TEST ENGINE SYSTEM

(TES)

SOFTWARE REQUIREMENTS SPECIFICATION
PHASE I

March 18, 2003

Version 1.0

Team Members

Test Engine System
Software Requirements Specification

Table of Contents

1.0 Introduction

The ‘test engine’ program provides test administrators an easy way to create exams, give exams, and monitor exam-taker responses. The ‘test engine’ will consist of an administration ‘view’ and a user ‘view’. The administrator of the system will be the professor administering the exam. The user of the system will be the exam-taker or the student taking the exam. The exam-taker will scroll through each exam question of the exam he is taking and provide a response to the question before moving on to the next question.
Exam questions will be entered by the exam administrator prior to the exam time. Each exam will be ‘activated’ by an exam administrator. Once the exam is ‘activated’, the exam-takers will be able to begin the exam and post answers to exam questions. The ‘test engine’ program will be delivered as an intranet solution. Use of the program is envisioned to take place in one classroom at one given instance of time. Hence, if two exams are going on at the same time by two different professors, each classroom will have its own URL / intranet website to view the exam and enter responses to exam questions.

1.1 Goals and objectives

The goal of the system is to provide test administrators an easy way to create exams, give exams and monitor exam-taker responses. The system will also provide a uniform interface to exam-takers and an ability to immediately display exam results to the exam taker and compile exam reports to the system administrator.

1.2 Statement of scope

The presentation layer will be divided into two parts: administration and user. The ‘logic engine’ will consist of a library to process calculations required by the system administrator based on student responses to questions during the exam. The ‘logic engine’ will reside on the same server as the web-server. The data-store will retain all data related to the exam.
Inputs into the system will be by both the administrator and by the exam taker of the system. The administrator will enter an exam into the system. Exam type will be specified. Example exam types include ‘real exam’ and ‘practice exam’. The administrator will enter an exam type. Example exam question types could be ‘multiple choice’ or ‘true / false’. Based on the question type, the administrator of the system is prompted for an exam question and possible answers. The amount of answer choices the administrator enters will be dependent on the exam question type selected.
Another input into the system is answer responses entered by the user taking the exam. The user is prompted for the exam type to take. Example exam types include ‘real exam’ and ‘practice exam’. The exam-taker is displayed an exam question and prompted for an answer to the exam question. Upon answering all questions for the exam, the user comes to the last question in the exam. Upon clicking finish, the exam-taker sends all results of his exam to a persistent data store.
The administrator can also enter usernames / passwords into the system. The username / password will be used to authenticate the user in the logon process.
The administrator will also have the ability to view reports based on the exam responses. Sample reports include: List of current students for an administrators’ classroom, exam results per student (group all student exam grades together) and student response report (showing a detailed response per student).

1.3 Software context

Currently, exams are offered by paper method. The administrator of the exam grades the exams and the next time the administrator meets with the test-takers, he gives the results. The administrator may also post exam results to a web page or some other interface easily accessed by large amounts of exam-takers. The ‘test engine’ seeks to fit within the existing process by automating the process and reducing duplicate processes performed by the administrator.
The ‘test engine’ fits into the overall process of the classroom test-taking process. The ‘test engine’ will attempt to automate the creation of exams and automate the evaluation of exam results. The ‘test engine’ will also provide the test administrator to re-use existing exam questions on future exams thus creating a ‘question pool’ from which to choose questions from.

1.4 Major constraints

A constraint of the system is that each user will need to have access to the system at the time specified. Ex: each user will need to have access to the system during the classroom time when the administrator is giving the exam. This may or may not be possible with current university computing constraints.
Another constraint of the system is that the ‘test-engine’ does not currently have a timer giving a certain time to the system. Hence, the administrator of the system will have to ‘activate’ an exam at the beginning of an exam and then make sure everyone is finished with the exam through an interface.

2.0 Usage Scenario

2.1 User profiles

The system should be simple enough to be used by people with elementary browsing skills.
The people who will use the system are: students and the professor. The system should permit students to take practice or real graded tests and also to view their test results. The professor should be able to create tests, activate them, evaluate questions and also view various reports.

The information entered in the system will be:

Students

Login name and password.
Test answers.

