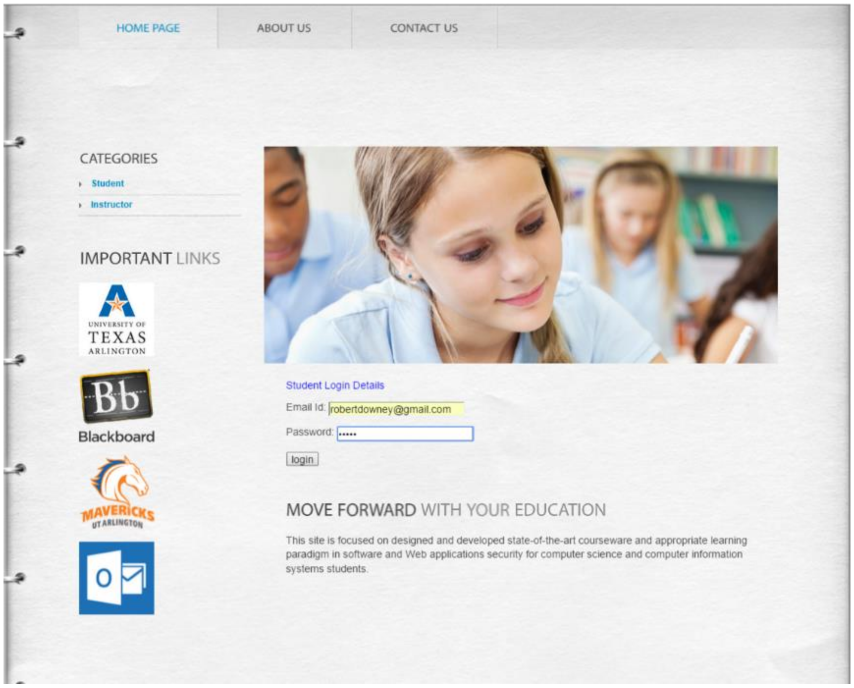
Education

Focused on designed and developed state-of-t
computer science and computer information sy

Name	Date modified	Typ
Adv1	4/17/2015 9:37 PM	Tex
Adv12	4/14/2015 12:41 PM	Tex
How to open file	4/14/2015 12:41 PM	Tex
security attack	4/25/2015 5:32 PM	Tex

