Texas A&M University – Corpus Christi
College of Science and Technology
Department of Computing and Mathematical Sciences

Design and Implementation of
READ, REVIEW & ASSESS SYSTEM

GRADUATE PROJECT
SPRING 2002

BY
ABHIJEET TRIVEDI

COMMITTEE MEMBERS

Dr. Dulal Kar,
Chairperson

Dr. Holly Patterson-McNeill,
Member

Dr. Mario Garcia,
Member
Design and Implementation of
READ, REVIEW & ASSESS SYSTEM

GRADUATE PROJECT
SPRING 2002

BY
ABHIJEET TRIVEDI

COMMITTEE MEMBERS

Dr. Dulal Kar,
Chairperson

Dr. Holly Patterson-McNeill,
Member

Dr. Mario Garcia,
Member
Abstract

This project consists of the design and implementation of an application for automatic assignment submission, review and evaluation. The application includes a set of utilities for administrator to administer the system and a set of tools for instructors and students to submit, read, review and assess assignments. The system is Web-based, with a Graphical User Interface and a database. This system will be used to involve students in peer evaluation and collaborative learning process in the computer science courses taught at Texas A&M University–Corpus Christi.
# Table of Contents

ABSTRACT................................................................................................................. i
LIST OF FIGURES........................................................................................................ v
LIST OF TABLES............................................................................................................ vi
1. BACKGROUND AND RATIONALE................................................................. 1
   1.1 Read, Review & Access System................................................................. 4
   1.2 Rubric............................................................................................................. 5
2. NARRATIVE........................................................................................................... 6
   2.1 Overview......................................................................................................... 6
   2.2 Features of Read, Review & Access System.................................................. 6
   2.3 Description of User Interface........................................................................ 7
   2.4 Common User Interface................................................................................ 8
      2.4.1 Main Page.................................................................................................. 8
      2.4.2 New User Page.......................................................................................... 8
      2.4.3 Login Page............................................................................................... 9
      2.4.4 Forgot Password Page.............................................................................. 9
   2.5 Student Interface........................................................................................... 10
      2.5.1 Student Page........................................................................................... 11
      2.5.2 Student Course Page............................................................................... 11
      2.5.3 Upload Assignment Page.......................................................................... 12
      2.5.4 Assignment Details Page.......................................................................... 13
      2.5.5 Review Assignments Page....................................................................... 14
      2.5.6 List Assignments Page............................................................................ 14
      2.5.7 Evaluation Page........................................................................................ 15
      2.5.8 Check Assignments Page......................................................................... 15
      2.5.9 Checked Assignment Page...................................................................... 16
      2.5.10 Edit Profile Page.................................................................................... 16
   2.6 Instructor Interface......................................................................................... 18
      2.6.1 Instructor Page....................................................................................... 19
      2.6.2 Instructor Course Page............................................................................ 19
      2.6.3 Edit Assignments Page............................................................................ 20
      2.6.4 Assignment Details Page........................................................................ 20
      2.6.5 Edit Rubric Page..................................................................................... 21
      2.6.6 Review Assignments Page....................................................................... 22
      2.6.7 List Assignments Page............................................................................ 22
      2.6.8 Evaluation Page........................................................................................ 23
      2.6.9 Checked Assignment Page...................................................................... 24
      2.6.10 Grade Book............................................................................................ 24
      2.6.11 Edit Profile Page.................................................................................... 24
   2.7 Administrator Interface.................................................................................. 25
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.7.1 Administrator Page</td>
<td>26</td>
</tr>
<tr>
<td>2.7.2 Edit Instructors Page</td>
<td>26</td>
</tr>
<tr>
<td>2.7.3 Instructor Details Page</td>
<td>27</td>
</tr>
<tr>
<td>2.7.4 Edit Courses Page</td>
<td>28</td>
</tr>
<tr>
<td>2.7.5 Course Details Page</td>
<td>28</td>
</tr>
<tr>
<td>2.7.6 Edit Students Page</td>
<td>29</td>
</tr>
<tr>
<td>2.7.7 Student Details Page</td>
<td>30</td>
</tr>
<tr>
<td>2.7.8 Edit Administrators Page</td>
<td>30</td>
</tr>
<tr>
<td>2.7.9 Administrator Details Page</td>
<td>30</td>
</tr>
<tr>
<td>2.7.10 Edit Classes Page</td>
<td>30</td>
</tr>
<tr>
<td>2.7.11 Class Details Page</td>
<td>31</td>
</tr>
<tr>
<td>2.7.12 Delete Class Page</td>
<td>32</td>
</tr>
<tr>
<td>3. PROJECT ENVIRONMENT</td>
<td>33</td>
</tr>
<tr>
<td>4. SYSTEM DESIGN AND IMPLEMENTATION</td>
<td>34</td>
</tr>
<tr>
<td>4.1 System Analysis</td>
<td>34</td>
</tr>
<tr>
<td>4.2 System Architecture and Overview</td>
<td>36</td>
</tr>
<tr>
<td>4.3 Database Design and Implementation</td>
<td>38</td>
</tr>
<tr>
<td>4.3.1 Design and Entity Relationships</td>
<td>38</td>
</tr>
<tr>
<td>4.3.2 Implementation of Tables</td>
<td>40</td>
</tr>
<tr>
<td>4.4 RRAS System Modules</td>
<td>41</td>
</tr>
<tr>
<td>4.4.1 Common Interface Modules</td>
<td>42</td>
</tr>
<tr>
<td>4.4.1.1 New User</td>
<td>42</td>
</tr>
<tr>
<td>4.4.1.2 Login</td>
<td>43</td>
</tr>
<tr>
<td>4.4.1.3 Forgot Password</td>
<td>44</td>
</tr>
<tr>
<td>4.4.2 Student Interface Modules</td>
<td>45</td>
</tr>
<tr>
<td>4.4.2.1 Student Page</td>
<td>46</td>
</tr>
<tr>
<td>4.4.2.2 Student Course Details</td>
<td>46</td>
</tr>
<tr>
<td>4.4.2.3 Upload Assignment</td>
<td>47</td>
</tr>
<tr>
<td>4.4.2.4 Assignment Details</td>
<td>47</td>
</tr>
<tr>
<td>4.4.2.5 Check Assignments</td>
<td>48</td>
</tr>
<tr>
<td>4.4.2.6 Checked Assignment Details</td>
<td>49</td>
</tr>
<tr>
<td>4.4.2.7 Review Assignments</td>
<td>50</td>
</tr>
<tr>
<td>4.4.2.8 List Assignments</td>
<td>51</td>
</tr>
<tr>
<td>4.4.2.9 Evaluation Page</td>
<td>51</td>
</tr>
<tr>
<td>4.4.2.10 Edit Profile</td>
<td>53</td>
</tr>
<tr>
<td>4.4.3 Instructor Interface Modules</td>
<td>54</td>
</tr>
<tr>
<td>4.4.3.1 Instructor Page</td>
<td>55</td>
</tr>
<tr>
<td>4.4.3.2 Instructor Course Details</td>
<td>55</td>
</tr>
<tr>
<td>4.4.3.3 Edit Assignments</td>
<td>56</td>
</tr>
<tr>
<td>4.4.3.4 Assignment Details</td>
<td>57</td>
</tr>
<tr>
<td>4.4.3.5 Edit Rubric</td>
<td>58</td>
</tr>
<tr>
<td>4.4.3.6 Review Assignments</td>
<td>59</td>
</tr>
<tr>
<td>4.4.3.7 List Assignments</td>
<td>59</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
</tr>
<tr>
<td>---------</td>
<td>------</td>
</tr>
<tr>
<td>4.4.3.8 Checked Assignment Details</td>
<td>60</td>
</tr>
<tr>
<td>4.4.3.9 Evaluation Page</td>
<td>60</td>
</tr>
<tr>
<td>4.4.3.10 Grade Book</td>
<td>61</td>
</tr>
<tr>
<td>4.4.3.11 Edit Profile</td>
<td>61</td>
</tr>
<tr>
<td>4.4.4 Administrator Interface Modules</td>
<td>62</td>
</tr>
<tr>
<td>4.4.4.1 Instructor Page</td>
<td>63</td>
</tr>
<tr>
<td>4.4.4.2 Edit Instructors</td>
<td>63</td>
</tr>
<tr>
<td>4.4.4.3 Instructor Details</td>
<td>64</td>
</tr>
<tr>
<td>4.4.4.4 Edit Students</td>
<td>65</td>
</tr>
<tr>
<td>4.4.4.5 Student Details</td>
<td>65</td>
</tr>
<tr>
<td>4.4.4.6 Edit Courses</td>
<td>66</td>
</tr>
<tr>
<td>4.4.4.7 Course Details</td>
<td>66</td>
</tr>
<tr>
<td>4.4.4.8 Edit Classes</td>
<td>68</td>
</tr>
<tr>
<td>4.4.4.9 Class Details</td>
<td>68</td>
</tr>
<tr>
<td>4.4.4.10 Edit Administrators</td>
<td>69</td>
</tr>
<tr>
<td>4.4.4.11 Administrator Details</td>
<td>70</td>
</tr>
<tr>
<td>5. RESULTS</td>
<td>71</td>
</tr>
<tr>
<td>6. SUMMARY AND FUTURE WORK</td>
<td>72</td>
</tr>
<tr>
<td>7. REFERENCES</td>
<td>73</td>
</tr>
<tr>
<td>8. APPENDICES</td>
<td>75</td>
</tr>
<tr>
<td>Appendix A Data Dictionary of RRAS Tables</td>
<td>76</td>
</tr>
<tr>
<td>Appendix B CD ROM</td>
<td></td>
</tr>
</tbody>
</table>
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Common User Interface Map</td>
<td>8</td>
</tr>
<tr>
<td>2</td>
<td>Student Interface Map</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>Student Page</td>
<td>11</td>
</tr>
<tr>
<td>4</td>
<td>Student Course Page</td>
<td>12</td>
</tr>
<tr>
<td>5</td>
<td>Upload Assignment Page</td>
<td>12</td>
</tr>
<tr>
<td>6</td>
<td>Assignment Details Page</td>
<td>13</td>
</tr>
<tr>
<td>7</td>
<td>Review Assignments Page</td>
<td>14</td>
</tr>
<tr>
<td>8</td>
<td>Check Assignments Page</td>
<td>16</td>
</tr>
<tr>
<td>9</td>
<td>Evaluation Page</td>
<td>17</td>
</tr>
<tr>
<td>10</td>
<td>Edit Profile</td>
<td>17</td>
</tr>
<tr>
<td>11</td>
<td>Instructor Interface Map</td>
<td>18</td>
</tr>
<tr>
<td>12</td>
<td>Instructor Course Page</td>
<td>19</td>
</tr>
<tr>
<td>13</td>
<td>Assignment Details Page</td>
<td>21</td>
</tr>
<tr>
<td>14</td>
<td>Edit Rubric Page</td>
<td>22</td>
</tr>
<tr>
<td>15</td>
<td>List Assignments Page</td>
<td>23</td>
</tr>
<tr>
<td>16</td>
<td>Grade Book</td>
<td>24</td>
</tr>
<tr>
<td>17</td>
<td>Administrator Interface Map</td>
<td>25</td>
</tr>
<tr>
<td>18</td>
<td>Administrator Page</td>
<td>26</td>
</tr>
<tr>
<td>19</td>
<td>Edit Instructors Page</td>
<td>27</td>
</tr>
<tr>
<td>20</td>
<td>Instructor Details Page</td>
<td>28</td>
</tr>
<tr>
<td>21</td>
<td>Course Details Page</td>
<td>29</td>
</tr>
<tr>
<td>22</td>
<td>Edit Classes Page</td>
<td>31</td>
</tr>
<tr>
<td>23</td>
<td>Class Details</td>
<td>32</td>
</tr>
<tr>
<td>24</td>
<td>System Architecture</td>
<td>37</td>
</tr>
<tr>
<td>25</td>
<td>Entity Relationship Diagram</td>
<td>39</td>
</tr>
</tbody>
</table>
# List of Tables