Professor

Login name and password.
Test questions.
Test answers.
Student login names and their passwords.

The recipients of the information produced by the system will be:

Students
View and take practice tests.
Take real exam.
View exam results.
View question answers after finishing the test.
Professor
View various reports such as class list, class grades, test questions and their respective answers.

Actors in the "Test engine" system will be:

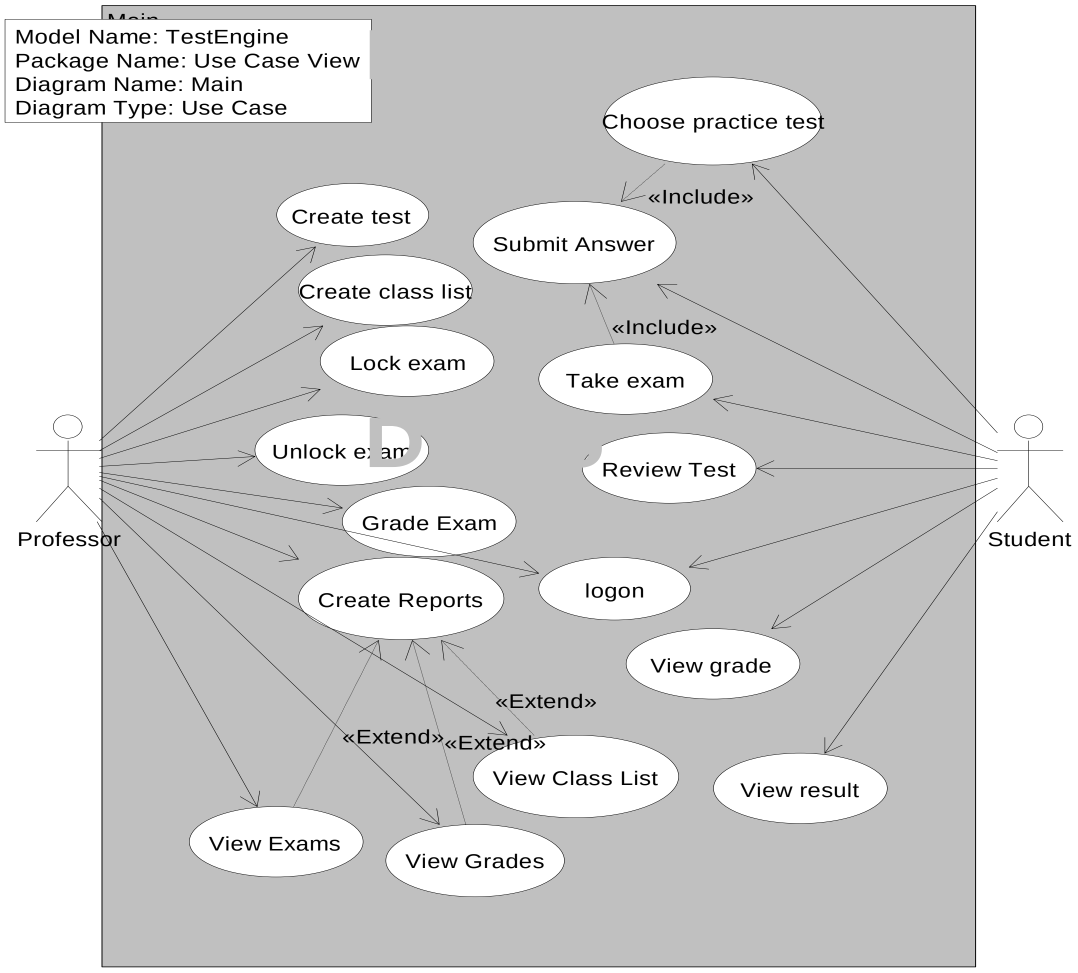
Professor: (System Administrator – privileged user)
Students: (Limited user)

Simple use case description:

Actor	System
The student/professor enters the login name and password	The system validates the user name and password (Basic authentication)
The student chooses a practice test	The system shows the first test question.
The student chooses to take the exam	The system checks whether the exam is unlocked and if yes shows the logon screen.
The student submits a response to a question	The system records test questions and answers in the system database.
The professor creates class list by entering user name and password for the students.	The system updates the class list records
The professor creates test	The system records test questions and answers in the system database.

