Faculty of Computer Science and Information Technology
University Malaya
Kuala Lumpur

Problem Based Learning Information System

By

Nicholas Barau Eddy
WET 000325

Under the supervision of
Pn. Abrizah Abdullah

Under the moderation of
Assoc. Prof. Dr. Zaitun Abu Bakar

Session 2002/2003

Abstract

The Problem Based Learning information system is an Internet based system developed for use in Faculty of Computer Science and Information Technology, University Malaya, to help facilitate and manage Problem Based Learning Courses.

This system helps both lecturers and students in their Problem Based Learning related work and offers a better student and lecturer interaction. The system is also capable of managing reports, assignment submission, evaluation tools, forum capability and online grading.

The Problem Based Learning information system is developed with PHP on the Microsoft Windows NT Server platform utilizing database created and stored in MySQL. The system was developed taking in to considerations in using open-source and is believed that the system of this nature will gradually become and essential in the future.

Acknowledgement

Firstly, I would like to thank my parents for their never ending moral support and giving me guidance and advice in the course of completing this project.

My utmost gratitude goes to me project supervisor Pn. Abrizah Abdullah, for her guidance and support for my thoughts and ideas throughout the development of this project. I would also like to thank my moderator Assoc. Prof. Dr. Zaitun Abu Bakar, for her suggestions and for spending time evaluating during my VIVA.

I also wish to express my greatest gratitude to my team member Tan Chee Haw, for his cooperation and assistance in completing this project together.

Also special thanks to all lecturers who have contributed their time to fill in questionnaires and given suggestions, without their help, this project would have been made possible.

Last but not least, I wish to thank my course mates and friends for their generous sharing of knowledge and their show of support.

Table of Contents

List of Tables

List of Figures

CHAPTER 1: INTRODUCTION AND OVERVIEW

1 Introduction and Overview

1.1 Project Overview

Problem Based Learning (PBL) is a different approach to teaching and learning and is far different from the traditional educational methodology. This PBL method also simultaneously develops problem solving strategies, disciplinary knowledge bases and skills in both lecturers and students. The main goal of this Problem Based Learning information system is to help students perform with the highest level of skill and confidence to solve problems without bringing up the question of "What is the right answer the lecturer wants to see?" but instead the question "What would be the best way to logically solve the problem as if it were in the real world?".

The proposed PBL information system that is to be developed for use in University Malaya is a user-friendly system which generates strong levels of interaction among students and lecturers in University Malaya. Problem Based Learning was first implemented in Faculty of Computer Science and Information Technology (FCSIT) during the second semester in the year 2002.

This PBL information system is a web-based client which conductively and effectively use Information Technology to create an information rich environment to be shared in interactions among lecturers and students. The system is dynamic and is able to facilitate implementation and the management of problem based learning.

Problem Based Learning is an alternative method for lecturers or academic staff to instruct a course. PBL places students in the active role of problem-solvers confronted in a structured problem which mirrors real world problems. These structured problems offer students and lecturers alike to work together and promotes motivational thinking, relevance and context train of thought, high-order thinking and authenticity. The curriculum of problem based learning consists of carefully selected and designed problems that demand from the learner acquisition of critical knowledge, problem solving proficiency, self-directed learning strategies, and team participation skills. The process replicates the commonly used systemic approach to resolving problems or meeting challenges that are encountered in life and career [1].

1.2 Problem Statements

FCSIT face the following problems in the implementation and management of Problem Based Learning:

i. Organization of PBL resources in the current implemented system is not automated and management of Problem Based Learning is complicated.
ii. Lecturers often find themselves having to create triggers, lesson plans and thinking points from scratch, without the ease of a template.
iii. The collection and marking of student submission is done manually and is particularly difficult to keep track off. Results from marking are also not instantaneous and cannot be posted automatically for students to view.

iv. Peer and course evaluation sheets are still currently printed and given out manually.

v. Students are currently unable to hand up assignments, tutorial work online or their PBL journals. Submission of assignments in to pigeonholes or holders in doors is done by students when lecturers are unavailable. When lecturers are available there usually are a multitude of students crammed outside the door waiting to submit their work.

vi. Peer and course evaluation forms are still filled manually on paper by students which are difficult for lecturers to view and tally.

1.3 Project Purpose

The purpose of this project is to improve and automate the management of Problem Based Learning in FCSIT. With the proposal and eventual integration and implementation of the proposed Problem Based Learning information system, efficiency of managing Problem Based Learning is made easier in FCSIT.

1.4 Project Objective

Generally, the objective of this project is to develop a Problem Based Learning Information System which consists of 2 modules. These modules are:

i. The Administrative or Academic Staff Module

ii. The End User or Student Module

Specifically the project objectives are threefold:

i. **Database Creation** – Database creation to accommodate storage of information

ii. **Interface Creation** – Creation of a user friendly interface which takes into account the Human Computer Interaction (HCI) principles

iii. **Search Engine** – Development of a localized search engine to traverse the system to return results of PBL resources links that match the query

1.5 Project Scope

The project encompasses two groups, the administrative staff consisting of the lecturers and deans, and the second group is the students. Therefore with these two groups, there are two different main modules which are split in to sub-modules for each group.

1.5.1 The Administrative Module

This module is utilized by lecturers and deans. The following lists the functionality and capability of this module:

i. A login and password specifically to each lecturer or dean allows different privilege levels to the Problem Based Learning Information System.

ii. Lecturers are able to upload, amend and update their PBL report.

iii. Deans are given the capability of uploading, amending and updating their PBL report mean for overall management.

iv. A problem scenario sub-module is available to offer lecturers the option of uploading their actual lesson plans and triggers to be used in the classroom using the problem based learning method.

v. An evaluation sub-module is made ready to offer lecturers the capability of creating, updating and amending of evaluation tools, such as, peer evaluation forms, course evaluation forms, and other available and versatile templates to be used for various journals or teaching materials.

vi. Lecturers are able to grade student journals online with a template which preferably is able to be customized for various situations and following grading, the template is able to format and calculate automatically the percentage for student journals. Enter their current progress with the course and this allows lecturers to track the progress of their students.

vii. Links to other problem based learning resources are easily created and maintained.

1.5.2 The End User or Student Module

This module is used by students. The following lists the functionality and capability of this module:

i. A login supplied to students, allows students their own access privilege levels to the Problem Based Learning Information System.

ii. Link to PBL journals and resources are available to students to follow.

iii. Creation of journals from templates and submission of journals are done online with the PBL information system.

iv. Students are able to fill in course evaluation and peer evaluation forms online. These course evaluation and peer evaluation forms filled by students are automatically tallied by the information system.

v. The PBL information system offers forum capabilities where students are able to create new threads and post messages to better facilitate group discussions either between students themselves or between students and lecturers.

vi. All available PBL resources linked on the site is searchable by a localized search engine.

1.6 Project Importance

The initialization of this Problem Based Learning information system project signifies a leap in new teaching methods of FCSIT. By using this PBL system, students are able to develop an active role in their educational experiences while building their communication and critical thinking skills. The system is user friendly and is used to simplify and facilitate tedious processes in the faculty related to Problem Based Learning by transferring manual work to a certain way of online automation.

Lecturers likewise are able to find and use PBL resources and work very independently just by referring to the PBL information system. With this PBL information system, students are prepared in facing professional work and possible real life situations. Lecturers are able to change their traditional role from a person taking center stage to a person who the student tries to work alongside. This encourages student-lecturer interaction which is currently lacking in the traditional method. The system allows lecturers to create a problem based course which has high faced validity for students.

This is further improved by building on previous work templates to create better structured problems from archived templates on the system.

Students are able to hone their team building and communication skills to aid cooperation in group work. This would also be an opportunity for faculty's to introduce to students activities which highly likely constitute their field of study or work.

Ultimately, the management and implementation of Problem Based Learning in FCSIT is facilitated by this Problem Based Learning information system.

1.7 Hardware and Software Requirements

1.7.1 Hardware Requirements

This project requires the following components which are listed in Table 1.1.

Table 1.1: Hardware Requirements

Hardware Requirements	Minimum Requirements	Suggested Requirements
Processor	Pentium I (166 MHz)	Pentium III (500 MHz)
RAM	64MB	128MB
Disk Space	5GB	20GB
Operating System	Windows 95	Windows XP

1.7.2 Software Requirements

In the process of developing this system, the following tools and software are expected to be utilized:

i. ASP

ii. Macromedia Dreamweaver

iii. MySQL

1.8 Project Schedule

The purpose of a project schedule or timeline is to ensure the goals of the project are met and defined within an established time period. This timeline allows for adequate definition and allocation of resources for the project. A more efficient planning and schedule of work is done based on the timeline. With a Gantt chart, the progress of the project is much easier to track and monitor.

For this project, 6 milestones have been created, they are:

i. **Concept definition, research and overview** – This is where the concept is studied and presented in an overall manner. The literature review and research of the project is to be conducted within this milestone.

ii. **Requirements specification and analysis** – Requirements of the Problem Based Learning Information System is acquired and used to determine the approach necessary to fulfill the project's general and specific requirements.

iii. **System Design** – This is where the overall system design is created and problems are stated as constraints or requirements and are turned in to solutions.

iv. **Incremental Prototyping** – The development of individual modules and sub-modules for the system is done in this milestone. Each module is developed separately for greater clarity of requirements for the system.

v. **Integration and Testing** – All the individual modules are integrated in a single environment to provide a view of the near finished product. Testing and debugging is done to ensure the information system has no errors.

vi. **Documentation** – Documentation regarding the usage and creation of the Problem Based Learning Information System is done in this milestone.