File name: All Files

<u>Instructor Upload page</u>



CATEGORIES

- Student
- Instructor

IMPORTANT LINKS

Student Login Details

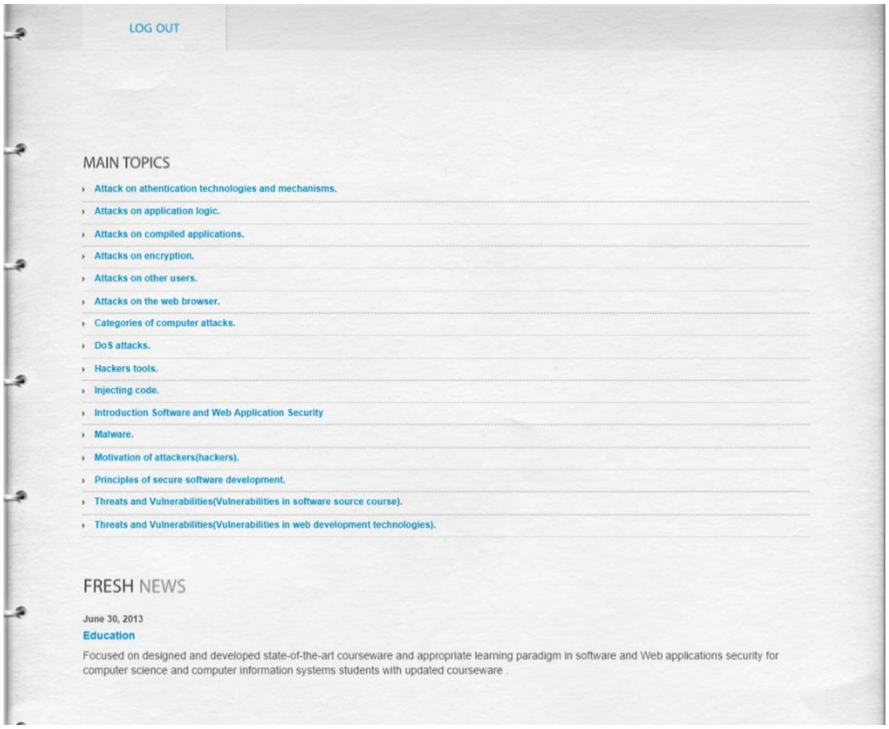
Email Id: robertdowney@gmail.com

Password: •••••

MOVE FORWARD WITH YOUR EDUCATION

This site is focused on designed and developed state-of-the-art courseware and appropriate learning paradigm in software and Web applications security for computer science and computer information systems students.

Student Login Page



MAIN TOPICS

- Attack on athentication technologies and mechanisms.
- Attacks on application logic.
- Attacks on compiled applications.
- Attacks on encryption.
- Attacks on other users.
- Attacks on the web browser.
- Categories of computer attacks.
- DoS attacks.
- Hackers tools.
- Injecting code.
- Introduction Software and Web Application Security
- Malware.
- Motivation of attackers(hackers).
- Principles of secure software development.
- Threats and Vulnerabilities(Vulnerabilities in software source course).
- Threats and Vulnerabilities(Vulnerabilities in web development technologies).

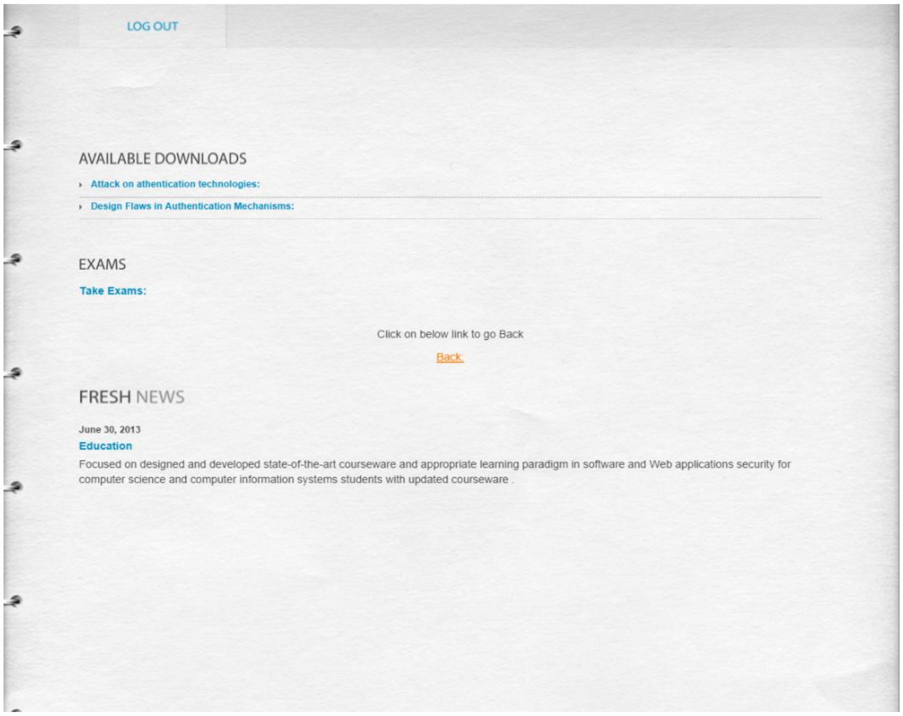
FRESH NEWS

June 30, 2013

Education

Focused on designed and developed state-of-the-art courseware and appropriate learning paradigm in software and Web applications security for computer science and computer information systems students with updated courseware .