The professor locks the exam	Students won’t be permitted to take a real exam, but they can still take practice tests.
The professor unlocks the exam	The system permits the students to take the exam.
The professor grades the exam	The system records the student grade

2.2 Use Case Diagram



2.3 Scenarios

Student 1:

Student types chooses to take a practice test
First test question shows up
Student answers the question
The following question show up until the end.
Student views the result of the test

Student 2:

Student chooses to take the exam.
The system checks: is the test locked? The test is locked
The student is notified that he can’t take the test.

Student 3

Student chooses to take the exam.
The system checks: is the test locked? The test is unlocked
The logon screen shows up
Student types the correct user name and password
The student proceeds with answering the questions.
The exam finishes and the result shows up.

Student 4

Student chooses to view his grade
The logon screen shows up
Student types the correct user name and password
The student grades show up

3.0 Data Model: Entity Relationship Diagram

3.1 Data Description

Major Data Objects

STUDENT:

STUDENT consist of the following attributes:

STUDENTID(Primary Key)- Student ID.
SFNAME- Student First Name.
SLNAME- Student Last Name.
SPWD- Student Password 6 characters long.

STUDENTRESP:

STUDENTRESP consist of the following attributes:

STUDENTID- Unique Student ID.
ANSID- Each answer has unique ID.
QUESTIONID- Question number.
EXAMID- Each exams has unique ID .

EXAM:

EXAM consist of the following attributes:

EXAMID- Unique exams ID.
EXNAME- Name of the exams.
EXTYPEID- Exams type ID (Practice or real exam)
ACTIVEDATE- Date of exams.

EXTYPE:

EXTYPE consist of the following attributes:

EXTYPEID- Exams type ID
TYPEDESCRIPTION- Practice or real exam

QUESTIONS:

QUESTIONS consists of the following attributes:

EXAMID- Exams ID
QUESTIONID- Number of question.

QTEXT-Question details
QTYPEID- Question type ID (True/false, Multiple choice or others type)

QUESTYPE:

QUESTYPE consist of the following attributes:

QTYPEID- Question type ID
TYPEDESCRIPTION- True/false, Multiple choice or others types.

ANSWERCHOICES:

ANSWERCHOICES consist of the following attributes:

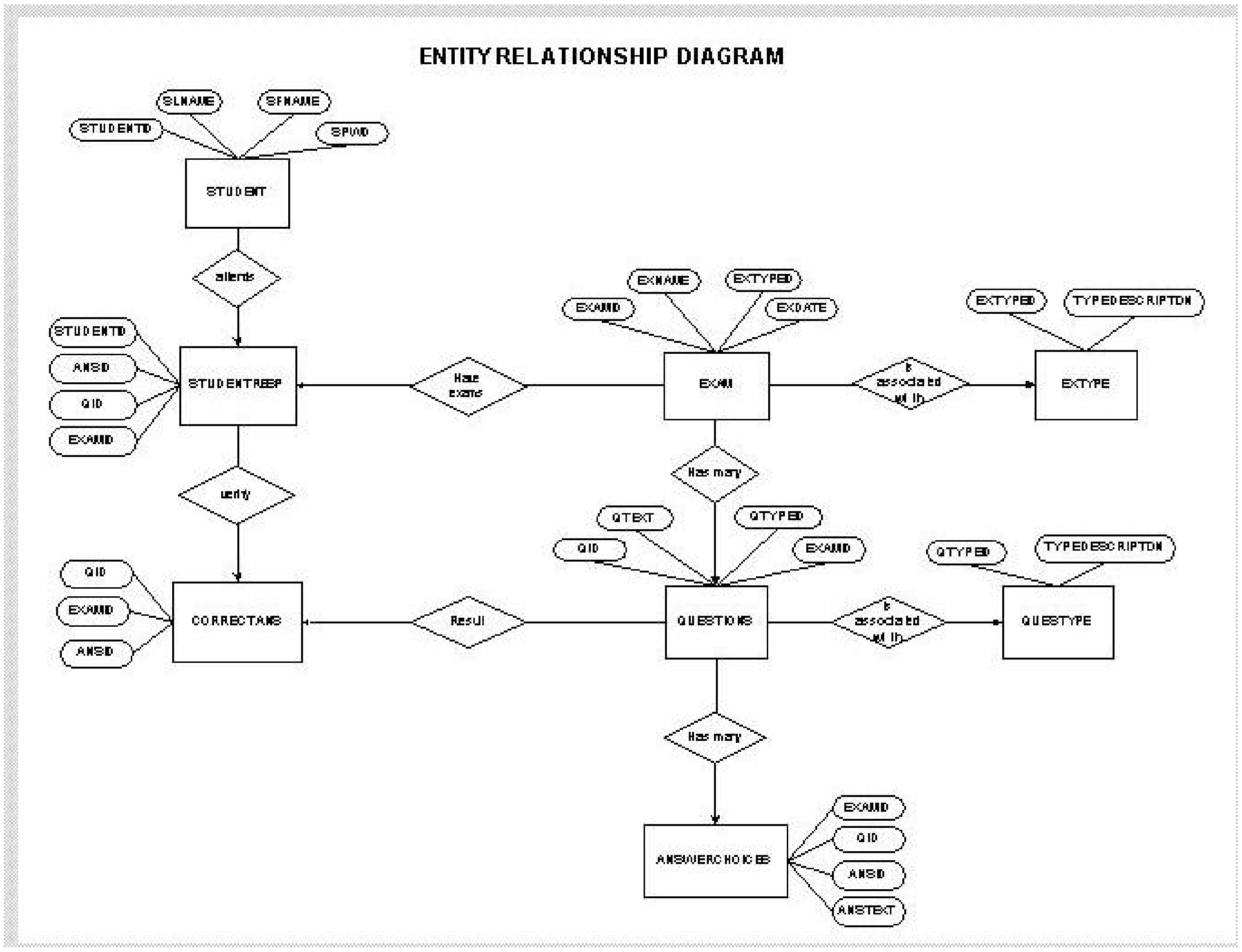
EXAMID- Exams ID
QUESTIONID- Question number
ANSID- Unique Answer ID
ANSTEXT- Answer description.

CORRECTANS:

CORRECTANS consist of the following attributes:

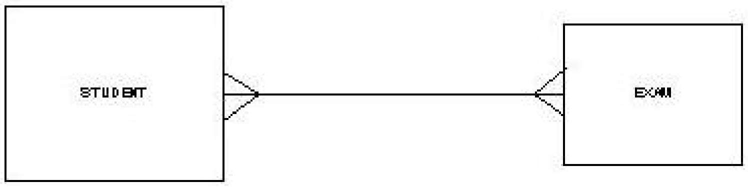
EXAMID- Exams ID
QUESTIONID- Question number
ANSID- Unique Answer ID

3.2 Entity Relationship diagram

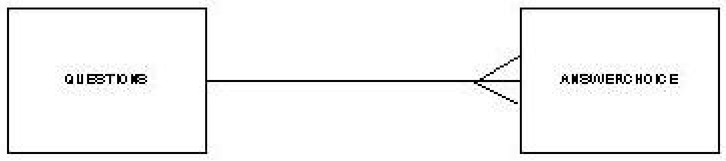


3.3 Justification for one-to-many, many-to-many and one-to one relationship

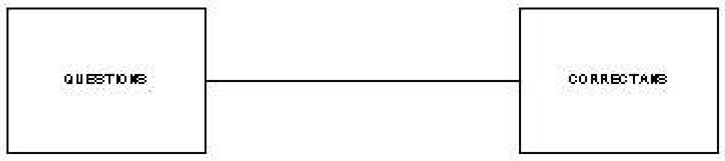
Student and Exam
many-to-many relationship. Each student attends many exams and each exam has many students.



Questions and Answerchoice
one-to-many relationship. Each question has more than one choice and each choice has a one question.



Questions and Correctans
one-to-one relationship. Each question has one correct answer and each correct answer has one question.