The span of each milestone is shown below in Figure 1.1:

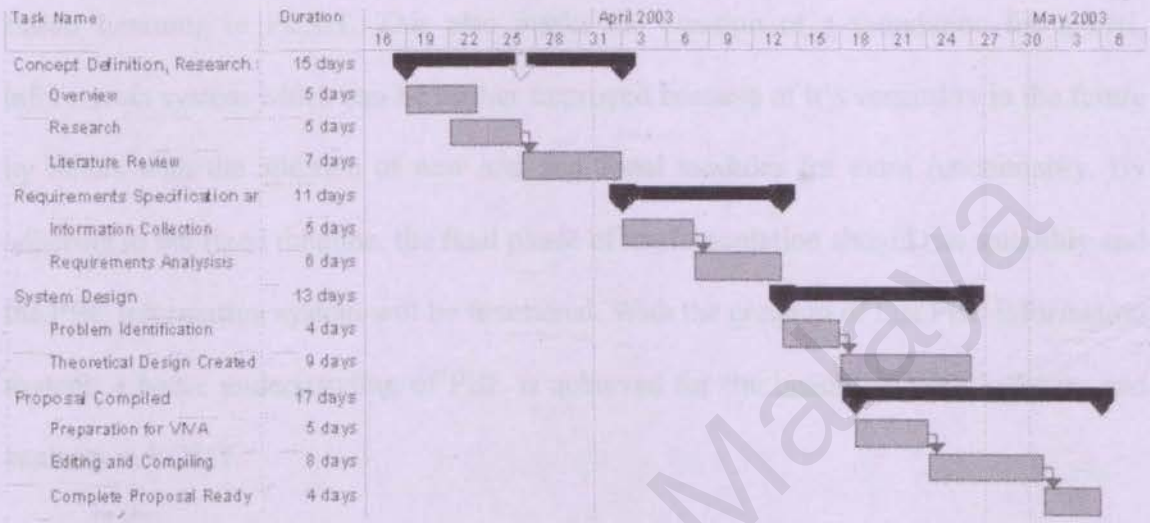


Figure 1.1: Problem Based Learning Information System Schedule, Phase 1

The second phase of the project, which is scheduled to start in the first semester of year 2003, is better shown in Figure 1.2.

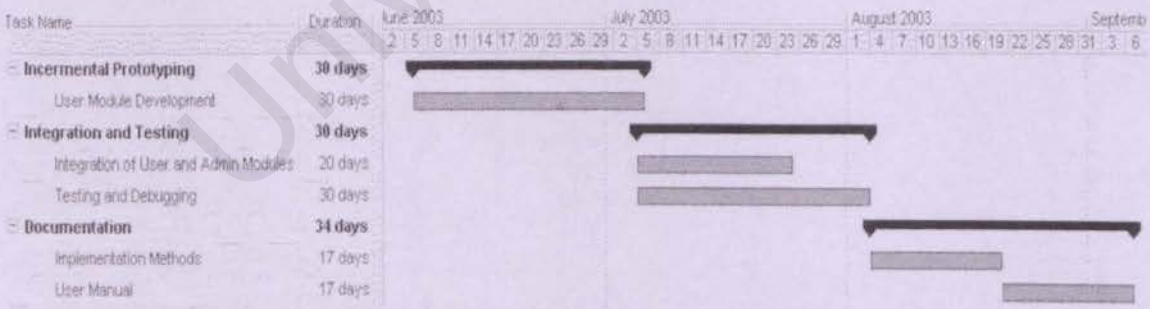


Figure 1.2: Problem Based Learning Information System Schedule, Phase 2

1.9 Summary

In summary, the development and creation of this Problem Based Learning Information System is to demonstrate the effectiveness of PBL with the ease of an information system to better facilitate management and implementation of Problem Based Learning in FCSIT. This also marks the creation of a foundation for a PBL information system which can be further improved because of it's versatility in the future by others with the addition of new and additional modules for extra functionality. By adhering to the fixed timeline, the final phase of implementation should run smoothly and the PBL information system will be functional. With the creation of this PBL information system, a better understanding of PBL is achieved for the benefit of both lecturers and students in FCSIT.

CHAPTER 2: LITERATURE REVIEW

2 Literature Review

2.1 Definition of Literature Review

A literature review is where most of thesis's and dissertations sources are cited. One must be able to locate previous search studies that have contributed to the field in a theme similar to what the thesis title proposes. The literature review section serves as a basis for further study on the said proposal. Literature reviews are usually written in an objective manner and rarely states any critical analysis but rather gathers information from results of similar studies done before. Overall the literature review gives an overall as well as an in-depth sight into the subject being studied by viewing result by other studies.

2.2 Problem Based Learning

2.2.1 What is Problem Based Learning?

As stated by Dr. Howard Barrows and Ann Kelson of Southern Illinois School of Medicine, PBL is a new approach to education and is both a curriculum and process. According to Finkle and Torp, "problem-based learning is a curriculum development and instructional system that simultaneously develops both problem-solving strategies and disciplinary knowledge bases and skills by placing students in the active role of problem-solvers confronted with an ill-structured problem that mirrors real-world problems."

The process of PBL replicates the commonly used systematic approach to resolving problems and meeting challenges that are encountered in life and career. Problem Based Learning was first used at McMaster University Medical School and is

only now slowly implemented in various undergraduate and graduate programs as well as elementary, secondary schools and colleges around the world. Students involved in PBL acquire knowledge and become proficient in problem solving, self-directed learning and team participation. This allows students to be better practitioners in their field of study [2].

The Problem Based Learning process is best described by Savery and Duffy's (1995) PBL process which states:

"The students do not know what the problem is until it is presented. As a group, students discuss the problem, generate hypotheses based on their prior experiences or knowledge, identify relevant facts in the case and identify learning issues. The learning issues are topics the group members feel are relevant to their problem and which the group feels they do not understand as well as they should.

At the conclusion of each session, each student verbally commits to a temporary position and assumes responsibility for particular learning issues that were identified. After the initial session, students engage in self-directed learning. They are responsible for gathering information pertinent to their learning issues from any available resources, much the same as any real-world physician. Assessment at the end of the process is achieved through peer and self-evaluation. These evaluations include suggestions for improvement in three areas: self-directed learning, problem solving, and skills as a group member."

Table 2.1 below shows the Problem Based Learning model done by Stepien, Gallagher and Workman in 1993.

Table 2.1: Problem-based learning model; Stepien, Gallagher, and Workman (1993).

What do we know? (Based on given case study, state the obvious facts)	What do we need to know? (From analysis of facts, determine critical information needed for problem)	What should we do? (Using analysis results, search for solutions and other alternative methods)

2.2.2 Why Problem Based Learning?

Problem Based Learning offers a change from the traditional method of instruction to a new curriculum of ill-structured possible real life problems. Below listed in Table 2.2 are the differences from traditional learning to the new problem based learning.

Table 2.2: Differences in Traditional Curricula and Problem Based Learning

Traditional Curricula	Problem Based Learning
Lecturers assume the role of an expert	Lecturers act as facilitators or mentors
Lecturers work in isolation	Lecturers works together as teams

Lecturers distribute information to students	Students take responsibility in learning and have a lecturer-student interaction
Students work alone	Students interact directly with lecturers to provide feedback on performance
Students look for the "right answer"	Students learn to look for alternative methods and make effective decisions
Grading of work is summative and the only evaluator is the lecturer	Students evaluate their own contributions as well as contributions of other members

2.3 Similar Systems

2.3.1 Real-time collaboration systems

2.3.1.2 CoDiagram and CoMMIT

These two systems as documented in a workshop position paper by Jeffrey D. Campbell are information systems that offer real-time group collaboration on diagrams. This technology can be based in the proposed PBL information system which requires students to be able to communicate with each other. The CoDiagram system receives a copy of a diagram submitted by a group member and is automatically updated on the server and clients in order to have the same information available to all members. This architecture was originally designed for a single-user application of graphically developing educational materials for the Internet using the Collaborative MultiMedia Instructional Toolkit system (CoMMIT). Both these systems allow users to create educational material to be shared amongst each other, which is partly one of the objectives of the proposed PBL information system, to share problems and educational

resources. The CoMMIT system offers similar functionality to the proposed PBL information system [3].

2.3.2 Online Assignment Submission Systems

2.3.2.1 MyInfoCom

Central Queensland University is currently using a system which allows their students to submit their assignments online. MyInfoCom is a system which is a webpage that provides services for students of Central Queensland University that are specific to each particular student's needs. One of their services is online submission of assignments.

This system follows 3 simple steps where the student finishes their assignment and saves it into proper files, the files are uploaded to a server where a database is updated to indicate the student's submission and then once the marker has graded the assignment, it is returned either by email accompanied by a mark sheet or by the Web where marks are posted. This MyInfoCom system requires students to have valid student number and password, an email address registered in the system and to be registered for related classes or courses. So far MyInfoCom has been operating since 2000 and has received over 12,200 assignments via the online submission system.

Assignments are saved in a certain format which is stated by each instructor for their courses. These assignments are labeled appropriately and when uploaded, directories for each student under different course names are created. This is to ensure easier management for markers or lecturers to sort through [4].

2.3.2.2 WEB-SUBMIT

Web-Submit is a current project being developed for efficient collection and management of electronically formatted assignments. This application is used by two parties, firstly, the faculty who wish to take advantage of an efficient means of assignment collection and redistribution and secondly, the students who are required to submit assignments electronically for their respective courses. The Web-Submit application system offers many functions such as submission and retrieval of assignments, tracking and recording activity, adding/removing courses, adding/activating/closing a course queue and many more.

This system is hosted on a publicly available server which runs Apache 1.3 as the HTTP web-server, MySQL 3.23 database and PHP 4.02 scripting which are all used to implement the Web-Submit system. The most appealing feature of this system is its availability virtually anywhere with an Internet connection. Due to its public availability on the Internet, security is a concern and all modules; database verification of users, database encryption and extra password protection is provided the relevant levels of security.

Overall, this system of submitting assignments online mirrors similarly the requirements of the PBL information system.