Student Main Topics Page



AVAILABLE DOWNLOADS

- Attack on athentication technologies:
- Design Flaws in Authentication Mechanisms:

EXAMS

Take Exams:

Click on below link to go Back

Back.

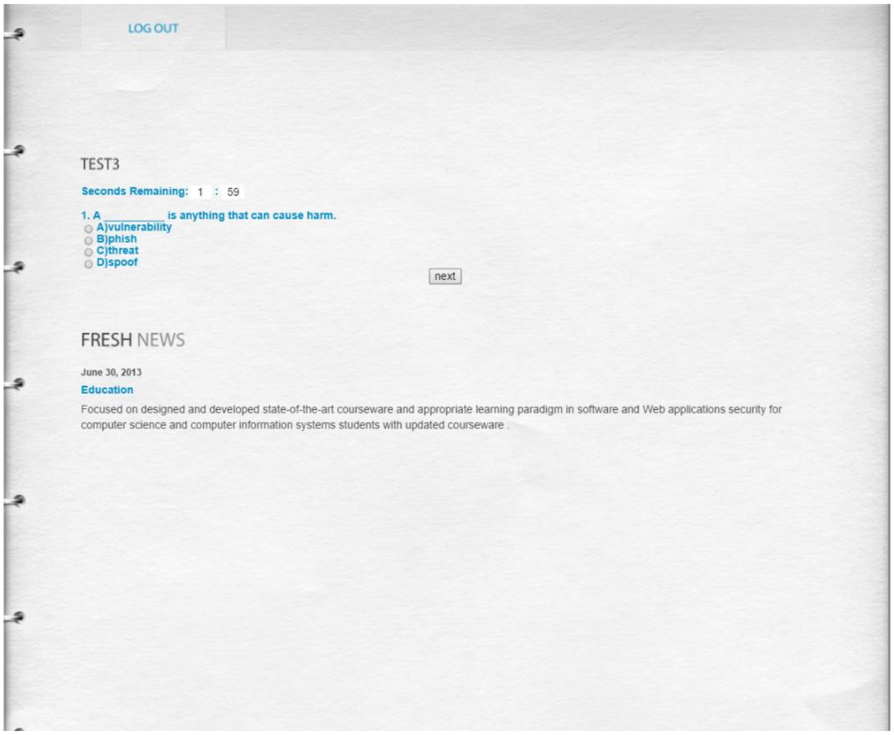
FRESH NEWS

June 30, 2013

Education

Focused on designed and developed state-of-the-art courseware and appropriate learning paradigm in software and Web applications security for computer science and computer information systems students with updated courseware .