1. Example Grading rubric for Programming Projects ..................  5
2. RRAS Database Tables .............................................  38
3. Common Interface Modules to Database Tables Matrix ............  42
4. Student Interface Modules to Database Tables Matrix ...........  45
5. Instructor Interface Modules to Database Tables Matrix .......  54
6. Administrator Interface Modules to Database Tables Matrix ....  62
1. Background and Rationale

Computer science courses, that have high practical programming component and are assessed by programming assignments, often lack enough feedback to students and do not provide an environment for collaborative involvement of the students with their peers. Consequently, students receive a limited perspective in problem solving technique. Instructors on the other hand face a stark choice between investing large amount of time and effort in retrieving student assignments, running and testing them manually or simply scanning a program and assigning a grade. Thus arises a need to develop a system, which would provide an environment for better course management and delivery with special utilities to implement the process of peer evaluation.

The use of the World Wide Web is becoming increasingly popular and effective medium for delivery of instruction [1]. The use of the Web in course delivery ranges from total course delivery to varying levels of support for course delivery. The Web is also seeing increasing use as an educational, administrative, and management tool [2, 3]. Institutions are providing on-line admission, course registration, tuition payment, and other administrative tasks. Students may also submit assignments and do grade queries and instructors can manage assignments, check class enrollments, submit grades, and monitor student progress over Internet [4].

Several Web environments have been developed for automated collection of coursework, effective electronic marking or online assessment, and automatic grading [5]. Blackboard
[6] provides a course management system a support for, customizable institution-wide portals, online communities, and an advanced architecture that allows a Web-based integration of multiple administrative systems. ClassNet [7] provides a set of tools for managing Internet-based instructional activities such as course creation, enrollment control, test creation and assignments, student submission and automatic grading, discussion forums, chat rooms and e-mail exchange between students and instructors. Ecourse [8] provides educators with a set of tools for managing and delivering online courses over the Internet. These tools include course management, outlining assignments, delivering lectures, testing, grading, and interaction among students. WebCT [9] is a tool that facilitates creation of a sophisticated World Wide Web-based educational environment. It provides an interface for designing the course presentation, educational tools to facilitate learning, communication, and collaboration. It also provides a set of administrative tools to assist the instructor in managing and improving the course. All these systems are aimed to provide a user-friendly interface for students to submit all forms of electronic course work, and an equally user-friendly interface for tutors to grade the coursework and give immediate feedback.

Collaborative learning is defined as "a learning process that emphasizes group or cooperative efforts among faculty and students. It stresses active participation and interaction on the part of both students and instructors." [10], Collaborative learning using networked computers or Internet is hardly a new topic. The importance and relevance of interaction to an effective learning process has been stressed by many theorists [11], and also by many advocates of situated learning [12]. Uses of interactive
Web tools like chat rooms, bulletin boards are also common for providing a collaborative learning environment that helps student share ideas among peers.

Then the question is why did we need another type of environment? Because we needed a system which could be used to involve students in a collaborative learning process using peer evaluation as a mode of grading coursework. This system incorporates utilities available in standard course handling systems as mentioned above. Developing techniques for automatic assignment submission and peer evaluation is an attempt to push teaching beyond the all too common "lecture-and-test" method. It not only relieves the instructor of an increasing workload but also improves the learning experience available to the students by providing them with facilities for immediate and effective feedback. The purpose of the peer evaluation is two folds. First, to help the authors of the evaluated piece make that piece and future pieces better, helping them understand what they are doing right and what they might be doing wrong. Second, to give students practice in making critical judgments and in developing an analytical skill in them. Such an environment can be further extended to create a full-scale computer-based learning and teaching system including a set of utilities for Web-based classes, online submission and grading, and plagiarism detection.
1.1 Read, Review and Assess System ( RRAS )

The project to create a Web-based assignment submission and assessment system evolved from the idea of providing an environment which uses peer evaluation in assessing students in computer science courses. The system has an automatic assignment submission tool, an interface for students to read, review and assess others assignments, check evaluation on their assignments and tools for instructors to administer the system. This system will be used to improve a student's participation and understanding in introductory programming courses using peer evaluation techniques. The system uses Rubrics as a standard to evaluate others work. This system attains the following objectives for students and teachers.

A. Students:
1) To gain different perspectives from peers towards solving problems
2) To be able to rank oneself in a class and try to improve
3) To learn to analyze other's work
4) To develop confidence and trust among peers for good work done

B. Teachers:
1) To reduce the amount of work and time in collecting and grading assignments
2) To provide better justification of grades assigned to students
3) To achieve improved evaluation of students
4) To provide faster feedback on assignments
1.2 Rubric: A Rubric is an authentic assessment tool, particularly useful in assessing criteria that are complex and subjective [13]. Authentic assessment is geared towards assessment methods, which correspond as closely as possible to real world experience. Table 1 shows a Rubric for Grading Programming Projects. (Example Scale: Very Good–4, Good–3, Normal–2, Bad–1, Very Bad–0)

<table>
<thead>
<tr>
<th>Overall Program</th>
</tr>
</thead>
<tbody>
<tr>
<td>Program Easy to Read and Understand ; Good Use of Indentation and White Space</td>
</tr>
<tr>
<td>Program Divided into Appropriate Functions</td>
</tr>
<tr>
<td>Functions are of Appropriate Length and Complexity</td>
</tr>
<tr>
<td>Accurate use of Repetitive Statements (Loops)</td>
</tr>
<tr>
<td>Accurate use of Conditional Statements</td>
</tr>
<tr>
<td>Accurate use of Descriptive Identifiers</td>
</tr>
<tr>
<td>Variables Properly Initialized</td>
</tr>
<tr>
<td>Proper use of Variables (No Global Variables)</td>
</tr>
<tr>
<td>Efficiency of Algorithm</td>
</tr>
<tr>
<td>Internal Documentation</td>
</tr>
<tr>
<td>Program Begins with Identifying Comments</td>
</tr>
<tr>
<td>Comments Used to Identify Variables, Functions or Self Documentation</td>
</tr>
<tr>
<td><strong>Input and Output</strong></td>
</tr>
<tr>
<td>Good Use of Prompts for User Input</td>
</tr>
<tr>
<td>User Instructions Provided as Needed</td>
</tr>
<tr>
<td>Input and Output neatness, Well Formatted, Not Cluttered</td>
</tr>
<tr>
<td>Correct Output is Obtained</td>
</tr>
<tr>
<td><strong>Executable Code</strong></td>
</tr>
<tr>
<td>Program Runs Without Errors</td>
</tr>
<tr>
<td>Program Meets the Specifications of the Problem</td>
</tr>
<tr>
<td>Program Correctly Solves the Problem</td>
</tr>
<tr>
<td><strong>Testing</strong></td>
</tr>
<tr>
<td>Test Cases Appropriate to the Problem</td>
</tr>
<tr>
<td>A Sufficient Number of Test Cases Considered</td>
</tr>
<tr>
<td><strong>Total</strong></td>
</tr>
</tbody>
</table>

Table 1. Example Grading Rubric for Programming Projects [14]
2. Narrative

2.1 Overview

The Read Review and Assess System (RRAS) automates the process of assignment submission and evaluation among peers and instructors. The system allows students to submit assignments, check grades, and review and evaluate assignments online. The student can evaluate other student's assignments depending on the setting created and the Rubric specified by the instructor. The instructor has a set of tools including grading assignments, reviewing student assignments, maintaining grades and giving feedback to students. The system is equipped with a user-friendly interface on a directory structure to store files, and a database. Providing proper security features and making the system compatible and easy to maintain in various environments was some of the main issues taken into consideration while developing the system. This system is to be used in introductory computer science programming.

2.2 Features of Read, Review and Access System

The application is a Web-based and has features to support the following tasks:

- submit assignments, and evaluate assignments
- check grade and assignments
- create course, edit course, add students, and edit profile
- grade students assignment and maintain grade book
- upload files
2.3 Description of User Interface

There are three types of users associated with the RRAS: – the students, the instructors and the administrator. Therefore, the design of the system is divided into three main interfaces: – a student interface, an instructor interface, and an administrator interface. To access RRAS the users has to go to the system's URL, which takes them to the Main Page having links to the Login Page and the New User Page. When a user accesses the system for the first time, he/she has to go to the New User Page to select a User ID and Password for the system. A confirmation email is sent to the user with a copy of the User ID and Password. The user can then go to the Login Page to access the appropriate interface.
2.4 The Common User Interface

Figure 1, shows the web pages the user sees as he/she navigates through the Student interface.

![Diagram showing the Common User Interface Map]

2.4.1 The Main Page

This page is the first page that the user sees when he/she accesses the system; the page gives a brief idea of the system and has links to the New User Page and the Login Page. If the user is using the system for the first time, he/she has to activate the New User link to register into the system. Once this is successfully completed, the user can go to the Login Page and login into the system.

2.4.2 New User Page

This page prompts the user to enter his/her information such as the ssn, name, email address, context (Student, Instructor, or Administrator), and to select a UserID and Password that is used by the user to access the system. When the user activates the Submit button, the system validates the information entered in the database. If valid, it
sends a confirmation email to the user with a copy of his/her UserID and Password and displays a confirmation message page. Otherwise, it displays an error message asking the user to reenter the information. After successfully completing the registration process, the user can go to the Login Page to access the system.

2.4.3 Login Page

This page allows the user to log in into the system. It prompts the user to enter his/her UserID, Password, and context (Student, Instructor, and Administrator); this identification follows the user through the entire session. When the user activates the Login button, the system validates the authentication of the identifying information entered by the user and, if valid, it transfers the user to the appropriate User Page. Otherwise, it displays an error message and transfers the user back to the Login Page. If the user forgets his/her Password, he/she can activate the Forgot Password link, which transfers him/her to the Forgot Password Page.