2.3.3 Online Resource Sharing

2.3.3.1 Kursus Online (KOL)

This is the current system being used by the Faculty of Computer Science and Information Technology (FCSIT) of University Malaya. This system is maintained and hosted by MDC. The sole purpose of this system is to allow lecturers the ability to post up their notes. Hence, students have the notes readily available anytime. By using this system, lecturers need not print and photocopy notes to be distributed to students. Usage of this system requires students and lecturers alike to have a different password each for each different subject. Students are only allowed to view and post messages in the forum area, while lecturers are able to post up notices and upload notes or assignment questions in related areas.

The functionality of a forum area allows students to indirectly communicate with each other and with the lecturer. Unfortunately the arrangement of posts and threads prove to be hard to sort through to find relevant information. Although this system allows uploading of notes and posting in the forum area, it does not however serve to be a PBL information system. The system does not allow online assignment submission or online grading of papers. It also does not offer evaluation tools whereby the lecturer can create evaluation forms for multiple purposes such as peer evaluation and course evaluation. A search engine is also unavailable to search through any resources that may be linked by lecturers. The biggest setback of this system is that each course has its own login and password and this proves to be a hassle as each course modules are difficult to manage by students and possibly lecturers who teach more than one course [5].

2.3.4 Online Grading Submission Systems

2.3.4.1 GradeBook

This is a Online Grading System used by UC Davis to keep track and calculate student grades. Its one particular feature of automating grade calculations eliminates busy work for their lecturers. It also offers interactive features that allow the lecturer to communicate grades to students.

Their system is completely integrated with their student information system; hence, it can retrieve course, student and grade information in a dynamic, real-time fashion. This allows the student list and course list to be up-to-date and grades can be submitted online. The GradeBook system allows students to track confidentially their progress by viewing their individual scores, class rank, score distribution and maximum and minimum scores.

The system allows as well for lecturers to customize the categories of their course and give appropriate weight to each category (e.g. course work, tutorials, mid-semester exams, final exams). The GradeBook also allows multiple users to access the same GradeBook system in the event of multiple lecturers teaching one same course [6].

2.3.4.2 E-Grades

E-Grades is the currently used system for online grade submission at UC Berkeley's. With this system, lecturers at UC Berkeley are able to submit grades into the

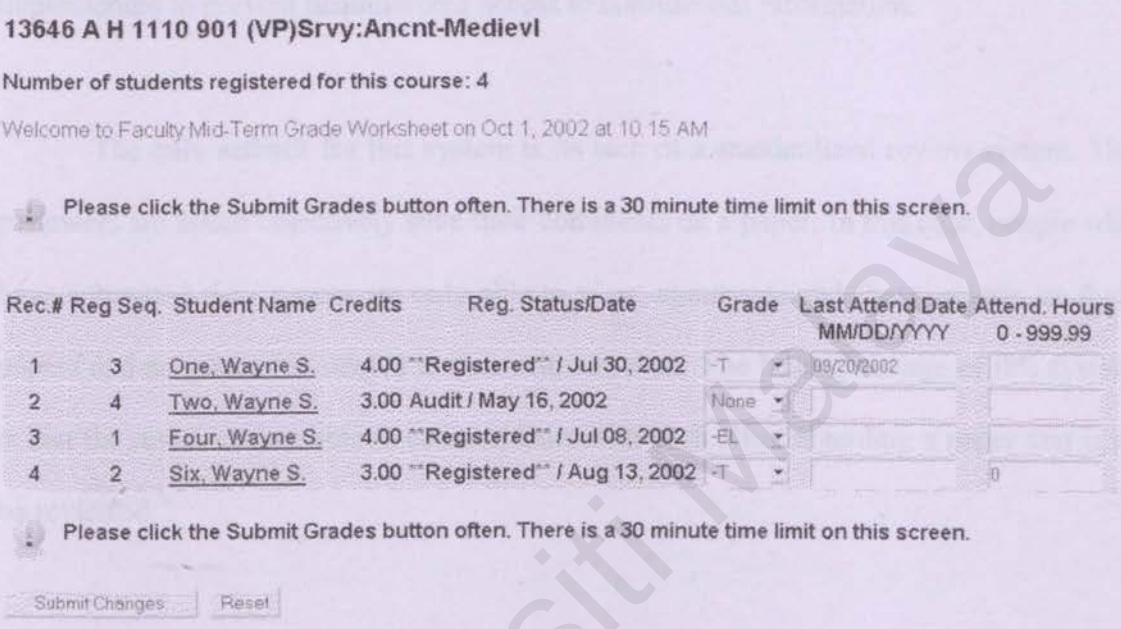
academic record so their students are able to view their respective grades the next day. With this application, the need for pre-printed course reports are removed and offers students easy accessibility to their grades. It also allows grades to be directly imported from Excel files into the system.

E-Grades is a very secure system whereby the system logs off any user who has been logged on for 90 minutes. To continue with using the system the user has to log in again. All grades that have been saved before the 90 minutes are kept but grades that have not been saved are not kept. This ensures that grades are valid and up-to-date.

The system is able to call up a class list, emails for the class, assign grades for each student, upload grades from an Excel sheet and assign other lecturers to input grades for students. Only students whose names who are on the class list can be graded. This is to ensure students that are graded have successfully registered for the class or course and are confirmed by the university to be in the class.

The system shows a list of student names and next to their names are placed action drop down menu's consisting of grades. The grades are selected and then the lecturer proceeds to the next page. This action automatically saves previous entry of grades for students. Lecturers are still able to change the grade on the "View Grades" page.

The system also allows for a grade roster to be created into Excel format. This allows for easier printout and maybe handing in of a hardcopy for management purposes [7]. Figure 2.1 shows a screenshot from this system.



13646 A H 1110 901 (VP)Srvy:Ancnt-Medievl

Number of students registered for this course: 4

Welcome to Faculty Mid-Term Grade Worksheet on Oct 1, 2002 at 10:15 AM

- Please click the Submit Grades button often. There is a 30 minute time limit on this screen.

Rec.#	Reg Seq.	Student Name	Credits	Reg. Status/Date	Grade	Last Attend Date MM/DD/YYYY	Attend. Hours 0 - 999.99
1	3	One, Wayne S.	4.00	**Registered** / Jul 30, 2002	-T	09/20/2002	
2	4	Two, Wayne S.	3.00	Audit / May 16, 2002	None		
3	1	Four, Wayne S.	4.00	**Registered** / Jul 08, 2002	-EL		
4	2	Six, Wayne S.	3.00	**Registered** / Aug 13, 2002	-T		0

- Please click the Submit Grades button often. There is a 30 minute time limit on this screen.

Submit Changes Reset

Figure2.1: E-Grades System Screenshot

2.3.5 Applicable Systems

2.3.5.1 Precision Conference Solutions: Online Submission and Review Systems

Precision Conference Solutions offers an online submission and review system of papers and supporting electronic materials. Although this is not considered to be a information system or a grading system of sorts, but its purpose can be applied to the PBL information system.

This system allows papers and electronic materials to be submitted online. It also handles online distribution of papers to reviewers and the online collection of reviews. Reviewers are picked by an administrator or committee member and assign to certain papers that are to be reviewed. Since this system is available online, security features are implemented to prevent unauthorized access to confidential information.

The only setback for this system is its lack of a standardized review system. The reviewers are asked objectively state their comments on a paper. In this case, people who have submitted their papers are only able to view, comments and short reviews on their papers and no grading is related to the review system. The best advantage of this system is that the reviewers are able to communicate with each other regarding a paper that is to be reviewed [8].

These functions are also applicable in the PBL information system. Figure 2.2 shows a screenshot where reviewers can edit and enter their reviews on a paper and submit it online.

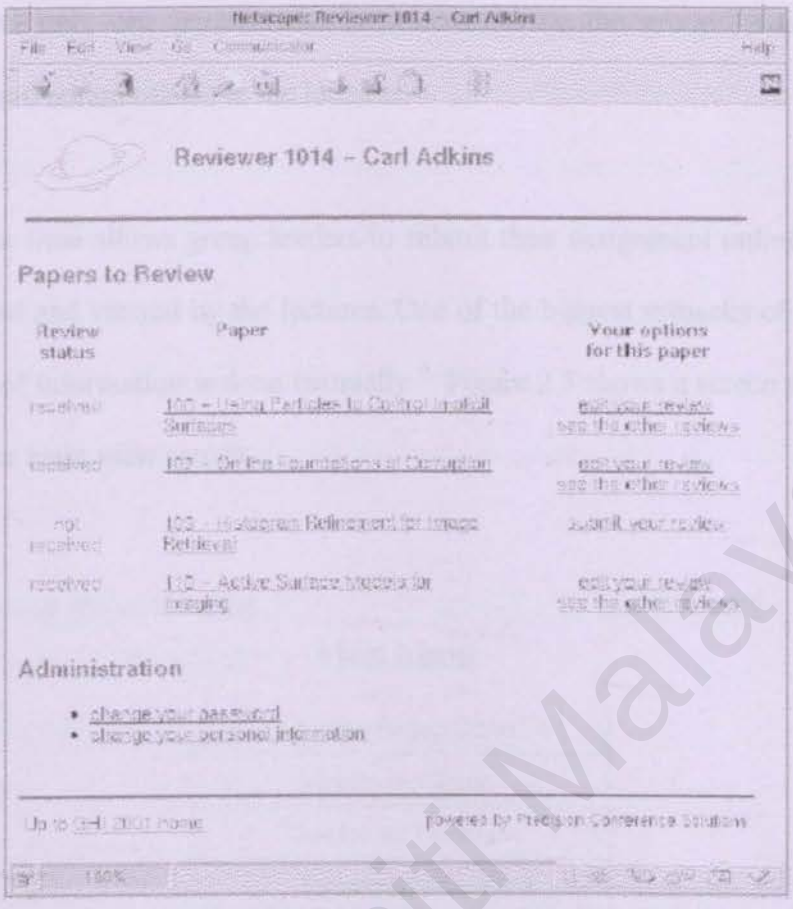


Figure 2.2: Precision Conference Solutions System Screenshot

2.3.5.2 Pesaka Ilmu

Pesaka Ilmu is an academic based electronic content tracking and publication system. The goal of Pesaka Ilmu is to track students' theses or project progress. This system allows monitoring of group work. The system allows interaction between students and lecturers and tutorials and note via an Internet enabled e-book.