3.4 Data Dictionary

STUDENT table

Attribute	Type	Size
STUDENTID	INT	
SLNAME	String	30
SFNAME	String	30
SPWD	String	6

STUDENTCHOICE table

Attribute	Type	Size
STUDENTID	INT	12
QUESTIONID	String	12
ANSID	String	12
EXAMID	String	12

EXAM table

Attribute	Type	Size
EXAMID	String	12
EXTYPEID	String	12

EXNAME	String	40
ACTIVEDATE	Date	-

EXTYPE table

Attribute	Type	Size
EXTYPEID	String	12
EXTYPENAME	String	30

QUESTIONS table

Attribute	Type	Size
QUESTIONID	String	12
QTEXT	String	100
QTYPEID	String	12
EXAMID	String	12

QUESTIONTYPE table

Attribute	Type	Size
QTYPEID	String	12
QTYPENAME	String	30

ANSWERCHOICES table

Attribute	Type	Size
EXAMID	String	12
QUESTIONID	String	12
ANSID	String	12
ANSTEXT	String	30

CORRECTANS table

Attribute	Type	Size
EXAMID	String	12
QUESTIONID	String	12
ANSID	String	12

4.0 System Requirements

4.1 Functional Requirements

Functional requirements describe what functions of the system will work instead of how the functions are implemented. The functional requirements are independent of implementation detail. The functional requirements are categorized within four sections: user interface, function, and data. Requirements are not duplicated across sections; however, one requirement may have a close relationship to a requirement listed in a separate section.

4.1.1 User Interface

This section describes functional capabilities related to screen design and navigation as well as responses to user initiated events. The Test Engine interface will be intuitive and easy-to-use. It will be designed to take advantage of existing user-interface guidelines which will facilitate navigation throughout the system.

Designing the user interface (UI)

I-UI-001	The system shall provide a graphical user interface including icons, pull-down menus, buttons, and hyperlinks as appropriate.
I-UI-002	The system should adhere to commonly accepted user interface guidelines on windowing systems. Ex: a button should perform like a button and not like a checkbox.
I-UI-003	The system shall provide a web-based interface.

4.1.2 Functions

This section describes specific functional capabilities to be provided by the system concentrating on the 'what' instead of the 'how'. The test engine system will collect data from professors or system administrator and display this data in the form of exams. The test engine will also collect data from students in the form of exam responses.

Performing data collection (DC)

I-DC-001 The system should allow all data collection to be performed through a web-based interface.

I-DC-002 The system will allow the system administrator to enter exam data by providing a 'wizard-like' process. The wizard-like process will have the following steps:

1. Enter exam name or choose existing exam name from drop-down combo-box.
2. Enter attributes of the exam such as: 'Active Date' and 'Exam Type'

I-DC-003 The system will allow the system administrator to enter exam question data by providing a 'wizard-like' process. The wizard-like process will have the following steps:

1. Enter Exam question and relevant exam question attributes such as. 'question type' from a drop-down list.
2. Dependent on the question type attribute selected, a display a different template.

I-DC-004 The system shall provide electronic data entry templates corresponding to each of the exam types allowed by the system. Exam question templates include:

1. Multiple Choice questions
2. True / False questions

I-DC-005 The system shall provide a web-based interface for students to view exam questions.

I-DC-006 The system shall display exam questions in templated form. The template used to display the exam question is dependent on the exam type entered by the system administrator.

I-DC-007 The system shall provide a web-based interface for students to enter responses to exam questions.

I-DC-008 The system administrator shall be able to enter new users into the system and de-activate current users of the system. The new-user / deactivation functionality shall be accessed through a we-based interface.

I-DC-009 The system administrator shall be able to 'activate' an exam through a we-based interface.

Conducting queries and requesting reports (QP)

I-QP-001 The system shall display reports based on data entered in the test engine system. The reports should be web-based.

I-QP-002 The system shall be able to display exam results based upon student responses. This report will be accessed by both the student and the system administrator.

I-QP-003 The system shall be able to display reports to the system administrator. Example reports include:

1. List of current students entered into the system by the administrator.
2. Display exam results per student. This report would have the following fields displayed: Exam Name as a report header, StudentID as a line item, Student total score as a line item.

4.1.3 Data

This section describes the data requirements necessary to support the functional requirements.

Maintaining database information (DB)

I-DB-001 The system shall maintain a listing of student that participates in the Test program.