2.4.4 Forgot Password Page

This page allows the users to retrieve their forgotten Password. It prompts the user to enter his/her basic details like ssn, UserID, name, and context. The user can activate the Submit button, which validates the information given by the user in the database. If valid, the system sends the correct Password from the database to the user via e-mail, it then transfers the user to the Main Page. If not it displays an error message and transfers the user back to the Forgot Password Page. The user can activate the Cancel button, to the clear the content of the form and can activate the Login link to go to the Login Page.
2.5 The Student Interface

Figure 2, shows the web pages the user sees as he/she navigates through the Student Interface.

Figure 2. Student Interface Map
2.5.1 Student Page

This page allows the student to select the course that he/she wants to access. The student can select from a list of courses he/she is registered in as shown in Figure 3. User can then activate the View link for a course to access the Student Course Details Page for that particular course. The student can activate the Logoff button to logout of the system and go back to the Main Page.

![Figure 3. Student Page](image)

2.5.2 Student Course Page

This page displays basic information about the student, the course selected and has options to Upload Assignment, Review Assignment, Check Assignments & Grade, Edit Profile, Home, Course Page and Logoff as shown in Figure 4. The student can select one of the links to go access the appropriate page. The user can activate the Home button to go back to the Student Page and Logoff button to logout of the system and go to the Main Page.
2.5.3 Upload Assignment Page

This page provides the user (Student) with a list of valid assignment numbers for him/her to upload into the system. It displays details about the assignment like assignment number, due time and date, assignment submission status as shown in Figure 5. The user
has to activate the *Submit Assignment* link to select an assignment he/she wants to upload. This transfers the user to the Assignment Details Page.

### 2.5.4 Assignment Details Page

This page allows the user to submit assignments by uploading the required assignment into the system. The user has to copy the assignment he/she wants to submit in text format and paste it in the blank text area in the page as shown in Figure 6. If the user has already submitted the assignment, the submitted assignment shows up the text area for the user to makes updates. The user can then activate the *Submit* button to upload the assignment or the necessary changes. It displays a confirmation message after successful completion of the upload. The user can also activate the *Back* link to go back to the Upload Assignment Page.

![Figure 6. Assignment Details Page](image)
2.5.5 Review Assignments Page

This page displays a list of assignment numbers each user has to review. The user can select an assignment number that he/she wants to review and activate the *Review Assignment* button. This transfers him/her to the List Assignments Page.

2.5.6 List Assignments Page

This page displays a list of assignments of peers in the class that the user has to evaluate or has evaluated as shown in Figure 7. If the user has not submitted the selected assignment, the user cannot view any submitted assignment of other students and the page displays an error message as a security feature to avoid plagiarism. If the user has submitted the assignment then he/she can select an assignment to review and activate the *Evaluate/Evaluated* button to go to the Evaluation Page for that assignment.

![Review Assignments Page](image)

*Figure 7. Review Assignments Page*
2.5.7 Evaluation Page

This page displays the assignment the user has selected to review in the Review Assignment Page. The page is divided into three sections; first displays the assignment, the second contains the rubric for evaluating the assignment, and a third displays the comment box for the user to submit his/her comments on the assignment. The rubric section has a set of entry boxes for the user to assign points for each criteria outlined on the basis of different categories of evaluation specified by the instructor. The user can activate the *Submit* button to submit the evaluation rubric and the comments when he/she has entered the necessary text boxes. This displays a message box asking whether the user is sure he/she wants to submit the evaluations and the comments. If the user selects *OK* the evaluation is added to the assignment, if the user selects *Cancel* option, it takes him/her back to the Evaluation Page. The user can activate the *Clear* button in the Evaluation Page to clear whatever entries he/she has made. The user can also activate the *Back* link to go back to the Review Assignment Page.

2.5.8 Check Assignments Page

This page allows the user to check grades and evaluations of assignments that he/she has submitted. The page displays a list of assignments along with the grades given by the instructor and peers as shown in Figure 8. The user can select an assignment by activating the *Evaluations* link that transfers him/her to the Checked Assignment Page to read the evaluations given by their peers and the instructor.
2.5.9 Checked Assignment Page

This page allows the user to read the evaluations given by his/her peers on the assignment. If there are more than one evaluation given on an assignment, links for each shows up in the page for the user to view one evaluation at a time as shown in Figure 9. The student only views the evaluations and does not know the evaluator’s identity other than when the evaluator is the instructor. The page displays the evaluations of the instructor in detail including the changes in the assignment made by the instructor. The student can activate the Back link to go back to the Check Assignment Page.

2.5.10 Edit Profile Page

This page allows a student to change some of his/her details such as name, e-mail address, UserID, and Password, and has Submit and Cancel buttons as shown in Figure 10. The student can enter new information, activate the Submit button that updates the database accordingly after checking for valid information, and shows a confirmation message. If the user changes his/her UserID, he/she has to log into the system again after
making the changes. To change the Password, the user has to enter new Password along with the old Password and activate the *Change Password* button; this updates the database accordingly and transfers the student back to the Edit Profile Page.

![Figure 9. Evaluation Page](image)

![Figure 10. Edit Profile](image)
2.6 The Instructor Interface

Figure 11, shows the web pages the user sees as he/she navigates through the Instructor Interface.
2.6.1 Instructor Page

This page allows the instructor to select the course that he/she wants to access. The instructor can select from a list of courses he/she is assigned and activate the View link to transfer to the Instructor Course Details Page for that particular course. The instructor can activate the Logoff button to log out of system and transfer to the Main Page.

2.6.2 Instructor Course Page

This page displays basic information about the instructor and the course that he/she is accessing. The page also has links to the various instructor tools like Edit Assignments, Check & Grade Assignments, Download Assignments, Grade Book, Edit Profile, Course Page, Home and Logoff button as shown in Figure 12. The instructor can activate the Home button to go back to the Course page to select a different course. The Logoff button can be activated to logout of the system and return to the Main Page.

![Figure 12. Instructor Course Page](image)
2.6.3 Edit Assignments Page

This page allows the instructor to add new assignments for a course or edit existing assignments. If the instructor wants to make changes in an assignment he/she can select an assignment number from the list of existing assignments and activate the *Edit Assignment* button. If the instructor wants to add a new assignment he/she can activate the *Add New Assignment* button. Both these buttons that takes the instructor to the Assignment Details Page.

2.6.4 Assignment Details Page

This page allows the instructor to enter new assignment details or edit existing assignment details as shown in Figure 13. If the instructor is adding a new assignment, he/she is prompted to enter the assignment number, the date and time its due, and the number of questions in the rubric for this assignment. After entering the correct information, he/she can activate the *Add* button to add the assignment in the system. If the instructor is editing an existing assignment, the details of that assignment are displayed when the page loads and the *Update* button is displayed instead of the *Add* button for the user to activate after making any changes. The instructor can also delete an assignment by activating the *Delete* button. After the instructor adds a new assignment the *Create Rubric* link becomes active. He/She can then activate the *Create Rubric* link to create a new rubric for the assignment; this takes him/her to the Edit Rubric Page. If the assignment and rubric already exists then the *Edit Rubric* link becomes active to allow the user to make changes in an existing rubric.
2.6.5 Edit Rubric Page

This page allows the instructor to create or edit the Rubric Details for an assignment. The instructor can create the rubric for an assignment in two ways, by uploading a file or by selecting questions from previous assignments as shown in Figure 14. If the instructor wants to use the file upload feature, he/she has to create a tab delimited text file in the desired format, containing the rubric question. He/she can then upload the file by selecting the file from the local machine using the *Browse* button and activating the *Upload* button to load the file into the system. Using the other feature, the instructor can also create a rubric by selecting questions of previous assignments from the drop down list boxes that displays a list of distinct rubric questions already in the system. The instructor has to specify the maximum points he/she wants to assign for each point in the boxes next to the rubric questions when using this feature before he/she can add the new
rubric in the system. This page also allows the instructor to make changes in existing rubric questions and also to delete a rubric by activating the Delete button. He/she can also activate the Back link to go back to the Assignment Details Page.

![Edit Rubric Page](image)

**Figure 14. Edit Rubric Page**

### 2.6.6 Review Assignments Page

This page displays a list of assignments that have past the due date for the instructor to grade. The instructor can select the assignment number he/she wants to grade by activating the corresponding View Assignments link. This takes him/her to the Submitted Assignments Page to view all the assignments that the students have submitted.

### 2.6.7 List Assignments Page

This page displays information about each students assignment like, late submissions, peer review grades, final grades and also creates links to View Evaluations and Evaluate Assignment as shown in Figure 15, for the instructor to select what he/she wants to do. If
the instructor wants to view the assignment and the peer evaluations, he/she can activate the corresponding View link, which opens the Checked Assignment Page for the selected assignment. To evaluate an assignment the instructor can activate the corresponding Evaluate link to open the Evaluation Page for that assignment. The instructor can also assign a grade to the student on an assignment and the grade automatically gets updated in a grade book. The instructor can activate the Back link to transfer back to the Check Assignments Page.

**Figure 15. List Assignments**

### 2.6.8 Evaluation Page

This page is the same as the Evaluation Page in the Student Interface as discussed in section 2.5.7 earlier. The only difference in the instructor interface is the instructor can make changes in the text area displaying the student assignment and submit the changes for the student to view later.
2.6.9 Checked Assignment Page

This page is the same as the Checked Assignment Page in the Student Interface as discussed in section 2.5.9 earlier.

2.6.10 Grade Book

This page maintains the grades assigned to students on the assignments. An instructor can change grades activating the *Change* button as shown in Figure 16. The instructor can activate the *Course Page* button to transfer to the Instructor Page.

![Grade Book](image.png)

*Figure 16. Grade Book*

2.6.11 Edit Profile Page

This page is the same as the Edit Profile Page in the Student Interface as discussed in section 2.5.10 earlier.
2.7 The Administrator Interface

Figure 17, shows the web pages the user sees as he/she navigates through the Administrator Interface.
2.7.1 Administrator Page

This page is the main page for the instructor interface. It displays basic information about the administrator and the status of the system. The page also has options to Edit Instructors, Edit Courses, Edit Students, Edit Classes, and Edit Administrator, Home and Logoff button as shown in Figure 18. The Logoff button can be activated to logout of the system and return to the Main Page.

Figure 18. Administrator Page

2.7.2 Edit Instructors Page

This page allows the administrator to view the list of all the instructors in the system as shown in Figure 19. Each instructor information is associated with an Edit link that enables the administrator to edit the instructor information. If the administrator wants to add a new instructor into the system, he/she can activate the Add New Instructor button that transfers him/her to the Instructor Details Page.
2.7.3 Instructor Details Page

This page allows the administrator to Add, Update and Delete instructors in the system. If the administrator is adding a new instructor, he/she enters the required information as shown in Figure 20, and activates the *Add* button. The system validates the entries, updates the database, and displays a confirmation message. If an error occurs it displays an error message. If the administrator wants to makes changes in an existing detail, he/she makes the necessary changes and activates the *Update* button. If the system finds an invalid entry, he/she gets an error message, or else a conformation message is displayed. If the administrator wants to delete an instructor, he/she can activate the *Delete* button. A confirmation message pops up to make sure the user wants to delete the information, if the user selects the *OK* button, the system goes ahead and deletes the information, if the user selects the *Cancel* button, it takes him/her back to the Instructor Details page.
2.7.4 Edit Courses Page

This page allows the administrator to view the list of courses in the system. Each course information is associated with an Edit link that enables the administrator to edit the instructor information. The administrator can activate the Add New Course button if he/she wants to add a new course into the system. This transfers him/her to the Course Details Page.