Although the system is still fairly new and the ideas and functionality it has implemented are quite good, it still needs to be improved. The system allows the creation

of groups, but only one person who is selected to be the group leader can submit assignments and communicate to the lecturer

Pesaka Ilmu allows group leaders to submit their assignment online where it can be downloaded and viewed by the lecturer. One of the biggest setbacks of this system is that all entry of information is done manually [9]. Figure 2.3 shows a screen shot of Pesaka Ilmu's lecturer main view screen.

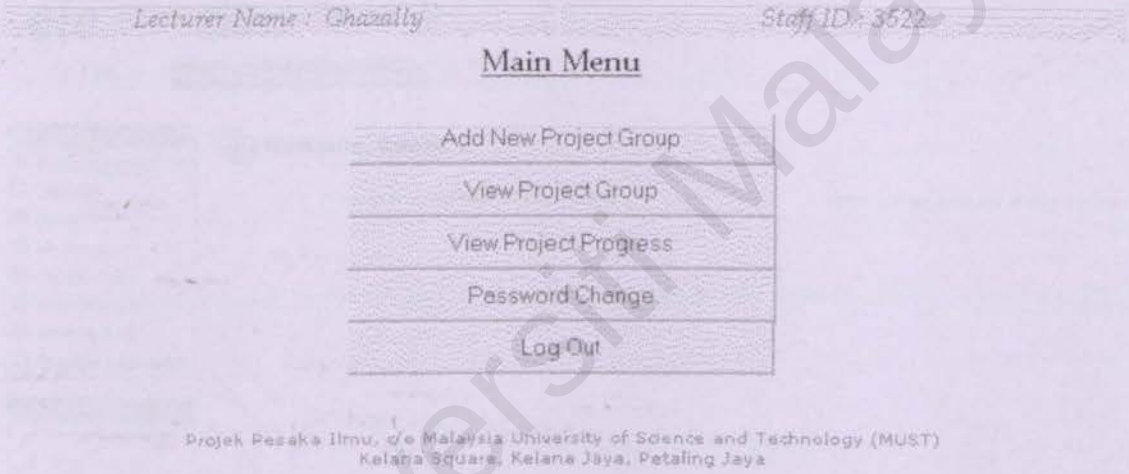


Figure 2.3: Pesaka Ilmu Screenshot

2.3.5.3 Electronic Learning Environment of University of Maastricht (ELEUM)

ELEUM is the system powered by Blackboard5 used by University of Maastricht. This is a system to facilitate Problem Based Learning in the University of Maastricht. This system offers an easy to use interface and has functions such as discussion area, a calendar feature and a page where students can view their marks. Unfortunately this system does not allow online assignment submission.

An analysis of the system shows an overall consistent layout that makes the page very easy to navigate. The ELEUM system also offers students the capability of browsing through a course catalog to register themselves for courses that are supported by the system. Analysis of this system shows that the functions offered do facilitate Problem Based Learning in the University of Maastricht [10]. The ideas and interface design from ELEUM can be adapted for use in the Problem Based Learning information system of FCSIT. Figure 2.4 shows a screenshot of the main page viewed by users once logged in.

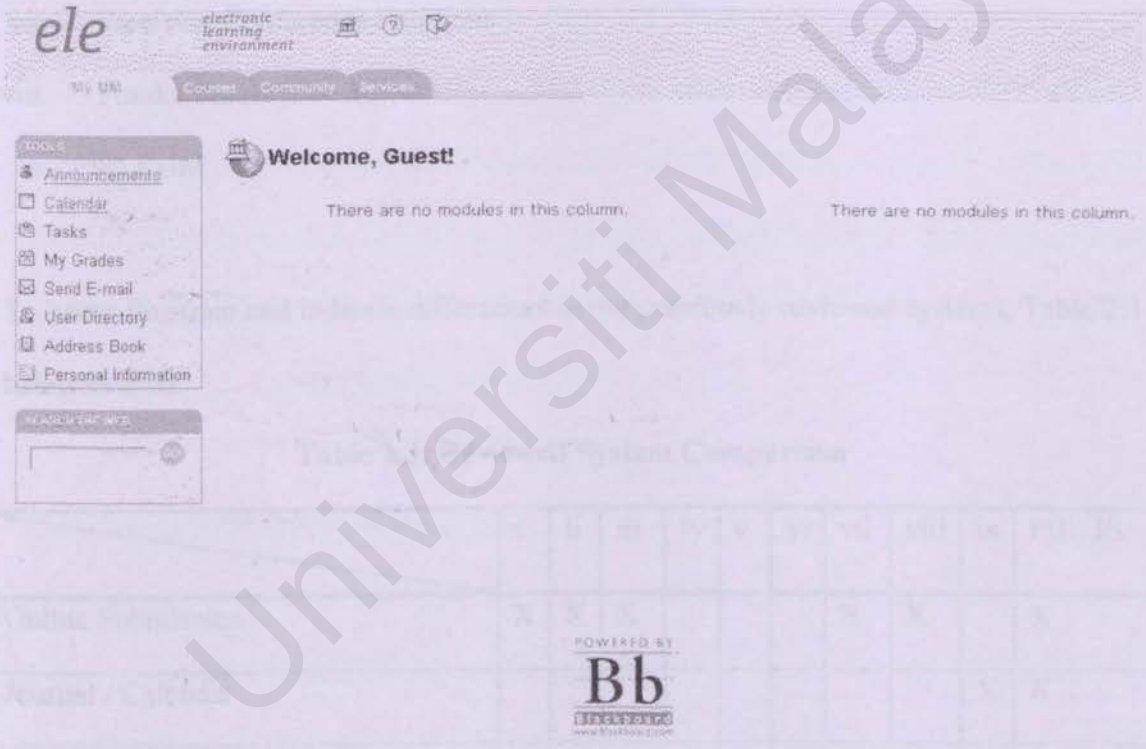


Figure 2.4: Electronic Learning Environment of University of Maastricht Screenshot

2.4 Comparison and Analysis of Reviewed Systems

The systems that have been reviewed are:

i. CoDiagram & CoMMIT

ii. MyInfoCom

iii. WebSubmit

iv. MDC Kursus On-Line

v. GradeBook

vi. E-Grades

vii. Precision Conference Solutions

viii. Pesaka Ilmu

ix. ELEUM

To better illustrate and indicate differences in the previously reviewed systems, Table 2.3 below is used.

Table 2.3: Reviewed System Comparison

	i	ii	iii	Iv	v	vi	vii	viii	ix	PBL IS
Online Submission	X	X	X				X	X		X
Journal / Calendar									X	X
Discussion Area	X			X			X		X	X
Online Grading					X	X	X		X	X
Evaluation Tools										X
Online System		X	X	X	X	X	X	X	X	X

From the analysis and comparison of each system, selected components or functions from various similar systems can be adapted in to the Problem Based Learning information system of FCSIT.

2.5 Problem Based Learning Information System Function and Purpose

All the previously reviewed systems partly offer similar functionality to the proposed PBL information system.

2.5.1 Submission, Amendment and Updating of PBL Reports

Faculty members are given the option of uploading their respective PBL reports to the main system for central management use. These reports are easily updated or amended. Students are also allowed to only view their respective reports but are not given the privilege of editing their reports. The viewing of reports is all done online, therefore giving students easy access through the Internet.

2.5.2 Creation of Evaluation Templates

Faculty members are able to create evaluation templates, either for peer evaluation, course evaluation or work evaluation. These evaluation sheets are all accessible through the Internet. Students can view the peer evaluation and course evaluation sheets and can enter respective marks or grades. The grades or marks for each sheet is automatically tallied, calculated and uploaded to a database. These completed sheets can be viewed and studied by the faculty member concerned to review his/her course standard. The peer evaluation sheets give an overall view on work done by other

group members to ensure students are doing their work accordingly. Work evaluation sheets are used by the lecturer to assess assignments, reports or journals submitted by students.

2.5.3 Searchable Problem Based Learning Resources

Resources that are linked or created by faculty members are searchable by students for reference. These resources are easily linked by means of a template and if possible verified to ensure the resource is still available over time.

2.5.4 Submission of Reports, Assignments and Papers

Students are able to submit their work online, creating a near paperless environment. This allows students to upload their work and receive notification of the complete upload procedure. Hence, this work is easily sorted and marked by faculty members who can then submit their grades online and these grades are viewed by students online.

2.5.5 Discussion or Forum Area

A discussion or forum area is available for students to communicate with each other regarding their work or post questions directed to faculty members. This is to ensure ease of communication if by any chance students are unable to be physically present for a group discussion. The forum area offers the creation of different threads which can be named appropriately and used for related work.

2.6 Summary

With all the above mentioned related systems, we can actually build a hybrid of all the systems to be used as a Problem Based Learning information system. The study of other systems gives a view of all possible problems that can be encountered and ways to overcome them. The literature review does not discuss available technology for the implementation of the PBL information system.

CHAPTER 3: METHODOLOGY AND SYSTEM ANALYSIS

3 Methodology and System Analysis

3.1 Project Prototyping

The software prototyping methodology is used due to the fact that is allows the entire system to be constructed quickly and easily. This assists in the understanding and clarifying requirements and objectives and reduces the risk of failure in the development of the Problem Based Learning information system.

Prototyping is a technique for building a quick and rough version of a desired system or part of that system. The prototype shows the system to users and designers what it would or could possibly look like and function. It allows the identification of flaws and creates avenues to improve the system. It allows the system designer to collaborate with users to review the designed system.

Listed below are the advantages of using the prototyping model [11]:

i. Prototyping encourages and requires active end-user participation. This increases end-user morale and support for the project. End-user morale is enhanced because the system appears real to the end-users.

ii. Iteration and change are a natural consequence of systems development because end-users tend to change their minds. Prototyping better fits this natural situation because it assumes that a prototype evolves.

iii. The assumption that end-users are not able to specifically identify all their requirements until they see the system, allows prototyping to be a flexible method.

iv. An approved prototype is a working equivalent to a paper design specification and allows flaws or errors to be detected much earlier.

v. Prototyping allows quicker user feedback.