I-DB-002 The system shall maintain a listing of Exams. Which is enter by the administrator

I-DB-003 The system shall maintain question data type of question and answer choice.

I-DB-004 The system shall maintain listing of correct answer of each question of each exam.

I-DB-005 The system shall maintain student response.

I-DB-006 The system shall display reports based on data entered in the test engine system.

I-DB-007 The system shall be able to display exam results based upon student responses.

I-DB-008	The system shall be able to display reports to the system administrator. These formats shall include graphs tables, charts.

4.2 Non-Functional Requirements

4.2.1 Operational

This section describes the reliability, availability, and recoverability requirements of the system as well as requirements to support and facilitate use of the system.
To date, no requirements for usability have been identified

4.2.2 Usability

This section addresses issues related to the ease-of-use and effectiveness of the system. To date, no requirements for usability have been identified.

4.2.3 Performance

This section identifies criteria for system response to user initiated events such as the speed of accessing data for ad hoc queries or standard reports.
To date, no requirements for performance have been identified.

Supporting users (SU)

I-SU-001	The system shall be deployed with a formal training program. This training program shall provide instruction on accessing the system, entering and submitting exams, selecting pre-defined reports, and initiating ad hoc queries.
I-SU-002	The system shall provide on-line context sensitive help features for each TESTENGINE function.
I-SU-003	The system shall provide a data entry capability for capturing information on Questions from students.

4.2.4 Security

Controlling user access (UA)

This section addresses the essential security for the system including general standards and guidelines relating to information systems security, physical security, data security, and requirements for controlling user access

I-UA-001 The system shall provide public users with web access to practice tests Exams shall not be available to public users.

I-UA-002 The system shall require a User ID and Password to complete and submit an exam. Only authorized students by the professor shall be permitted to take an exam.

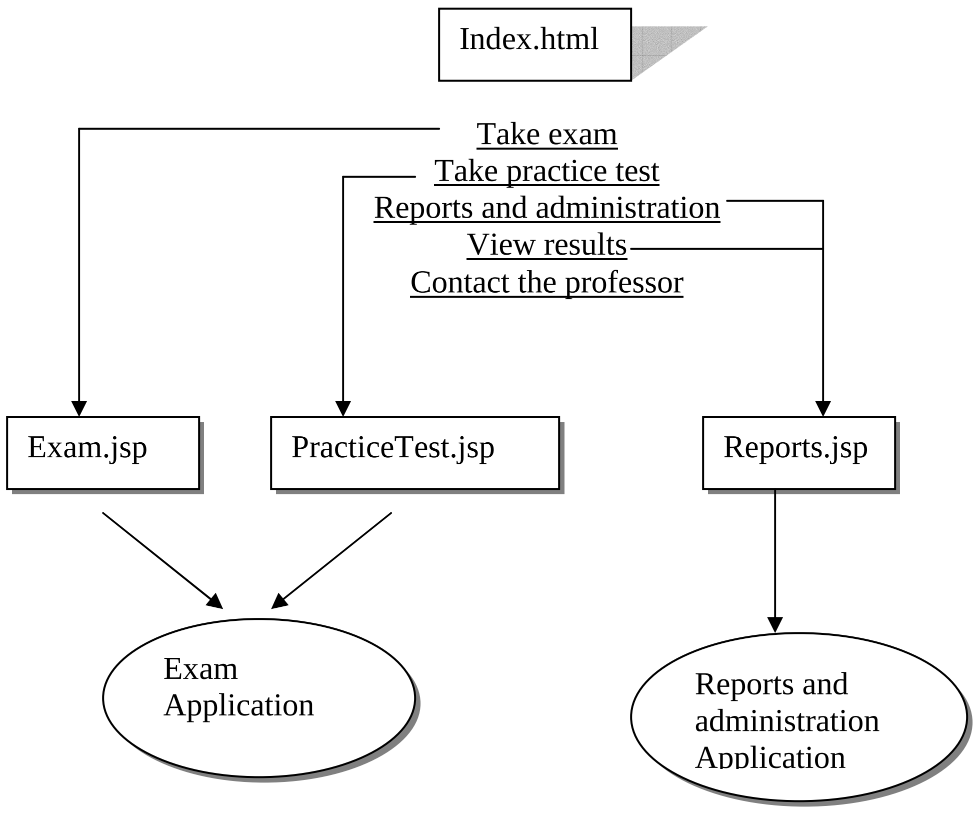
I-UA-003 The system shall require a User ID and Password to create, modify, or delete information in database tables. Only one administrator will have this rights.