2.7.5 Course Details Page

This page allows the administrator to Add, Update and Delete courses in the system. It prompts the administrator to enter the CourseID, Course Number and any other course details as shown in Figure 21. If the administrator is adding a new course, he/she enters the required information and activates the Add button. The system does the required validation, updates the database, and displays a confirmation message. If the administrator wants to makes changes in an existing course detail, he/she makes the
necessary changes and activates the *Update* button. If the system finds an invalid entry, 
he/she gets an error message, or else a conformation message is displayed. If the 
administrator wants to delete a course, he/she can activate the *Delete* button. A 
confirmation message pops up to make sure the user wants to delete the information, if 
the user selects the *OK* button, the system goes ahead and deletes the information, if the 
user selects the *Cancel* button, it takes him/her back to the Course Details page.

![Course Details Page](image)

2.7.6 Edit Students Page

This page allows the administrator to search for existing students in the system and add 
new students in the system. The user has to enter the SSN of the student he/she wants to 
search or wants to add. He/she then activates the *Search* button and this transfers him/her 
to the Student Details page with the necessary information.

2.7.7 Student Details Page
This page allows the administrator to Add, Update and Delete students in the system. If the administrator is adding a new student, he/she enters the required information and activates the Add button. The system does the required validation, updates the database, and displays a confirmation message. If an error occurs it displays an error message. If the administrator wants to make changes in an existing course detail, he/she makes the necessary changes and activates the Update button. If the system finds an invalid entry, he/she gets an error message or else a confirmation message is displayed. If the administrator wants to delete a student, he/she can activate the Delete button. A confirmation message pops up to make sure the user wants to delete the information, if the user selects OK button, the system goes ahead and deletes the information, if the user selects the Cancel button, it takes him/her back to the Student Details page.

2.7.8 Edit Administrators Page

This page is the same as the Edit Instructors Page in the Administrator Interface as discussed in section 2.7.2 earlier.

2.7.9 Administrator Details Page

This page is the same as the Instructor Details in the Administrator Interface as discussed in section 2.7.3 earlier.

2.7.10 Edit Classes Page

This page allows the instructor to create and update class information for a course. The page displays a list of courses that are existing in the system as shown in Figure 22. The administrator has to create the course first, before he/she can add a class for that course.
2.7.11 Class Details Page

This page allows the administrator to Add, Update and Delete students in a class for a particular course. This page displays a number of drop down boxes as shown in Figure 23, which contain the information about students in the system that the administrator has already added. The administrator can select the students that are in a class and activate the Add button to add the students selected for that particular class. The administrator can also update an existing class information by changing students or adding new ones and activating the Update button. If the administrator wants to delete a student in a class or delete the whole class he/she can activate the Delete Class button. This transfers him/her to the Delete Class Page.
2.7.12 Delete Class Page

This page allows the administrator to delete single student in a class or delete the whole class details. The page displays a list of students in the class with a link to Delete Student that enables the administrator to delete a particular student. If the administrator wants to delete the entire class, he/she can activate the Delete Entire Class button, which prompts the user with a confirmation message to make sure whether the user really wants to delete the class information.

3. Project Environment

The RRAS is a web-based application, and can be viewed using any Web browser like Netscape and Internet Explorer. The system has a MySQL database in UNIX/Linux environment as the backend. Server side PHP scripts and CGI scripts with embedded queries to interact with the MySQL database and Hypertext Markup Language was used to generate dynamic Web pages. Structured Query Language (SQL) is used for writing the queries to access the database. Client side JavaScripts was used to validate user input and Images were created using Microsoft Image Composer.
4. System Design and Implementation

4.1 System Analysis

The Design and Analysis Phase of the project to develop the Read, Review, and Assess System (RRAS) started with the following steps:

1. Discussed with Dr. Holly Patterson-McNeill and Prof. Carol Binkerd about what type of features were needed in the RRAS. The features discussed included user interface, security, database design, and data integrity.

2. Designed a framework of what the user interface would look like, proposed, and analyzed the design with Dr. Holly Patterson-McNeill and Dr. Dulal Kar.
3. Designed and developed a relational schema for the database based on E-R model components including entities, attributes, and relationships between the entities.

4. Researched various Web database applications to choose which software to use in the development of RRAS.

RRAS is a Web-based application; the main issues that came up during the design process were to provide good user interface, security and data integrity.

• **User Interface:** The reason for developing the RRAS is to provide a user-friendly Web environment to implement peer review for evaluating students. Most users of this system are new to the idea and have very limited understanding of the whole process of peer review. To design a good and user friendly interface became a vital component in system design because the user should have no problems in using the tools provided and in navigating through the various pages in the system. Providing proper documentation and help features also formed an integrated part of the user interface.

• **Security:** Security is important in an environment where access criteria to the database depends on the type of user. The system uses sessions set during the user login to provide appropriate access to the system. Security features like user identification and authentication, system time out, disabling browser cache entries, and disabling options when the user is not qualified to use a certain feature are some of the security features that is used in the system.

• **Database Integrity:** The design of any database system has issues of data integrity related to it. The various data integrity features that has been used while developing
the RRAS are: handling improper data, updating proper tables and mapping standard MySQL error messages with user friendly messages.

The following steps were taken to complete the development and testing of the Read Review and Access System on the Web.

- Implemented the system using HTML, MySQL, and PHP scripts.
- Installed and tested the system with active data.
- Developed user manual.

4.2 System Architecture & Overview

The RRAS architecture is ordered into three tiers: the user interface in the client browser, the Web server, and the database server. Figure 24 shows a graphical representation of the architecture. The functional description of each tier in the architecture is given below:

- **First Tier**: This comprises of the user interface as viewed on the client’s Web browser. The user interface is implemented using HTML pages with or without JavaScript support. JavaScript is used to validate user input and is executed on the client’s Web browser.
- **Second Tier**: The second tier consists of an Apache Web server with mod_php (PHP module). The PHP module includes support for several database backends including
MySQL. The Apache Web server is used to execute PHP and CGI scripts which process the user input, generate SQL queries to access the database and then process the output in dynamically generated HTML pages.

- **Third Tier**: The third tier consists of the MySQL database server that receives requests from the PHP/CGI scripts to execute SQL statements. The MySQL server executes the SQL statements and send the queried data back to the scripts. The server accesses the database files to retrieve the stored data.

An overview of the system is as follows: The user enters the URL to access the RRAS in his/her Web browser. The browser sends a request to the Web server to fetch an HTML page or execute a PHP/CGI script. A PHP/CGI scripts uses SQL queries to access the RRAS database tables, insert, update or retrieve the data and dynamically generated HTML pages to send back to the client’s browser. An HTML page returned to the browser has links to utilities in the system and other pages for the user to navigate through the RRAS.
4.3 Database Design and Implementation

4.3.1 Design and Entity Relationship

The database for the RRAS is built-in MySQL 3.22 relational database system running under the Unix/Linux operating system. The database design is done keeping in mind any future expansions in the RRAS system and to maintain the data integrity of the system. The database is designed in a relational model; It consists of five static tables and four dynamically created tables for each course in the system. Table 2 presents a list of static and dynamic tables in the system and their brief description.
### TABLE NAME | DESCRIPTION
---|---
**Static Tables:**
Student_Details | Student information and UserID
Instructor_Details | Instructor information and UserID
Administrator_Details | Administrator information and UserID
Course_Details | Course information (CourseID (Primary Key))
Student_Courses | Student Course map

**Dynamically Generated Tables: (For each Course in the System)**
CourseID | Class Details
CourseIDAAssignment | Assignment details
CourseIDRubric | Rubric details
CourseIDRubricAssignment | Evaluated Rubric Details

---

**Table 2. RRAS Database Tables**

Figure 25 shows the Entity-Relationship diagram for RRAS. The static tables Student_Details, Instructor_Details, and Administrator_Details store the information about the various types of users in the system. The primary key is SSN, which is used as a reference key or foreign key by other tables to maintain the integrity constraint. The Course_Details table contains the information about each course; the primary key CourseID is used as a key to create tables for each course dynamically. There are 4 dynamically created tables per course, The CourseID table holds the Class Details, Submitted Assignment, and Grade details of each Student in that course. The CourseIDAAssignment stores the assignments and CourseIDRubric table stores rubric information. For each student the CourseIDRubricAssignment table stores the evaluations done by his/her peers on his/her assignments. The data dictionary for each table is provided in Appendix A.
4.3.2 Implementation of Tables

The static tables are created first and in the following order; Administrator_Details, Student_Details, Instructor_Details, Course_Details and Student_Courses. The Administrator_Details table is populated for the first time using the SQL command “Insert”. The rest of the tables including the Administrator Details can be populated using the Administrator Interface. The CourseID related tables are created dynamically once the Course_Details table is populated with data by the Administrator. The CourseID table has fields that are static and dynamic. The static fields are the ones that are created when the table is created i.e. StudentSSN and Grade, the dynamic fields are created whenever the instructor adds a new assignment. The fields are as follows: Assignment#,
GradedAssignment#, Grade#, SubTime#, and SubFlag#. The CourseIDRubric table can be populated using a “File Upload” if required. The data from the file is loaded into the table using the “Load Data” function in MySql. The syntax of the Load Data function is given below. The details of the format of the Text file are given in Appendix A.

```sql
LOAD DATA [LOW_PRIORITY] [LOCAL] INFILE 'file_name.txt'
[REPLACE | IGNORE]
  INTO TABLE tbl_name
  [FIELDS
    [TERMINATED BY '\t']
  ]
  [LINES TERMINATED BY '\n']
```

4.4 RRAS System Modules

This section contains description of the different modules, which constitutes the Common, Student, Instructor, and Administrator Interfaces of RRAS. The Common interface consists of the modules that are common to all users i.e. NewUser, Login, and ForgotPassword. The security of the Web pages for the three main interfaces of RRAS is maintained by using session cookies that are generated when the user logs into the system. Generated session cookies include UserID, SessionID and the Context of the user, the UserID is selected by the user when he/she registers in the system, the SessionID is generated from a random value and the Context is selected by user when
he/she logs in. All other scripts of the interface compare these values with the values stored in the corresponding Context_Details table. If the values do not match, the user is transferred back to the Login Page.

4.4.1 Common Interface Modules

Following are the module numbers to module name keys for the Common Interface.