Even with all these advantages, prototyping has its own disadvantages. These disadvantages are listed below [11]:

i. Prototyping does not eliminate the need for the system analysis phases

ii. Substitution of a paper specification and design cannot be fully done with prototyping.

iii. Prototyping often leads to premature commitment to a design.

iv. Complexity and scope of a system can expand beyond original plans.

v. Prototyping can reduce creativity in designs, meaning certain solutions are overlooked during implementation.

As with all other software development, prototyping begins with requirement gathering. After the identification of requirements, a design is formulated. It focuses on the top level architecture and the data description issues. This in turn creates the prototype. The prototype is tested and evaluated to refine the requirements and fix whatever flaws that maybe encountered. The prototyping model consists of 6 steps and is shown in Figure 3.1:

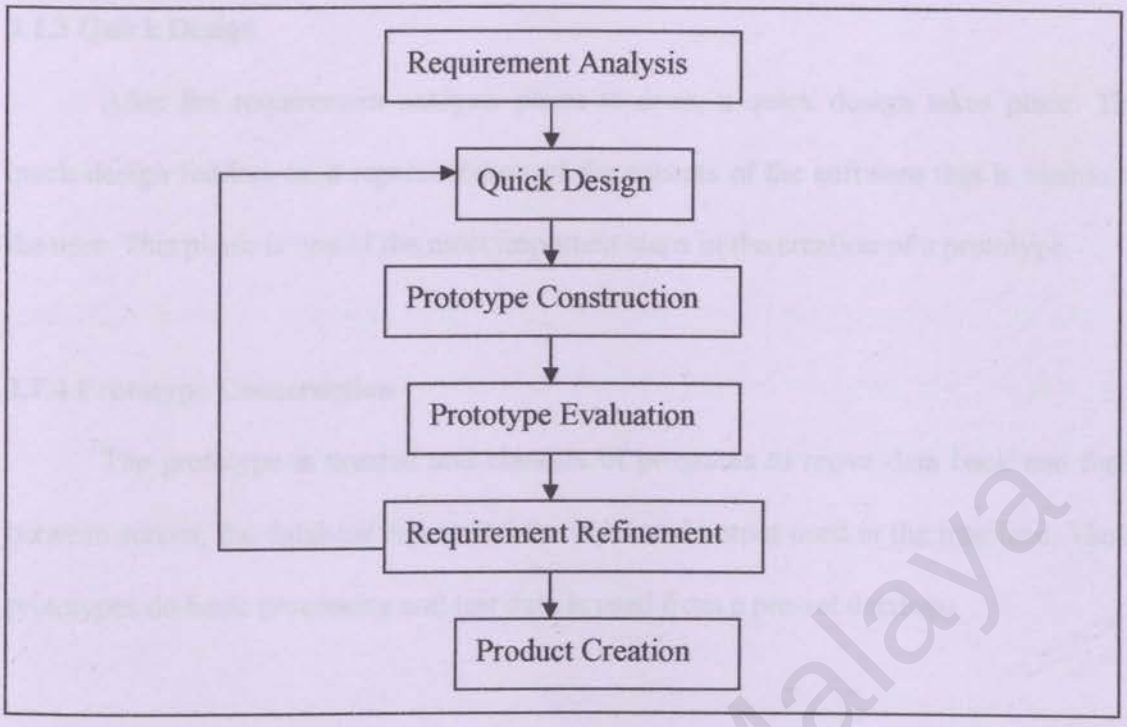


Figure 3.1 Prototyping Model

3.1.2 Requirement Analysis

Prototyping begins with requirement gathering and analysis. In this step, the developer and user meet and define the overall objectives for the software and identify requirements that are needed and outline areas where further definition is mandatory. End-user interviews and questionnaires are given out during this step. Feedback from the questionnaires pinpoint that Problem Based Learning report management and creation is the most important factor in the system.

3.1.3 Quick Design

After the requirement analysis phase is done, a quick design takes place. The quick design focuses on a representation of the aspects of the software that is visible to the user. This phase is one of the most important steps in the creation of a prototype.

3.1.4 Prototype Construction

The prototype is created and consists of programs to move data back and forth between screen, the database report and the input and output used in the interface. These prototypes do basic processing and test data is used from a pre-set database.

3.1.5 Prototype Evaluation

The user or owner evaluates the tested design prototype. They are allowed to test, report flaws and suggest improvement. Users are allowed to have a hands-on trial of the prototype and evaluate the prototype.

3.1.6 Requirement Refinement

The prototype evaluation step is used to redefine the requirements for the system that is to be developed. A process of re-iteration happens as the prototype is improved to satisfy the needs of the users. New requirements can be added as seen fit.

3.1.7 Product Creation

This is when all the requirements are met and the engineering of the actual product can be put together.

3.2 Project Life Cycle

The software engineering process consists of a set of steps that encompass methods, tools and procedures. These steps are often referred to as software life cycle models. A model for software engineering is chosen based on the nature of the project and applications, the methods and tools that are to be used and the controls and deliverables that are required [12]. For this project, the waterfall method or life cycle is utilized. In Figure 3.2, the waterfall method is graphically shown in a series of steps.

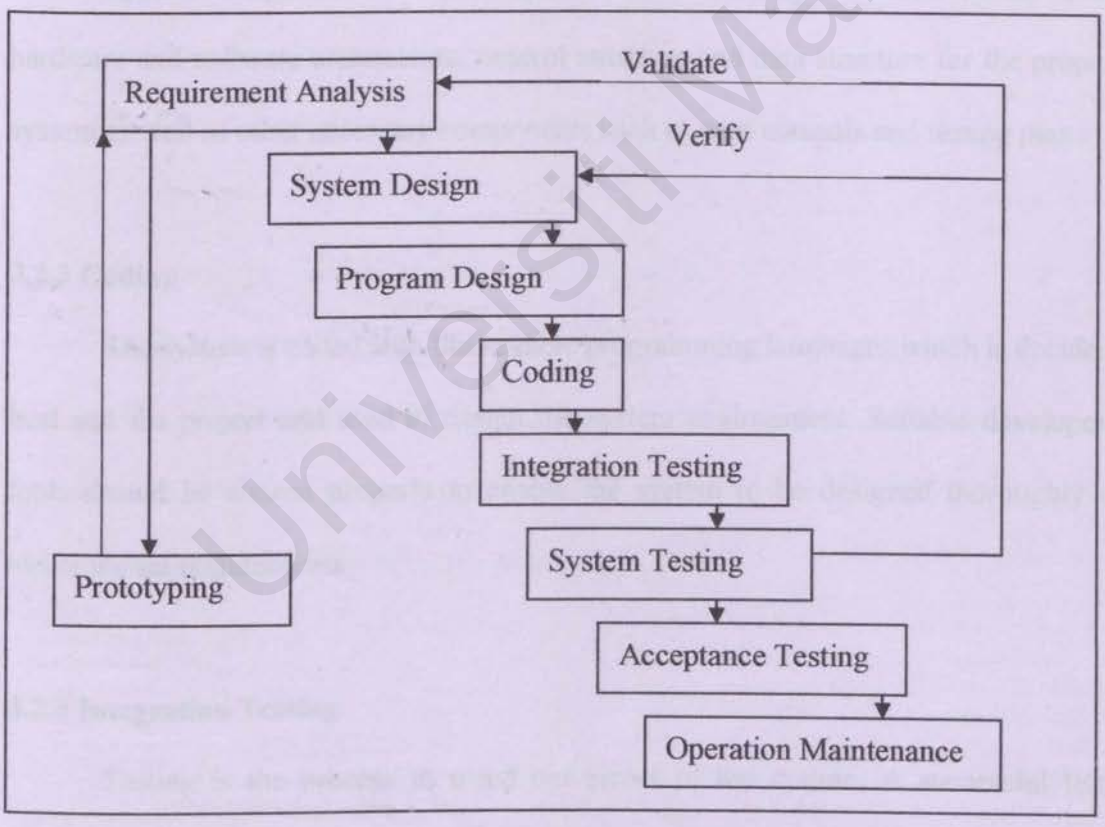


Figure 3.2: The Waterfall Model

3.2.1 Requirement Analysis

The first process in the entire methodology process is the Requirement Analysis. It mentions about verified specification of the required functions, interfaces and performance for the system. To determine the requirement of the system, gathering information is important. There are a variety of technologies that can be used which includes sampling and investigation, data, interviewing, using questionnaires, prototyping and observations. Different techniques are employed depending on the situation.

3.2.2 System Design

System design involves a complete and verified specification of the overall hardware and software architecture, control structure and data structure for the proposed system as well as other necessary components such as user manuals and testing plans.

3.2.3 Coding

The system is coded with the suitable programming languages which is decided to best suit the project and used to design the system environment. Suitable development tools should be chosen properly to enable the system to be designed thoroughly and meets the set requirements

3.2.4 Integration Testing

Testing is the process to weed out errors of the system. A successful test is considered to be a test that establishes the presence of one or more errors or bugs in the

system which is undergoing the test. When all prototype modules have been tested, the compilation and integration of all modules is done to produce the final system.

3.3 Information Gathering Techniques

Information gathering serves as a method to outline and help identify requirements needed for the development of the proposed Problem Based Learning information system. The information gathering techniques that have been utilized in the development of the system is listed below:

i. Documentation and related system review
ii. Internet
iii. Interview & Questionnaire

3.3.1 Documentation and Related System Review

Many information systems developed are based, partially or fully, on existing systems. Therefore the review of existing systems and their documentation is where the first facts are found. The researcher is able to identify good characteristics of other systems and adapt it to the proposed system.