I-UA-004 The system shall maintain a table of authorized students containing the valid User ID and Password for each authorized user.

I-UA-005 The system shall check each user login against the table of authorized users before enabling access to exam.

5.0 Behavioral Model and Description

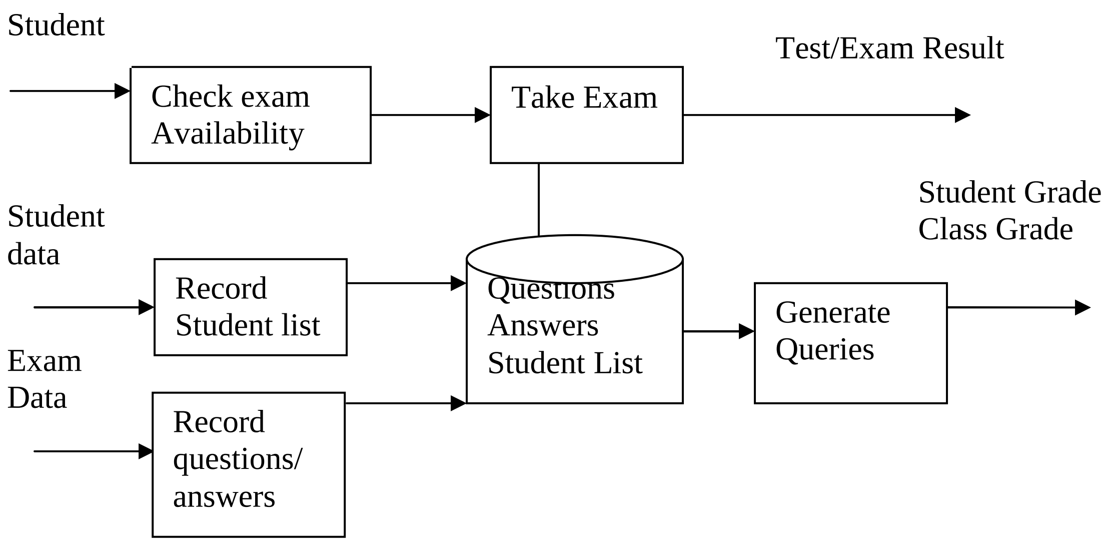
Web Site Structure



Description:

From the index page by clicking to the "Take exam" or "Take practice test" you can start the "Exam application". By clicking the "Reports and administration button" you can start the "Reports and administration application" which makes various reports (grades, class list) and also creates tests.

Data Flow Diagram



6.0 Restrictions, Limitations, and Constraints

Design Constraints

- Must use html, javascript and jsp to develop the project
- Must use apache tomcat web server
- Must use mySQL database server

Limitations

- Timing of the exam will not be done by the software.
- The system administrator will have to manually activate the exams.
- Essay type questions won't be possible to be graded automatically.

Security

7.0 Validation Criteria

7.1 System Acceptance/Testing

This section identifies the conditions that the system must satisfy in order to be accepted by the user, customer, or authorizing entity. The TESTENGINE developer' team will prepare a formal test plan to guide the overall testing process. The plan will identify the activities necessary to test that each requirement has been met in the redesigned system and will include a schedule for the completion of all tests. The testing should encompass three distinct levels: (1) the redesigned software components will be unit tested by the developer individually; (2) integration testing of components will be conducted to ensure that each component works together; and (3) functionality testing will be conducted to ensure that the requirements have been met. As the testing process continues, different students and professors will also be asked to test and evaluate the system. This test and evaluation process will be conducted both informally (limited structure, focus on free play) and formally (structured walk through with written evaluation results). Problems will be corrected as they are identified; suggestions for enhancements (i.e., functionality that exceeds documented requirements) will be assessed for schedule and cost implications and may be deferred to a later development phase or referred to other more appropriate systems for development.