<table>
<thead>
<tr>
<th>Module No</th>
<th>Module Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.4.1.1</td>
<td>New User</td>
</tr>
<tr>
<td>4.4.1.2</td>
<td>Login</td>
</tr>
<tr>
<td>4.4.1.3</td>
<td>Forgot Password</td>
</tr>
</tbody>
</table>

Table 3 displays the relationship between database tables and Common Interface modules.

<table>
<thead>
<tr>
<th>Module</th>
<th>Student_Detials</th>
<th>Instructor_Detials</th>
<th>Admin_Details</th>
<th>Course_Details</th>
<th>Student_Courses</th>
</tr>
</thead>
</table>

41
<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4.4.1.1</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>4.4.1.2</td>
<td>R,M</td>
<td>R,M</td>
</tr>
<tr>
<td>4.4.1.3</td>
<td>R,M</td>
<td>R,M</td>
</tr>
</tbody>
</table>

Notation: R=Read, M=Modify

### 4.4.1.1 New User (NewUser.php, UpdateUser.php)

**Input:** SSN, First Name, Middle Initial, Last Name, Email, UserID, Password, Context.

**Function:** This NewUser.php script displays the New User Page, which prompts a new user, who is accessing RRAS for the very first time to input his/her details to register in the system. When the user submits the form the UpdateUser.php script is executed, this verifies the data entered by the user with the value of SSN stored in the corresponding Context_Details table. If the input data is correct, the UpdateUser.php script updates the Context_Details table with the information entered by the user. It then sends a confirmation e-mail to the user with a copy of UserID, Password and a message of successful activation of the user’s account on RRAS. A message is then displayed confirming the registration and the user is taken to the Login page to access the system. If the input data is incorrect the UpdateUser.php script transfers the user back to the New User Page and displays an error message prompting the user to re-enter the data again. This module also displays buttons as links to other pages.

**Tables:** Admin_Details, Instructor_Details, and Student_Details

**Output:** If the data entered by the user is correct, a page containing the confirmation that “the email has been sent” is displayed. If the data entered by the user is incorrect, the module displays the New User Page with an error message and prompts the user to re-
enter the information again. The module also displays buttons as links to the Main Page and the Login Page.

4.4.1.2 Login (Login.php, CheckLogin.php)

**Input:** User ID, Password, and Context

**Function:** The Login.php script deletes all previously set cookies and displays the Login Page that prompts the user to enter his/her UserID and Password. When the user submits the form, the CheckLogin.php script is executed. The CheckLogin.php script authenticates the user by comparing the values entered with the values stored in the Context_Details table. If the values provided are incorrect, the user is transferred back to the Login Page, and an error message is displayed, otherwise the CheckLogin.php script generates a random session value and updates the SessionID field in the Context_Details table. The CheckLogin.php script also generates three session cookies; UserID of the user, Context of the user and a random value called SessionID. The user is then transferred to the corresponding User Page.

**Tables:** Admin_Details, Instructor_Details, and Student_Details

**Output:** If the values provided by the user are incorrect, the script displays the Login Page containing an error message and prompting the user to re-enter his/her SSN and Password. If the values provided are valid, the corresponding Context Page is displayed.

4.4.1.3 Forgot Password (ForgotPasswd.php, NewPasswd.php)

**Input:** SSN, First Name, Last Name, UserID, and Context
**Function:** The *ForgotPasswd.php* script displays the Forgot Password page, which prompts the user to enter his/her details. When the user submits the form the *NewPasswd.php* script is executed, this verifies the values entered by the user with the values stored in the Context_Details table. If the data provided by the user is correct, then the *NewPasswd.php* script generates a new Password and sends it to the user via e-mail. A confirmation message is displayed that the Password has been delivered to the user via e-mail. If the data provided by the user is incorrect then the user is transferred back to the Forgot Password page and an error message is displayed. The *NewPasswd.php* script accesses the Context_Details table in order to retrieve all the pertinent information about the user.

**Tables:** Admin_Details, Instructor_Details, and Student_Details

**Output:** A page containing the confirmation message is displayed.

### 4.4.2 Student Interface Modules

Following are the module numbers to module name keys for the Student Interface.

<table>
<thead>
<tr>
<th>Module No</th>
<th>Module Name</th>
<th>Module No</th>
<th>Module Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.4.2.1</td>
<td>Student Page</td>
<td>4.4.2.6</td>
<td>Checked Assignment Details</td>
</tr>
<tr>
<td>4.4.2.2</td>
<td>Student Course Details</td>
<td>4.4.2.7</td>
<td>Review Assignments</td>
</tr>
<tr>
<td>4.4.2.3</td>
<td>Upload Assignment</td>
<td>4.4.2.8</td>
<td>List Assignments</td>
</tr>
<tr>
<td>4.4.2.4</td>
<td>Assignment Details</td>
<td>4.4.2.9</td>
<td>Evaluation Page</td>
</tr>
<tr>
<td>4.4.2.5</td>
<td>Check Assignments</td>
<td>4.4.2.10</td>
<td>Edit Profile</td>
</tr>
</tbody>
</table>

Table 4 displays the relationship between database tables and Student Interface modules.
### 4.4.2.1 Student Page

**Input:** None

**Session Cookies:** StudentID, SessionID

**Function:** The StudentPage.php script displays the Student Page, which consists of information of courses in which the student is registered. The script uses the StudentID to access the Student_Details table to retrieve the Student SSN. Using the SSN, it retrieves the CourseID of courses that the student is registered from the Student_Courses table. It then retrieves the course information to display on the page from Course_Details table using the CourseID as the key. The StudentPage.php script also displays a View link for
the student to access each course. When the student activates the View link it takes the student to the course Details Page for that particular course. The student may opt to activate the Logout link to logout of the system.

**Tables:** Student_Details, Student_Courses, and Course_Details

**Output:** A page containing a list of courses and their details in which the student is registered. This script also displays links to Logout and View options.

### 4.4.2.2 Student Course Details

**Input:** CourseID

**Session Cookies:** StudentID and SessionID

**Function:** This StudentCourseDetails.php script displays the Student Course Details Page with detail information about the student and a set of links to the various utilities that a student can access in the system. This script uses the StudentID to retrieve the student details from the Student_Detail and CourseID to retrieve the course information from the CourseID table. This script also displays buttons as links to other pages.

**Tables:** Student_Details, CourseID, and Course_Details

**Output:** A page containing information about the student. This page also contains links to the various utilities in the system given as follows: Upload Assignment, Review Assignment, Check Assignment, Edit Profile, Course Page, Home, and Logoff.

### 4.4.2.3 Upload Assignment

**Input:** None

**Session Cookies:** StudentID, SessionID, and CourseID
Function: This *UploadAssignment.php* script displays Upload Assignment Page with a list of assignment numbers for the user to submit. The script uses the *CourseID* to query the CourseIDAssignment table to retrieve the list of assignments that are active. It also queries the CourseID table to retrieve information about the assignment submission status for each assignment for that particular user using the *StudentSSN*. The script displays a link to *Submit Assignment* for each assignment that the user can activate to submit an assignment. The script also displays links to the various tools that the student has access.

Tables: Student_Details, CourseID, and CourseIDAssignment

Output: A page containing a list of assignments for the course, their details and the user's submission status are displayed. The script also displays a link to each assignment submission page.

4.4.2.4 Assignment Details (*AssignmentDetails.php*, *AddUpdateAssignment.php*)

Input: Assignment Number, Assignment

Session Cookies: *StudentID*, *SessionID*, and *CourseID*

Function: The *AssignmentDetails.php* script displays the Assignment Submission Page that contains a text area for the user to enter his/her assignment and button to *Add*, *Delete* and *Update* an assignment. When the student submits the changes in the assignment and activates the buttons, a JavaScript displays a confirmation message, asking the user to confirm his/her submission. If the user selects the *OK* option the *AddUpdateAssignment.php* script is executed, which adds or updates the valid Assignment#, related fields like Submission Time and Submission Flag in the CourseID
table using the CourseID, Assignment Number and the StudentSSN fields. The AddUpdateAssignment.php script then displays a confirmation message. If the user selects the Cancel option in the confirmation message, the AssignmentDetails.php script takes the control back the Assignment Details Page. The AssignmentDetails.php script also displays links to the Upload Assignment Page and the Student Course Details Page.

**Tables:** Student_Details, CourseID, and CourseIDAssignment

**Output:** Assignment Details Page containing the text area for the user to enter his/her assignment. When the user activates the Add, Update or Delete button, the assignment changes are made in the database and a confirmation message is displayed.

### 4.4.2.5 Check Assignments

**Input:** None

**Session Cookies:** StudentID, SessionID, and CourseID

**Function:** The CheckAssignment.php script displays the Check Assignment Page that contains information about the students assignment like, assignment number, grade, submission status, peer evaluation grade and a link to View the evaluations. The script uses the StudentSSN and CourseID to retrieve information about the grade and submission time for each assignment from the CourseID and CourseIDAssignment table. The script also retrieves the peer evaluation grade for each assignment of that particular student from the CourseIDRubricAssignment table using the StudentSSN and doing the calculations to get the average grade. The script creates a link to View the evaluations for each assignment which when activated takes the user to the Checked Assignment Details Page.
Tables: Student_Details, CourseID, CourseIDRubricAssignment, CourseIDAssignment

Output: A page containing the information about all assignments for that student including assignment grade, submission status, peer evaluation grade and a link to View the evaluations.

4.4.2.6 Checked Assignment Details

Input: Assignment Number, Rubric ID

Session Cookies: StudentID, SessionID, and CourseID

Function: The CheckedAssignmentDetails.php script displays the Checked Assignment Details Page, which contains the evaluation of the assignment by peers and instructor. The evaluations are in the form of rubric with points assigned to each question and individual comment by the evaluators. The script uses the StudentID and AssignmentNumber to query the CourseIDRubricAssignment table to retrieve the evaluations and the comments on an assignment. Querying the CourseIDRubric table using the AssignmentNumber and CourseID retrieves the rubric questions and other details. Each student’s assignment may have multiple evaluations, so the script creates a numeric link to each set of evaluation and comment for the student to view. If the evaluator is the instructor, the script displays a corrected copy of the assignment by querying the CourseID table and lets the student know that the particular evaluation was by the instructor. The script also displays links to the Check Assignment Page and the Student Course Details Page.

Tables: Student_Details, CourseID, CourseIDRubricAssignment, CourseIDAssignment, and CourseIDRubric.
Output: A page displaying the evaluation for an assignment by peers and instructor in the form of rubric and comments. The evaluations of the instructor are displayed with a copy of the corrected assignment. The script also displays links to the Check Assignment Page and the Student Course Details Page.

4.4.2.7 Review Assignments

Input: None

Session Cookies: StudentID, SessionID, and CourseID

Function: The ReviewAssignment.php script displays the Review Assignment Page, which displays a list of assignments that the user wants to review. It displays a drop down list box containing a list of assignments retrieved by querying the CourseIDAssignment table. When the user selects an assignment number and activates the Review Assignment button, the script transfers him/her to the List Assignment Page.

Table: Student_Details, CourseIDAssignment

Output: A page displaying the form to select the assignment number and a Review Assignment button for the user to activate to review the selected assignment.