3.3.2 Internet

Traversing the Internet for related and up-to-date technologies proves to be very worthwhile. The Internet offers a wide range of information pertaining to available technology and their latest improvements. Through the Internet, previous review or

documentation can be retrieved and this method works well with the technique of documentation and related system review.

3.3.3 Interview and Questionnaire

Interviews are carried out with end-users who are directly involved with Problem Based Learning in FCSIT. Interviewing is used to further identify system requirements and pin-point areas of development that need to be stressed. A questionnaire is also created to be distributed to end-users who are unavailable for interview to gain their suggestions and requirements for the system.

3.4 Requirements Analysis

The initial task of the requirements analysis phase is to identify requirements. Actually, the objectives of the information system serve as a basis to outline the functional and nonfunctional requirements that are needed to develop the Problem Based Learning Information System.

The requirements for any system are divided in to two parts, the functional requirements and the nonfunctional requirements.

A functional requirement is a description of activities and services the system must provide. Functional requirements are frequently identified in terms of inputs, outputs, processes and stored data that are needed to satisfy the system objectives.

A nonfunctional requirement is a description of other features, characteristics and constraints that sums up to a satisfactory system. Nonfunctional requirements would cover requirements such as performance, ease of use, user-friendliness, security, quality management and such.

3.4.1 Functional Requirements

The functional requirements cover the operation ability of all the modules. As stated in the project overview, the two modules to be covered are Administration Module and the Student Module.

3.4.2 Administration Module Functional Requirements

The administration module can be divided into three separate sub modules which are the Lecturer sub module, the Dean sub-module and the overall Administration Module. Each of these modules has their own functionality and can be better shown graphically in figure 3.3, figure 3.4 and figure 3.5. Some of the functionality from each module may overlap one another.

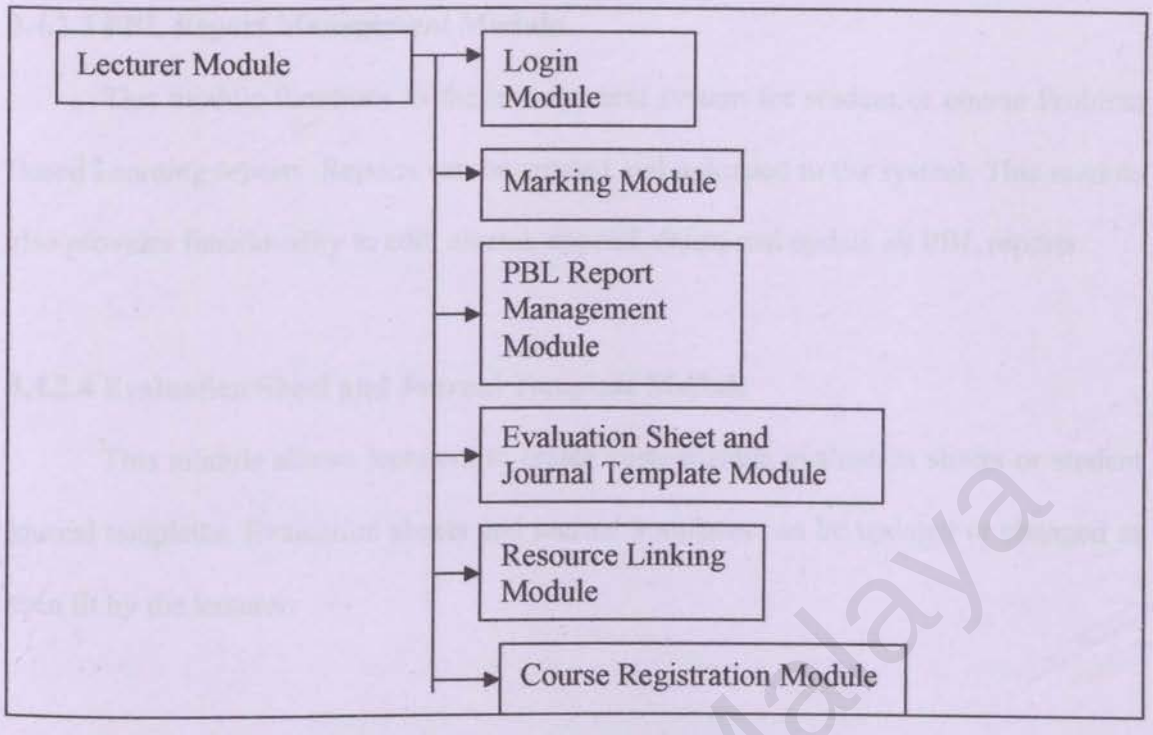


Figure 3.3: Lecturer Module and Sub Modules

3.4.2.1 Login Module

This module facilitates and verifies the login and password access for lecturers.

3.4.2.2 Marking Module

This module automates the calculation of marks and facilitates the ease of marks and their entry into the system by the lecturer. Marks are automatically tallied for assignments and student journals.

3.4.2.3 PBL Report Management Module

This module functions as the management system for student or course Problem Based Learning reports. Reports can be created and uploaded to the system. This module also provides functionality to edit, amend, append, delete and update all PBL reports.

3.4.2.4 Evaluation Sheet and Journal Template Module

This module allows lecturers to create customizable evaluation sheets or student journal templates. Evaluation sheets and journal templates can be updated or changed as seen fit by the lecturer.

3.4.2.5 Resource Linking Module

The creation of links to resources is easily maintained and managed with this module. This module facilitates the ease of creating these links.

3.4.2.6 Course Registration Module

This module lets lecturers involved in PBL courses to register their course in to system. This facilitates the tracking of how many lecturers are involved in PBL courses and allow students to view a list of PBL courses available.

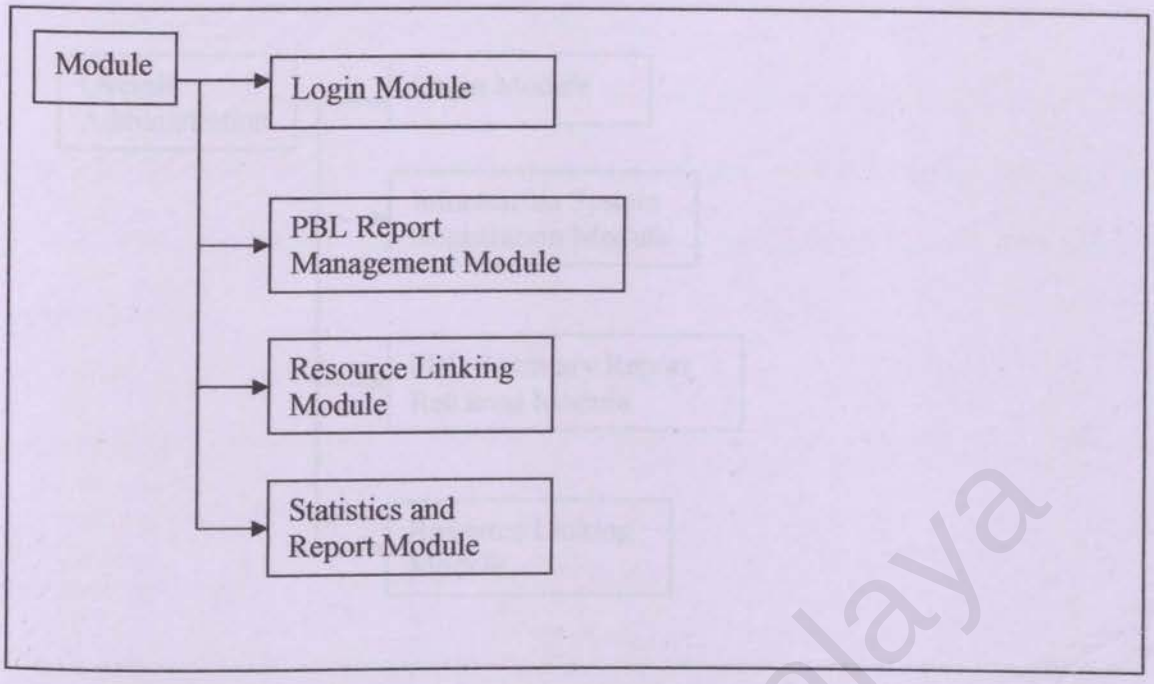


Figure 3.4: Module and Sub Modules

3.4.2.6 Statistic and Report Module

The automatic tally and report creation of the number of lecturers are currently registered in the system is automated here. This also includes the statistics of student usage of the system.

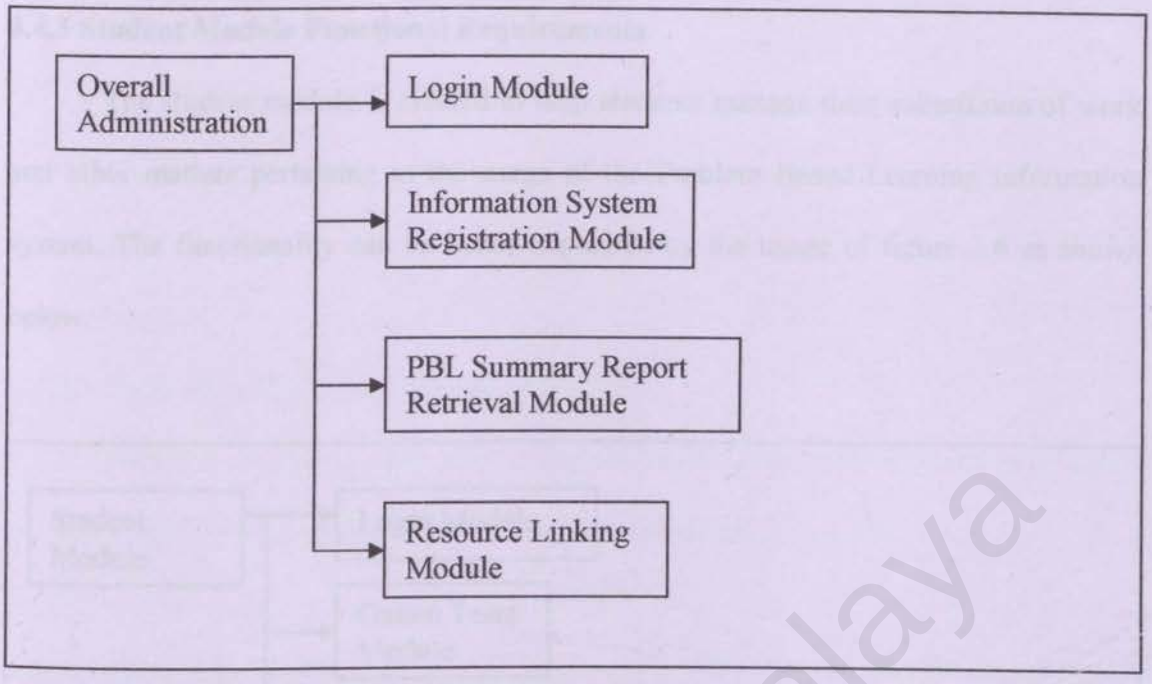


Figure 3.5: Overall Administration Module

3.4.2.7 Information System Registration Module

This allows the functionality of registering certain courses and registration of students and lecturers in the system. It also explains the courses which are already registered in the system for students to view what PBL courses are available.