Student Downloadable Content, and Available Tests Page



TEST3

Seconds Remaining: 1 : 59

1. A __________ is anything that can cause harm.

○ A)vulnerability
○ B)phish
○ C)threat
○ D)spoof

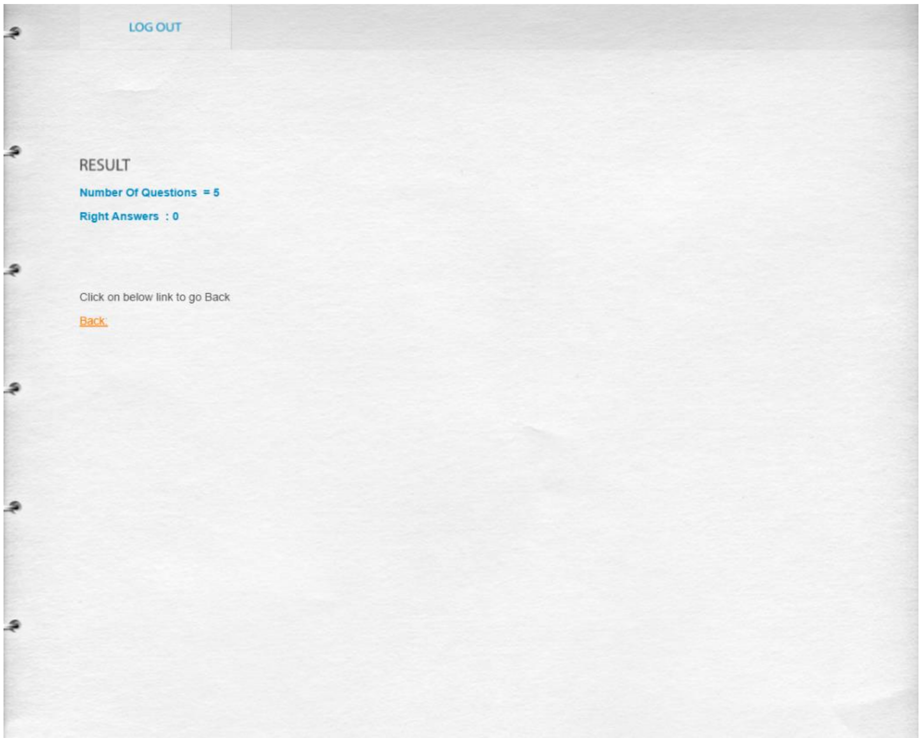
FRESH NEWS

June 30, 2013

Education

Focused on designed and developed state-of-the-art courseware and appropriate learning paradigm in software and Web applications security for computer science and computer information systems students with updated courseware .

Examination Page along with Timer



RESULT

Number Of Questions = 5

Right Answers : 0

Click on below link to go Back

Back:

Results Page

5.6 WORK DIVISION

The entire project implementation was split into three broad sections:

1. User Interface Implementation: This section consists of all the client side web pages and interfaces. This will be implemented by Archana Mohan.
2. Database Implementation: Consists of all the databases used by the system and the DBMS to manage them. This section will be implemented by Srija Reddy Patel
3. Server Implementation: Consists of the entire server side implementation, including integration with the User interface and Databases. Also consists of server clustering, session replication, and remote communication. This being the large section of the project will be implemented by two of us: Poojitha Karlapalem, and Uma Naga Sai Jyothi Kotamraju.

6. THEORETICAL / SIMULATION STUDY

6.1 COMMUNICATION OVERHEAD

Our project can be scaled effectively for the whole class (40 students). At any given time, the server is designed to handle requests from all the students. The service is so seamless that the difference is not noticed by the client machines.

- Time taken to query the results from the DB
- The server can collect the answers from the clients to store it at the database at regular intervals. The results from the intermediate questions are stored at the client side. One advantage of this arrangement is that the load on the server is reduced. For example, every student takes the test at their own pace. So there will be a difference of the start time among the users. If the result is collected from the client after every 5 questions answered, the server will not be choked by the incoming requests.
- The server handles the incoming threads by spawning threads. Memory-aware Task Scheduling on bus-based multiprocessor systems.

6.2 STORAGE OVERHEAD

We estimate that a storage of 5 TB is required for the entire application. This should easily accommodate the student information table, faculty information table, the questions encountered in the tests (could include videos), the result tables, the history table (containing the details of all the exams taken by the student).

6.3 COMPUTATION OVERHEAD

- Optimum querying of the database(the query should return only relevant data so that the resource utilization is minimum)
- Displaying of the graphs should only be done when requested by the student. This prevents unnecessary load on the server.
- Time taken for the results to be displayed on the web page should be as real-time as possible.

7. FUTURE WORK

- Multiple faculty clients can administer the exam simultaneously.
- Logically set questions: The next set of questions given can be changed based on the performance of the student in the previous section.

- Graphical representation of the results to faculty for analysis.
- Display all the answers to the student along with the key answers.

8. REFERENCES

1. Stack Overflow: www.stackoverflow.com
2. Apache Software Foundation: www.apache.org
3. Wikipedia: www.wikipedia.com
4. Exam Software Test Engine: www.exam-software.com
5. Apache Tomcat Clustering: http://www.richardnichols.net
6. Local Network Hotspot www.connectify.me