4.4.2.8 List Assignments

Input: Assignment Number

Session Cookies: StudentID, SessionID, and CourseID

Function: This ListAssignment.php script displays List Assignment Page with a list of assignment to be evaluated by the user for an assignment. The script uses the StudentID and AssignmentNumber to query the CourseID table to check if the student has submitted
that particular assignment. If the student has not submitted the assignment, the script displays a message notifying the user that he/she has to submit the assignment in order to review other’s assignment. The scripts also retrieves the due date and time from the CourseIDAssignment table to check if the present date and time is past due date time. If not, the user is requested to try again after the due date and time else it displays the list of assignments. The list of assignments to be evaluated is accompanied by a link to the Evaluation Page for each assignment. The link is in “Evaluate” status, when the user has not evaluated the assignment and in “Evaluated” status when he/she finishes evaluating the assignment. A student may evaluate the assignment as many times as he/she wants to.

Tables: Student_Details, CourseID, CourseIDRubricAssignment, and CourseIDAssignment

Output: A page containing a list of assignments to be evaluated, and an Evaluate/Evaluated link to Evaluation Page for each of them.

4.4.2.9 Evaluation Page (EvaluationPage.php, UpdateRubric.php)

Input: Assignment ID, Assignment Number, Comments and Rubric details

Session Cookies: StudentID, SessionID, and CourseID

Function: The EvaluationPage.php displays the Evaluation Page for an assignment. The page allows the user to evaluate the assignment selected in the List Assignment Page. The EvaluationPage.php script uses the AssignmentID and AssignmentNumber to query the assignment to be evaluated from the CourseID table and displays it in the text area of the page. Querying the CourseIDRubric table using the AssignmentNumber retrieves the rubric questions and the maximum points for each question. It then displays the rubric template with a set of entry boxes for the user to enter the points for each criterion.
specified and a comment field for the user to enter his/her comments on the assignment. When the user submits the evaluation a JavaScript check for valid entries and prompts user with a message box for his/her confirmation. If the user selects OK option, the \textit{UpdateRubric.php} script is executed which saves the evaluated rubric and comment in the CourseIDRubricAssignment table. The \textit{UpdateRubric.php} script then displays a confirmation message. If the user selects Cancel option, the \textit{EvaluationPage.php} script displays the Evaluation Page again. This module also displays buttons as links to other pages.

\textbf{Table:} Student\_Courses, CourseID, CourseIDRubricAssignment, and CourseIDRubric

\textbf{Output:} A page with the assignment to be evaluated, rubric for evaluating the assignment and comments field for the user to enter the comments for the assignment. After the assignment is evaluated and submitted the \textit{UpdateRubric.php} script is executed, this saves the evaluated rubric and displays a confirmation message. The Page also displays buttons as links to the Student Course Details Page and List Assignment Page.

\subsection*{4.4.2.10 Edit Profile Page (EditProfile.php, UpdateUser.php)}

\textbf{Input:} First Name, MiddleIni, Last Name, Email, Year, Major, UserID, Old Password, and New Password

\textbf{Session Cookies:} StudentID, SessionID, and CourseID

\textbf{Function:} This \textit{EditProfile.php} script allows the user change his/her profile entry, UserID and Password in the database. The script displays two forms one for displaying and making changes in the personal details and UserID and the second for displaying and making changes in the Password. It displays the existing student details by querying the
Student_Details table using the Student ID. If the data entered by the user is valid, the UpdateUser.php script updates the Student_Details with the input data and displays a confirmation message. If the data entered is invalid, the script displays an error message and it restores the original data in the Student_Details table. If the user changes his/her UserID, the system prompts him/her to login to the system again after making the changes in the database.

**Table:** Student_Details

**Output:** A page displaying two forms with the student details. If the data entered by the user is valid the UpdateUser.php script updates the Student_Details and displays a confirmation message. If the data entered is invalid an error message is displayed and the Edit Profile Page is displayed again.

### 4.4.3 Instructor Interface Modules

Following are the module numbers to module name keys for the Instructor Interface.

<table>
<thead>
<tr>
<th>Module No</th>
<th>Module Name</th>
<th>Module No</th>
<th>Module Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.4.3.1</td>
<td>Instructor Page</td>
<td>4.4.3.7</td>
<td>List Assignments</td>
</tr>
<tr>
<td>4.4.3.2</td>
<td>Instructor Course Details</td>
<td>4.4.3.8</td>
<td>Checked Assignment Details</td>
</tr>
<tr>
<td>4.4.3.3</td>
<td>Edit Assignments</td>
<td>4.4.3.9</td>
<td>Evaluation Page</td>
</tr>
<tr>
<td>4.4.3.4</td>
<td>Assignment Details</td>
<td>4.4.3.10</td>
<td>Grade Book</td>
</tr>
<tr>
<td>4.4.3.5</td>
<td>Edit Rubric</td>
<td>4.4.3.11</td>
<td>Edit Profile</td>
</tr>
<tr>
<td>4.4.3.6</td>
<td>Review Assignments</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 5 displays the relationship between database tables and Instructor Interface modules.
<table>
<thead>
<tr>
<th>Module</th>
<th>Instructor Details</th>
<th>Student Details</th>
<th>Course Details</th>
<th>Course ID</th>
<th>CourseID Assignment</th>
<th>CourseID Rubric</th>
<th>CourseID Rubric Assignment</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.4.3.1</td>
<td>R</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.4.3.2</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.4.3.3</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.4.3.4</td>
<td>R</td>
<td>M,D</td>
<td>R,A,D</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.4.3.5</td>
<td>R</td>
<td></td>
<td></td>
<td>R,M</td>
<td></td>
<td>R,M</td>
<td></td>
</tr>
<tr>
<td>4.4.3.6</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.4.3.7</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>4.4.3.8</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>4.4.3.9</td>
<td>R</td>
<td>R,M</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R,A,M,D</td>
<td></td>
</tr>
<tr>
<td>4.4.3.10</td>
<td>R</td>
<td>R</td>
<td>R,M</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.4.3.11</td>
<td>R,M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Notation: R=Read, A=Add, M=Modify, D=Delete

**4.4.3.1 Instructor Page**

**Input:** *None*

**Session Cookies:** InstructorID, SessionID

**Function:** The *InstructorPage.php* script displays the Instructor Page that consists of information of courses that are assigned to the instructor. The script uses the InstructorID to access the Instructor_Details table to retrieve the Instructor SSN. Using the SSN, it retrieves the CourseID of courses that the instructor is assigned from the Course_Details table. It then retrieves the course information to display on the page from Course_Details table using the CourseID as the key. The *InstructorPage.php* script also displays a *View* link for the instructor to access each course. When the instructor activates the *View*
hyperlink, it takes the instructor to the Course Details Page for that particular course. The instructor may opt to activate the Log out link to logout of the system.

**Tables:** Instructor_Details, and Course_Details

**Output:** A page containing a list of courses and their details that the instructor is assigned. This script also displays links to Log out and View options.

### 4.4.2.2 Instructor Course Details

**Input:** CourseID

**Session Cookies:** InstructorID, SessionID

**Function:** This InstructorCourseDetails.php script displays the Instructor Course Details Page with detail information about the instructor and a set of links to the various utilities that a instructor can access in the system. This script uses the InstructorID to retrieve the instructor details from the Instructor_Details and CourseID to retrieve the course information from the CourseID table. This script also displays buttons as links to Edit Assignment, Review Assignment, Edit Grade, Edit Profile, Course Page, Home, and Log out options.

**Tables:** Instructor_Details, CourseID, and Course_Details

**Output:** A page containing information about the instructor. This page also contains links to the various utilities in the system given as follows: Edit Assignment, Review Assignment, Edit Grade, Edit Profile, Course Page, Home, and Log out.

### 4.4.3.3 Edit Assignments

**Input:** None
**Session Cookies:** InstructorID, SessionID, and CourseID

**Function:** The *EditAssignment.php* script displays the Edit Assignment Page, which displays a list of assignments that the instructor wants to edit. It displays a drop down list box containing a list of assignments retrieved by querying the CourseIDAssignment table. When the instructor selects an assignment number and activates the *Edit Assignment* button, the script takes the control to the Assignment Details Page. The script also has an *Add New Assignment* button which when activated transfers the user to the Assignment Details Page for a new assignment.

**Table:** Instructor_Details, CourseIDAssignment

**Output:** A page displaying the form to select the assignment number and two buttons to *Edit Assignment* and *Add New Assignment.*

### 4.4.3.4 Assignment Details (*AssignmentDetails.php, AddUpdateAssignment.php*)

**Input:** AssignmentNumber, DueOn, and Rubric Size

**Session Cookies:** InstructorID, SessionID, and CourseID

**Function:** The *AssignmentDetail.php* script displays the Assignment Details Page which contains information about an assignment the instructor has created or wants to create and displays the *Add, Update* and *Delete* buttons. The script uses the AssignmentNumber to query the CourseIDAssignment table and retrieves the information about the assignment. When the user activates the *Add/Update* button, the script checks for valid input using JavaScript and displays the appropriate error messages. If the data is valid the *AddUpdateAssignment.php* script is executed and the CourseIDAssignment table is
updated. The *AddUpdateAssignment.php* script also modifies the CourseID table adding new columns i.e. Assignment#, GradedAssignment#, Grade#, SubTime# and SubFlag# for all the students in the CourseID table. The *AddUpdateAssignment.php* script displays a confirmation message and then it transfers the user back to the Assignment Details Page to create a rubric for the assignment.

**Table:** Instructor_Details, CourseIDAssignment, CourseID

**Output:** A page displaying details about the Assignment and *Add, Update* and *Delete* buttons. If the submitted data is valid, the *AddUpdateAssignment.php* script displays a confirmation message. If an assignment is already created the *Create Rubric or Edit Rubric* link is activated for the instructor to create a rubric or make changes in one.

### 4.4.2.5 Edit Rubric (*EditRubric.php, AddUpdateRubric.php*)

**Input:** *AssignmentNumber, Rubric Questions, MaxMarks, and FileName*

**Session Cookies:** *InstructorID, SessionID, and CourseID*

**Function:** The *EditRubric.php* script displays the Rubric Details Page which contains information about the rubric for an assignment the instructor has created or wants to create and displays the *Add, Update* and *Delete* buttons. The script uses the AssignmentNumber to query the CourseIDRubric table and retrieves the information about the rubric. The *EditRubric.php* script displays two forms as method to create a rubric template. The first form takes the file name containing the rubric questions and other details. When the *Upload* button is activated the *AddUpdateRubric.php* script is
executed and the file is uploaded to the Web Server and the contents of the text file is loaded into the CourseIDRubic table. This method has to be used by the instructor when creating the very first assignment and can also be used for later assignments. The AddUpdateRubric.php script then displays a confirmation message and takes the control back to the Edit Rubric page. The second form in the EditRubric.php script displays a list of drop down selection boxes which has a list of rubric questions from the previous assignments retrieved from the CourseIDRubic table. The user can select questions from the list and prepare a rubric template he/she can then activate the Add/Update button to create a rubric. The AddUpdateRubric.php script is then executes, which updated the CourseIDRubic table and returns control back to the Edit Rubric Page. The Page also has links to Instructor Course Details Page and Assignment Details Page.