3.4.2.8 PBL Summary Report Retrieval Module

The module allows students and lecturers alike to view the PBL reports for students and courses. Restrictions for viewing are set in user privileges and students can only view while lecturers can edit and amend.

3.4.3 Student Module Functional Requirements

The student module is created to help students manage their submission of work and other matters pertaining to the usage of the Problem Based Learning information system. The functionality can be better explained by the usage of figure 3.6 as shown below.

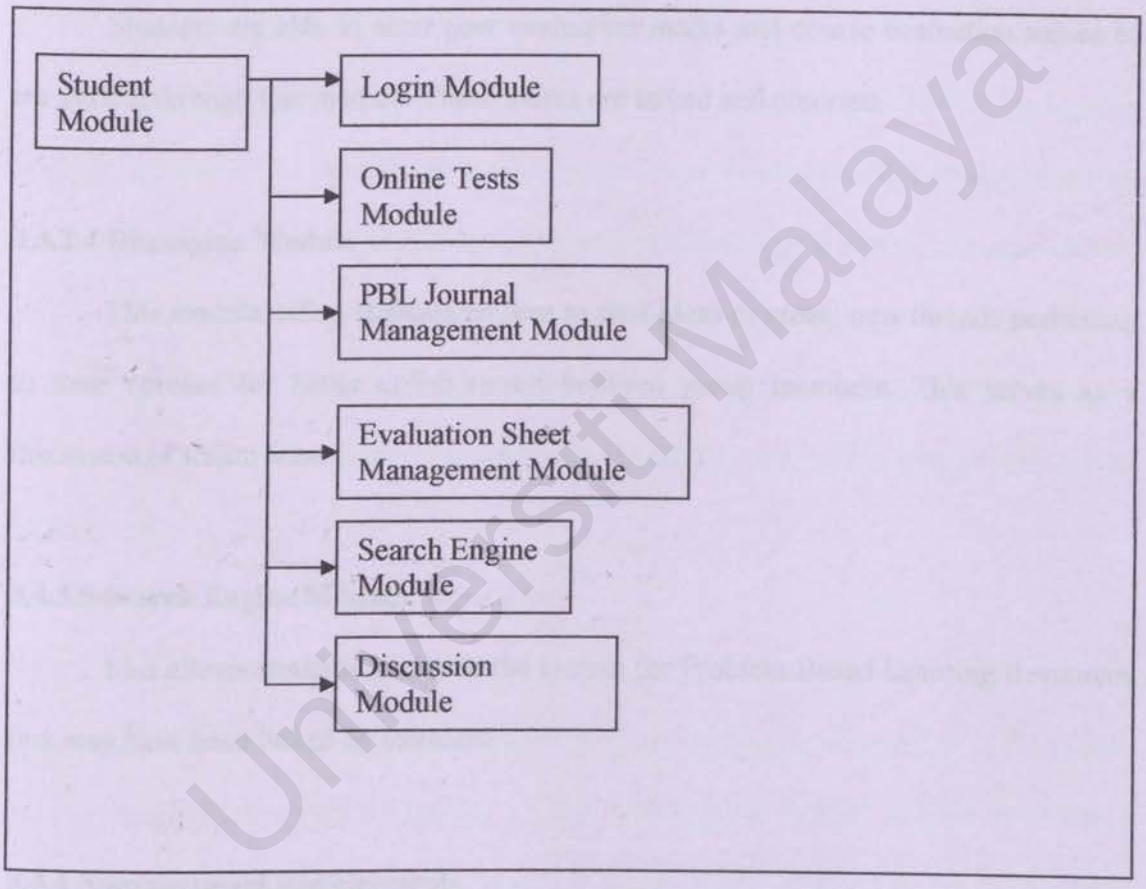


Figure 3.6 Student Module

3.4.3.1 Online Tests Module

This module enables students to complete simple test questions online and is given a mark on the spot. The online test is set by lecturers.

3.4.3.2 PBL Journal Management Module

The functionality of creating student journals, editing the journals and submitting journals are done by this module. This allows students to regularly report their progress in a course.

3.4.3.3 Evaluation Sheet Management Module

Students are able to enter peer evaluation marks and course evaluation values to the system through this module. These marks are tallied and reported.

3.4.3.4 Discussion Module

This module offers students an area to post ideas or create new threads pertaining to their courses for better collaboration between group members. This serves as a discussion or forum area.

3.4.3.5 Search Engine Module

This allows students to search the system for Problem Based Learning Resources that may have been linked by lecturers.

3.4.4 Nonfunctional Requirements

The nonfunctional requirements for the proposed Problem Based Learning information system are:

i. User-friendly and Usability

ii. Reliability

iii. Maintainability and Expandability

iv. Modularity

v. Response Time

vi. Security

3.4.4.1 User-friendly and Usability

The system can be considered as an attractive or as an easy to use application because the users only have to operate a Graphic User Interface (GUI) which is easy to navigate. The user of the Problem Based Learning information system is more confident in using the system due to the usage of suitable icons or aptly named buttons and text boxes. Any processes which cover the updating or deletion of records from a database are accompanied with a re-confirmation message to curb the irreversible loss of data due to human error.

3.4.4.2 Reliability

If the system does not fail or does not stop functioning when used in the proper manner, it is considered to be a reliable system. The Problem Based Learning information system must be developed to be reliable as it is used by most of the students and lecturers of the Faculty of Computer Science and Information Technology of University Malaya. It is crucial in maintenance and operation, as frequent breakdowns of the system increases the cost of maintenance and debugging of the system. Therefore it is expected of the system to run reliably.

3.4.4.3 Maintainability and Expandability

The system designed must be easily understood, corrected, updatable and able to be enhanced without much difficulty. A system is considered easily maintained and easily expanded if its architecture components, algorithm, data structure and processes are able to be expanded and modified with ease.

3.4.4.4 Modularity

The system architecture should embody modularity, which means the system is divided into separately named and addressable modules and sub modules. In the case of the proposed Problem Based Learning information system, the modules are the Administration Module and the Student Module. This is done to isolate the functionality from each module.

3.4.4.5 Response Time

The response time to retrieve the data and information contain within the Problem Based Learning information system should be within a reasonable amount of time. This means, all desired data or information required by the user must be available in a acceptable time period.

3.4.4.6 Security

The security of such a system as the Problem Based Learning information system is of the utmost importance. This system is available publicly. Therefore, implementation

of a user login and password is required to facilitate the security of the system. A different user privilege is given to different users characterized by their modules.

3.5 Web Programming Languages

Web programming languages can be divided in to two categories, static and dynamic. The following sub sections outlines with detail the available web programming languages. The static languages that are covered are HTML, and the dynamic languages that are covered are PHP, ASP and other related dynamic languages.

3.5.1 Static Web Programming Languages

3.5.1.1 HTML

Hypertext Mark-Up Language (HTML) is the language used to create nearly all WWW documents. HTML allows the creation of formatted text, including images on pages and adds links from one document to another.

HTML is a simple subset of the Standard Generalization Mark-Up Language (SGML). Specifically, HTML is a Document Type Definition which defines precisely descriptive elements from a specific type of document.

As a subset of SGML, HTML is not concerned with the format of a document, instead it describes a documents logical structure leaving it up to the client side to render the document as desired by the user.

The HTML is the primary language of documents served by web servers. It provides a rich and growing set of tags that are embedded in documents to specify how the content should be formatted on a page. These tags enable establishment of hypertext links from content in one document to content in other documents. HTML also provides mechanisms for invoking programs and services on web servers.

3.5.2 Dynamic Web Programming Languages

3.5.2.1 PHP

PHP, which stands for "PHP: Hypertext Preprocessor" is a widely-used Open Source general-purpose scripting language that is especially suited for Web development and can be embedded into HTML. Its syntax draws upon C, Java, and Perl, and is easy to learn. The main goal of the language is to allow web developers to write dynamically generated web pages quickly.

What distinguishes PHP from client-side languages such as JavaScript is that the code is executed on the server. Hence, the underlying code looks far different from normal HTML and therefore users are not be able to discern the function or true purpose of the code. The best thing in PHP is that it is rather simple to learn for a newcomer but still offers many advanced features for a professional programmer.

PHP is mainly focused on server-side scripting, so you can do anything any other CGI program can do, such as collect form data, generate dynamic page content, or send and receive cookies. The three main fields where PHP scripts are used are as follows:

i. Server-side scripting - This is the most traditional and main target field for PHP. 3 things are needed to make this work; the PHP parser, a web server and a web browser.

ii. Command line scripting - PHP script can be executed without any server or browser. Only the PHP parser is needed. This sort of usage is ideal for scripts regularly executed using cron (available on Linux) or Task Scheduler (on Windows). These scripts can also be used for simple text processing tasks.

iii. Writing client-side GUI - PHP would not be the best language to write windowing applications, but PHP offers advanced features that can be used with PHP-GTK to be able to write cross-platform applications. PHP-GTK is an extension to PHP.