**Table:** Instructor_Details, CourseIDRubic

**Output:** A page displays two forms, one for uploading the rubric file and second to select rubric questions from a list of existing question and the Add, Update and Delete buttons. If the submitted data is valid, the AddUpdateRubric.php script displays a confirmation message. The Page also has links to Instructor Course Details Page and Assignment Details Page.

**4.4.3.6 Review Assignments**

**Input:** None

**Session Cookies:** InstructorID, SessionID, and CourseID

**Function:** The ReviewAssignment.php script displays the Review Assignment Page that contains a list of assignment to be evaluated by the instructor for that particular course.
The scripts retrieves the Assignment information like the due date and time from the CourseIDAssignment table and displays information like Assignment Past Due Date or not. The list of assignments to be evaluated is accompanied by a link to Review Assignment that transfers the instructor to the List Assignment Page for that assignment.

**Tables:** Instructor_Details, CourseIDAssignment,

**Output:** A page containing the information about all assignments for that course and a to Review Assignment link to List Assignment Page.

### 4.4.2.8 List Assignments

**Input:** Assignment Number

**Session Cookies:** InstructorID, SessionID, and CourseID

**Function:** This ListAssignments.php script displays List Assignment Page with a list of student assignments to be evaluated by the instructor. It displays information about the students assignment like, assignment number, grade, submission status, peer evaluation grade and a link to Checked Assignment Page and Evaluation Page. The script uses the StudentSSN and CourseID to retrieve information about the grade and submission time for each assignment from the CourseID and CourseIDAssignment table. The script also retrieves the peer evaluation grade for each assignment of that particular student from the CourseIDRubricAssignment table using the StudentSSN. The script creates a link to View the evaluations for each assignment which when activates takes the user to the Checked Assignment Details Page and an Evaluate link to the Evaluation Page for each assignment.

**Tables:** Instructor_Details, Student_Details, CourseID, CourseIDRubricAssignment,
CourseIDAssignment

**Output:** A page containing a list of student assignments to be evaluated, their details and links to Evaluation Page and Checked Assignment Page for each of them.

### 4.4.3.8 Checked Assignment Details

This page is the same as the Checked Assignment Page in the Student Interface as discussed in section 4.4.2.6 earlier.

### 4.4.3.9 Evaluation Page (EvaluationPage.php, AddUpdateRubric.php)

**Input:** AssignmentNumber, StudentID, Comments, and Rubric details

**Session Cookies:** InstructorID, SessionID, and CourseID

**Function:** This page is the same as the Evaluation Page in the Student Interface as discussed in section 4.4.2.9 earlier. The only difference is the Instructor can make changes in the Text Area displaying the student assignment and submit the changes.

**Tables:** Instructor_Details, CourseID, CourseIDAssignment, CourseIDRubric, and CourseIDRubricAssignment

**Output:** Same as the Evaluation Page in the Student Interface as discussed in section 4.4.2.9 earlier.

### 4.4.3.10 Grade Book (EditGrade.php, UpdateGrade.php)

**Input:** Final Grade, Assignment Grades, Assignment Numbers, and StudentID

**Session Cookies:** InstructorID, SessionID, and CourseID
**Function:** This *EditGrade.php* script allows the user to view and edit grades of each student in a particular course. It uses the *CourseID, StudentID, and AssignmentNumber* to retrieve the grades on each assignment of all the students in the course. The script accepts the user input and updates the records in the CourseID table using the *UpdateGrade.php* script. This module also displays buttons as links to other pages.

**Table:** Instructor_Details, CourseID, and CourseIDRubricAssignment

**Output:** A page displaying the grades of all the students in the course. The module checks for any changes in the user input and updates the database accordingly. The module also displays a buttons as link to the Instructor Course Details Page.

### 4.4.4 Administrator Interface Modules

Following are the module numbers to module name keys for the Administrator Interface.

<table>
<thead>
<tr>
<th>Module No</th>
<th>Module Name</th>
<th>Module No</th>
<th>Module Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.4.4.1</td>
<td>Admin Page</td>
<td>4.4.4.7</td>
<td>Course Details</td>
</tr>
<tr>
<td>4.4.4.2</td>
<td>Edit Instructors</td>
<td>4.4.4.8</td>
<td>Edit Classes</td>
</tr>
<tr>
<td>4.4.4.3</td>
<td>Instructors Details</td>
<td>4.4.4.9</td>
<td>Class Details</td>
</tr>
<tr>
<td>4.4.4.4</td>
<td>Edit Students</td>
<td>4.4.4.10</td>
<td>Edit Administrators</td>
</tr>
<tr>
<td>4.4.4.5</td>
<td>Student Details</td>
<td>4.4.4.11</td>
<td>Administrator Details</td>
</tr>
<tr>
<td>4.4.4.6</td>
<td>Edit Courses</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 6 displays the relationship between database tables and Administrator Interface modules.
### 4.4.4.1 Administrator Page

**Input:** None

**Session Cookies:** AdminID, SessionID

**Function:** The *AdminPage.php* script displays the Administrator Page, which consists of basic information about the administrator along with a set of options. The set of options or tools allows the administrator to maintain the system and administer other users.

**Tables:** Admin_Details

**Output:** A page containing basic information about the administrator. This page also displays the following set of options to the user: Edit Instructors, Edit Courses, Edit Students, Edit Class, Edit Administrator, Home, and Logout.

---

<table>
<thead>
<tr>
<th>Module</th>
<th>Admin_Details</th>
<th>Instructor_Details</th>
<th>Student_Courses</th>
<th>Course_Details</th>
<th>Student_Courses</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.4.4.1</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.4.4.2</td>
<td>R</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.4.4.3</td>
<td>R</td>
<td>R,A,M,D</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.4.4.4</td>
<td>R</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>4.4.4.5</td>
<td>R</td>
<td></td>
<td></td>
<td>R,A,M,D</td>
<td></td>
</tr>
<tr>
<td>4.4.4.6</td>
<td>R</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>4.4.4.7</td>
<td>R</td>
<td>R</td>
<td></td>
<td>R,A,M,D</td>
<td></td>
</tr>
<tr>
<td>4.4.4.8</td>
<td>R</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>4.4.4.9</td>
<td>R</td>
<td></td>
<td></td>
<td>R,A,M,D</td>
<td></td>
</tr>
<tr>
<td>4.4.4.10</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.4.4.11</td>
<td>R,A,M,D</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Notation: R=Read, A=Add, M=Modify, D=Delete
4.4.4.2 Edit Instructors

Input: None

Session Cookies: AdminID, SessionID

Function: The EditInstructors.php script displays the Edit Instructors page, which consists of a list of instructors in the system. Each instructor detail has an Edit link to the details for that instructor which the administrator can activate to make changes in the instructor details. If the administrator wants to add a new instructor in the system, he/she activates the Add New Instructor and the EditInstructors.php transfers the user to the Instructor Details Page.

Tables: Admin_Details, Instructor_Details

Output: A page containing basic information about all the instructors in the system. An Edit link to the Instructor Details page and a button to Add New Instructor.

4.4.4.3 Instructor Details (InstructorDetails.php, AddUpdateInstructor.php)

Input: InstructorID, First Name, MiddleIni, Last Name, Email, Phone, Department, UserID, and Password.

Session Cookies: AdminID, SessionID

Function: The InstructorDetails.php script displays the Instructor Details Page, which contains the details of an instructor. The page displays the information about the instructor if the Edit option was selected in the Edit Instructors Page, The administrator can then makes changes and activate the Update button. The AddUpdateInstructor.php script is executed and the necessary changes are made in the system. If the administrator is adding a new instructor, a blank form is displayed. After the instructor has entered the
necessary details he/she can activate the Add button and this executes the AddUpdateInstructor.php script to add the record in the system after making necessary validation.

**Tables:** Admin_Details, Instructor_Details

**Output:** A page containing basic information about the instructors, Add, Update and Delete buttons to carry out necessary functions. When the administrator activates any of the buttons, the InstructorDetails.php script checks for errors in the entered data and prompts the user, if the data entered is correct it asks for a confirmation for updating the database. If the user selects the OK option, it executes the AddUpdateInstructor.php script. If the AddUpdateInstructor.php finds an invalid entry it displays an error message and prompts the user to reenter the information in the Instructor Details Page else it update the database and displays a confirmation message. If the user selects the Cancel option at the confirmation prompt, the control goes back to the Instructor Details Page.

4.4.4.4 Edit Students

**Input:** None

**Session Cookies:** AdminID, SessionID

**Function:** The EditStudents.php script displays the Edit Students page, which consists of a search field for the administrator to look for a particular student. The administrator enters the SSN of the student he/she wants to Add/Update or Delete from the system and activates the Search button. The EditStudents.php transfers the control to the Student Details Page with the necessary information if found.

**Tables:** Admin_Details, Student_Details
**Output:** A page containing a text box to enter the SSN of the student the Administrator wants to search.

### 4.4.4.5 Student Details (StudentDetails.php, AddUpdateStudents.php)

**Input:** `StudentID, First Name, MiddleIni, Last Name, Email, Year, Major, UserID, and Password`.

**Session Cookies:** `AdminID, SessionID`

**Function:** Same as the InstructorDetails in the Administrator Interface as discussed in section 4.4.4.4 earlier.

**Tables:** `Admin_Details, Student_Details`

**Output:** Same as the InstructorDetails in the Administrator Interface as discussed in section 4.4.4.4 earlier.

### 4.4.4.6 Edit Courses

**Input:** `None`

**Session Cookies:** `AdminID, SessionID`

**Function:** The `EditCourses.php` script displays the Edit Courses page, which consists of a list of courses in the system. Each course has an `Edit` link to the details for that course which the administrator can activate to make changes in the course details. If the Administrator wants to add a new course in the system, he/she activates the `Add New Course` and the `EditCourses.php` transfers the user to the Course Details Page.

**Tables:** `Admin_Details, Course_Details`
**Output:** A page containing basic information about all the courses in the system. An *Edit* link to the course Details page and a button to *Add New Course.*

### 4.4.4.7 Course Details (CourseDetails.php, AddUpdateCourses.php)

**Input:** CourseID, CourseNumber, Course Name, Course Details, Semester, Year, Department, and Instructor

**Session Cookies:** AdminID, SessionID

**Function:** The *CourseDetails.php* script displays the Course Details Page, which contains the details of a course. The page displays the information about the course if the Edit option is selected in the Edit Course Page; the administrator can then make changes and activate the *Update* button. The *AddUpdateCourse.php* script is executed and necessary changes are made in the system. If the Flag fields of the CourseID tables are not set in the Course_Details table, it creates the corresponding tables dynamically. If the Administrator is adding a new course, a blank form is displayed allowing the administrator to enter the necessary details. He/she can activate the *Add* button and this executes the *AddUpdateCourse.php* script to add the record in the system after making necessary validation. The *AddUpdateCourse.php* script then creates four tables for the course that are as follows: CourseID, CourseIDAssignment, CourseIDRubricAssignment, and CourseIDRubric. The administrator can delete a course by activating the *Delete* button; this executes the *AddUpdateCourse.php* and checks if the class table for this course exists. If so it displays a message for the Administrator to delete the class details, first and then delete the Course Details.
Tables: Admin_Details, Instructor_Details, Course_Details, CourseID, CourseIDRubric, CourseIDAssignment, and CourseIDRubricAssignment

Output: A page containing basic information about the course, Add, Update and Delete buttons to carry out necessary functions. When the administrator activates any of the buttons, the CourseDetails.php script checks for errors in the entered data and prompts the user, if the data entered is correct it asks for a confirmation for updating the database. If the user selects the OK option, it executes the AddUpdateCourse.php script. If the AddUpdateCourse.php finds an invalid entry it displays an error message and prompts the user to reenter the information in the Course Details Page else it updates the database, creates the corresponding CourseID tables, and displays a confirmation message. If the user selects the Cancel option at the confirmation prompt, the control goes back to the Course Details Page.

4.4.4.8 Edit Classes

Input: None

Session Cookies: AdminID, SessionID

Function: The EditClass.php script displays the Edit Class page, which consists of a list of courses in the system. The administrator can select a course to edit the class information by activating the Edit Class button. This transfers him/her to the Class Details Page.

Tables: Admin_Details, Course_Details
Output: A page containing a list of courses in the system that the administrator can select to create or edit class details.

4.4.4.9 Class Details (ClassDetails.php, DeleteClass.php, AddUpdateClass.php)

Input: CourseID, StudentSSN

Session Cookies: AdminID, SessionID

Function: The ClassDetails.php script displays the Class Details Page, which contains the information about the class and the Add, Update and Delete Class buttons. The class information consists of the students that are in a particular class. The page displays a list of drop down selection boxes, which contains a list of student registered in the system. The administrator can select from the list of students and create a class with the number of student specified in the Course Details Page. The administrator can then activate the Add/Update button to execute the AddUpdateClass.php script that updates the CourseID and Student_Courses table with the class information. The administrator can activate the Delete Class button executes the DeleteClass.php script which displays the Delete Class Page. The page displays the students in the class and a Delete link to delete each student; there is also a Delete Entire Class button that the administrator can activate to delete the entire class. This executes the AddUpdateClass.php script that then deletes the class information and displays a confirmation message.

Tables: Admin_Details, Student_Details, Student_Courses, Course_Details, and CourseID

Output: A page displaying the Class Details and buttons to Add/Update and Delete class information. When the administrator activates the Add/Update buttons the
AddUpdateClass.php script updates the database and displays a confirmation message. If the administrator select the Delete Class button, it transfers to the Delete Class Page where the administrator can delete the class information. The AddUpdateClass.php script is executed when the administrator activates the Delete Student link or the Delete Entire Class button to make changes in the CourseID and Student_Courses tables; it then displays a confirmation message.

4.4.4.10 Edit Administrators

**Input:** None

**Session Cookies:** AdminID, SessionID

**Function:** Same as the Edit Instructor in the Administrator Interface as discussed in section 4.4.4.3 earlier.

**Tables:** Admin_Details

**Output:** Same as the Edit Instructor in the Administrator Interface as discussed in section 4.4.4.3 earlier.

4.4.4.5 Administrator Details (AdminDetails.php, AddUpdateAdmins.php)

**Input:** AdminID, First Name, MiddleIni, Last Name, Email, Phone, Department, UserID, and Password.

**Session Cookies:** AdminID, SessionID

**Function:** Same as the Instructor Details in the Administrator Interface as discussed in section 4.4.4.4 earlier.

**Tables:** Admin_Details
**Output:** Same as the Instructor Details in the Administrator Interface as discussed in section 4.4.4.4 earlier.

### 5. Results

The Read, Review & Assess System (RRAS) is intended to be used in the Computer Science programming courses at the Texas A&M University–Corpus Christi. The RRAS is an online system that can be used to involve students in a collaborative learning process using peer evaluation as a mode of grading course work. This system provides functionalities available in standard assignment handling systems like tools for students...
to submit assignments online, check grades, and a set of tools for an instructor to manage a course online. It also includes a special set of utilities for the students to evaluate his/her peer's work, and for the instructor to use these peer reviews to evaluate students in the course. It provides a visual interface for the administrator to maintain and troubleshoot the system in case of any problems.
6. Summary and Future Work

RRAS is developed successfully and can be implemented with the required resources to install the system. However, the current system has some disadvantages like storing the submitted assignments in a database and hence having some size restrictions. It increases the size of the database that may cause problems if the system has to support a large number of courses. This can be replaced by a file system to store the assignments in appropriate directories and deleted when required. This option requires access to a dedicated Web Server and access as a root user for the administrator. The RRAS environment can be further extended in the future to create computer-based learning and teaching system including a set of utilities for Web-based classes, online submission and grading, and plagiarism detection. It can also be integrated with other existing course management systems to provide the peer review facility. The main use of the system is in the computer science courses, so the system can be extended to compile the submitted assignment programs automatically in a server using appropriate data sets as inputs.
7. Reference


[7] ClassNet, Iowa State University Computation Center, ClassNet documentation, [Online document], (Feb 2001), Available at HTTP: http://classnet.cc.iastate.edu/help/what_is_classnet.html


[13] Catherine M. Gynn & Cable T. Green, Online Learning Center, [Online document], (Mar. 2001), Available at HTTP: http://jac.sbs.ohio-state.edu/co850learning/

[14] Lathrop High School, Computer Science Dept, Grading Rubric for Programming


8. Appendices

Appendix A  Data Dictionary of Read, Review and Access System Tables

Appendix B  CD-ROM Containing:

- MySQL database files, (MySQL Tables)
- Microsoft Word file for this report. (Report.doc)
- Source Code for RRAS Interface (rras folder)
- User Manual
- readme.txt (installation file)
APPENDIX A

1. Student_Details

This table contains the personal information of the students. This table has the following attributes:

a. SSN: Represents the social security number of the student.
b. FirstName: Represents the first name of the student.
c. MiddleIni: Represents the middle initial of the student.
d. Last Name: Represents the last name of the student.
e. Email: Represents the E-mail address of the student.
f. Year: Represents the student’s year of study.
g. Major: Represents the student’s major field of study.
h. StudentID: Represents the userid of the student to access the system.
i. Password: Represents the password of the student to login in the system.

*Primary Key: (SSN)*

2. Instructor_Details

This table contains the personal information of the instructors. This table has the following attributes:

a. SSN: Represents the social security number of the instructor.
b. FirstName: Represents the first name of the instructor.
c. MiddleIni: Represents the middle initial of the instructor.
d. Last Name: Represents the last name of the instructor.
e. Email: Represents the E-mail address of the instructor.
f. Phone: Represents the instructor’s campus phone extension number.
g. Department: Represents the department of the instructor.
h. InstructorID: Represents the userid of the instructor to access the system.
i. Password: Represents the password of the instructor to login in the system.

*Primary Key: (SSN)*

**3. Admin_Details**

This table contains the personal information of the administrators. This table has the following attributes:

a. SSN: Represents the social security number of the administrator.
b. FirstName: Represents the first name of the administrator.
c. MiddleIni: Represents the middle initial of the administrator.
d. Last Name: Represents the last name of the administrator.
e. Email: Represents the E-mail address of the administrator.
f. Phone: Represents the administrator’s campus phone extension number.
g. Department: Represents the department of administrator.
h. AdminID: Represents the userid of the administrator to access the system.
i. Password: Represents the password of the instructor to login in the system.

*Primary Key: (SSN)*

**4. Course_Details**

This table contains the detail information of the courses. Each time a new course is added, four new course related tables are created dynamically. Following are the attributes of this table:

a. CourseID: Represents the unique course identification number. This is used to reference other course tables. Related course tables are created when the administrator creates a new course.
b. CourseNumber: Represents the course number.
c. CourseName: Represents the name of the course.
d. CourseDetails: Represents the detail information for the course.
e. Semester: Represents the semester the course is offered.
f. Year: Represents the year the course is offered.
g. Department: Represents the department offering the course.
h. Instructor: Represents the SSN of instructor who is assigned this course.
i. ClassSize: Represents the number of students in the class.
j. ClassTable: Represents the flag to check for CourseID table.
k. AssignTable: Represents the flag to check for CourseIDAsgmTable table.
l. RubricTable: Represents the flag to check for CourseIDRubric table.
m. RubricTable: Represents the flag to check for CourseIDRubricTable.

Primary Key: (CourseID)

5. Student_Courses

This table contains information about the students and the course they are registered. This table is updated when the administrator creates or deletes class information. The table has the following attributes:

a. StudentSSN: Represents the student’s SSN.
b. CourseID: Represents the course that the student is registered.

6. CourseID

This table contains information about the student assignments and grades. Some of the attributes of the table are created dynamically when the instructor creates a new assignment. The table has the following attributes:

a. StudentSSN: Represents the student’s SSN.
b. Grade: Represents the course grade of the student.
c. Assignment#: Represents the submitted assignment of the student.

d. GradedAssignment#: Represents the student’s assignment graded by the instructor.

e. Grade#: Represents the grade of the student on a particular assignment.

f. SubTime#: Represents the time the student submitted an assignment.

g. SubFlag#: Represents the flag to check for submitted assignment.

*Primary Key: (StudentSSN)*

Note: # denotes the assignment number

7. CourseIDAssignment

This table contains information about the assignments created by the instructor. The table has the following attributes:

a. AssignmentNumber: Represents the assignment number.

b. DueOn: Represents date and time an assignment is due.

c. RubricSize: Represents the number of questions in the rubric for an assignment.

d. RubricFlag: Represents the flag to check for rubric.

*Primary Key: (AssignmentNumber)*

8. CourseIDRubric

This table contains information about the rubric for an assignment. The table has the following attributes:

a. AssignmentNumber: Represents the assignment number.

b. RubricNumber: Represents the rubric question number.

c. Question: Represents the rubric question.

d. MaxMarks: Represents the points for each rubric question.
9. **CourseIDRubricAssignment**

This table contains information about the peer evaluations and evaluations by the instructor on an assignment for each student. The table has the following attributes:

a. **OwnerStudentSSN:** Represents the student whose assignment is been reviewed.
b. **ReviewerStudentSSN:** Represents the student who is reviewing the assignment.
c. **AssignmentNumber:** Represents the assignment number.
e. **Rubric##:** Represents the points assigned on each rubric question by the reviewer.
f. **Comments:** Represents the comments of the reviewer on an assignment.
g. **TotalMarks:** Represents the aggregate points assigned on an assignment by the evaluator.

Note: ## denotes the rubric question numbers (maximum limit: 25).