PHP can be used on all major operating systems, including Linux, many UNIX variants, Microsoft Windows, Mac OS, and RISC OS. PHP has also support for most of the web servers today. This includes Apache, Microsoft Internet Information Server, Personal Web Server, Netscape and iPlanet servers, Oreilly Website Pro server, Caudium, Xitami, OmniHTTPd, and many others. For the majority of the servers PHP has a module, for the others supporting the CGI standard, PHP can work as a CGI processor. Hence, with PHP one can have the freedom of choosing an operating system and a web server. Furthermore the freedom of choice in using procedural programming, object oriented programming or a mixture of both. Although not every standard Object Oriented Programming (OOP) feature is realized in the current version of PHP, many code libraries and large applications are written in OOP code.

Output for PHP is not only limited to HTML. PHP's abilities include outputting images, PDF files and even Flash movies. Text such as XHTML and other XML files can be outputted as well. PHP can auto generate these files, and save them in the file system, instead of printing it out, creating a server-side content for a dynamic system.

One of the strongest and most significant feature in PHP is its support for a wide range of databases. A few of the popular databases supported are dBase, mSQL, MySQL, ODBC, Oracle, Sybase and UNIX dbm.

PHP also has support for communicating with other services suing protocols such as LDAP, IMAP, SMNP, NNTP, POP3, HTTP and countless others. PHP even has support for instantiation of JAVA objects. The above mentioned are only just a few of the features offered by PHP [13].

3.5.2.2 ASP

Active Server Pages is an open, compile free application environment which is able to combine Hypertext Mark-Up Language (HTML), scripts and reusable Active X server components to create a dynamic and powerful web-based business solution. ASP enables server side scripting for Internet Information Server (IIS) with native support for both VBScript and JavaScript.

Files created with ASP have the extension .ASP. With ASP files, the web-page can be activated using any combination of HTML scripting such as JavaScript or VBScript and other components written in any language. This means the ASP file is simply a file that can contain any combination of HTML, scripting and components.

When an ASP file is changed on the server, the changes are only saved to the related file and the next time the page is loaded, the script is automatically compiled. This happens because ASP is built directly into Microsoft's web servers, and is supported on all Microsoft web servers such as Windows NT Internet Information Server (IIS), Windows NT Workstation and Windows 95 Personal Web Server.

When a browser requests for a ASP file from the web server, the server calls ASP to send the ASP file, executing any of the commands contained within and sending the resulting HTML page to the browser. An ASP file can contain any combination of HTML, script or command. The script can assign values to variables, request information from the server or combine any set of commands into processes.

Active Server Pages uses the delimiters <% and %> to enclose script commands. The scripting languages supported by ASP in turn support use of the IF-THEN-ELSE construct. ASP includes five standard objects for global use:

i. Request – To retrieve information from server
ii. Response – To send information to user
iii. Server – To control the Internet Information Server

iv. Session – To store information about settings for the users current web-server session

v. Application – To share application level information and control settings for the duration of the application

The request and response objects contain collections which is information that are accessed in the same way and these objects use methods to do some type of process and properties to store any of the object attributes [14].

3.5.2.3 Java Server Pages

Java Server Pages (JSP) technology allows web developers and designers to rapidly develop and easily maintain information rich and dynamic web pages. As part of the Java family, JSP technology enables rapid development of web-based applications that are platform independent. JSP technology separates the user interface from content generation enabling designers to change the overall page layout without altering the underlying dynamic content.

Java Server Pages uses similar XML tags and scripts written in the Java programming language to encapsulate the logic that generates the content for the pages. Additionally, the application logic can reside in server based resources that the page accesses with these tags and scripts. All formatting tags are passed directly back to the response page. By separating the page logic from its design and display and supporting a

reusable component based design, JSP makes it faster and easier than ever to build web based applications.

Java Server Pages is an extension of the Java servelet technology. Servelets are platform independent and 100% pure Java server-side modules that fit easily into a web server framework and can be used to extend capabilities of a web server with minimal overhead, maintenance and support. Unlike other scripting languages, servelets involve no platform specific consideration or modifications. They are Java application components that are downloaded on demand to the part of the system that needs them. Together, JSP technology and servelets provide an attractive alternative to other types of dynamic web scripting or programming that offers platform independency, enhanced performance, separation of logic from display, ease of administration and ease of use [15].

3.6 Database and Server Technologies

3.6.1 Microsoft SQL Server 7.0

Microsoft SQL Server 7.0 makes giant strides in performance, reliability and scalability, offering many avenues to create intelligent, real-world database server solutions.

From the beginning, Microsoft designed their SQL server to make it easy for database administrators to build, manage and deploy business applications. This version automates standard database administration processes. Sophisticated new tools included in this version simplify complex managing processes.

Microsoft SQL Server is a significant tool in many regards. It allows data warehousing to applications that require a large amount of information, multiple users and is a key component in data management requirements and is a comprehensive database.

Microsoft SQL Server is a perfect example of a n-tier system. Users can manipulate the data directly from the client side. Most of the time, the data is validated first before it is updated into the database in the server side. It is the best database for Windows NT server.

Microsoft SQL Server maintains integrity and security and ensures that operation of the server and database can be recorded in the event of failures. This server can control the access for the type of information that can be retrieved by the user.

Microsoft SQL Server can serve laptop databases and even large terabyte sized databases. It integrates well with existing applications and it provides a cost effective environment for customizing and building new applications that meet identified requirements.

Microsoft SQL Server supports database integration. Users are allowed to automate the publishing of database information in HTML documents. It allows users to build active websites and conduct processes on the Internet. When combined with

Internet Information Server (IIS) and the SQL Server Internet Connection, it allows complete database publishing capabilities.

It also provides the function of transparent distributed transactions. This means that it provides automatic distributed update capability across 2 or more SQL servers which are transparent to the desktop application. It guarantees the integrity of transaction of updating multiple servers.

Microsoft SQL Server allows 2 billion tables within each of them 32,767 databases which can be defined. The number of rows in a table is unlimited. It allows users to define up to 250 columns for each table. It also allows users to combine columns for as many as 16 different tables in a single query [16].

3.6.2 Personal Web Server 4.0

Personal Web Server is Microsoft's slimmed down web-server, which provides a basis on which to develop networked applications. It can be run on any home machine that runs all the popular Microsoft operating systems. Personal Web Server is very similar to Internet Information Server (IIS) and comes with ASP 2.0. Personal Web Server allows ASP pages to be created and tested on the same machine as though a external user is accessing the page.

3.6.3 Microsoft Internet Information Server

Internet Information Server (IIS) is designed to deliver a wide range of Internet and Intranet server capabilities. Below, important features of IIS are listed [17]:

i. Integration with Windows NT Server – With the tight integration with Windows NT Server, IIS is easy to set-up, manage and secure.

ii. Enables easy to develop web based applications – IIS introduces Active Server Pages (ASP), which enables posting of dynamic content and development of web based applications to be carried out easily.

iii. Easy installation for a secure environment – IIS includes security features that are easy to install. It works well with the Microsoft Transaction Server to access databases and to provide control at the transaction level.

3.6.4 Microsoft Windows NT Server

Microsoft Windows NT Server is a popular network operating system. It is designed to help developers build and deploy applications faster. Below are listed the important features of Microsoft's Windows NT Server [17]:

i. Windows NT Server is a powerful operating system that provides an integrated environment for developing client-server applications. It provides ease of user and a set of management tools.

ii. Windows NT Server is a complete platform for building and hosting web based applications. It also supports multiple web sites on a single machine. Innovative web publishing features and customizable tools are also offered.

iii. Extensive security support is provided in Windows NT Server. This enables the implementation of user access control in certain files or applications.

iv. Windows NT Server also supports a mid range of network protocols and Remote Access Protocols. This simplifies the development process of publicly available web base applications.

3.6.5 MySQL

MySQL is an open source relational database management system (RDBMS) that uses Structured Query Language (SQL), the most popular language for adding, accessing and processing data in a database. Being open source, anyone can download MySQL and tailor it to their needs in accordance with the general public license. MySQL is used mainly for its speed, reliability and flexibility.

The MySQL relational database system was released in January 1998 and is fully multi-threaded using kernel threads and provides application program interfaces for C, C++, Java, Perl, PHP and many others. MySQL currently runs on Linux, UNIX and Microsoft Windows platforms [18].

3.7 Technology and Tools Selection

3.7.1 Web Language Selection

PHP is chosen as the web programming language to be used. Upon analyzing and comparison with other languages reviews such as HTML and ASP, PHP proves to be the better choice. This being PHP is an open sourced language and is easily understandable. PHP is also one of the most current and new technology available on the Internet. HTML does not prove to serve well as it cannot handle database and Structure Query Languages very well as compared with PHP and ASP. ASP was not picked as it does not support a wide rage of databases as compared with PHP.

3.7.2 Server Selection

Windows NT Server is chosen as the server to be used as it is the most stable amongst all others. The popularity of this server plays an integral part of the choice. Most institutions run this server due to its reliability. Other servers have proven to be less stable as compared with Windows NT Server. Servers which were not as stable are Windows 2000 Server, Microsoft's Personal Web Server and Microsoft's Internet Information Server. Microsoft's Personal Web Server was not picked as it is out dated as compared with newer technologies [19].

3.7.3 Database Management System Selection

MySQL is selected as the database management system as it is open source and easily customizable as compared with Microsoft SQL Server 7.0. MySQL proves to offer more database capabilities that would be useful to the proposed Problem Based Learning

information system. MySQL can also support many other web languages and is used for its speed, reliability and flexibility. Microsoft SQL Server 7.0 was not picked due to the fact that it offers too many features to be used in the system.

3.7.4 Authoring Tool Selection

A few authoring tools are used in the development of the system. Listed below are tools that are used.

3.7.4.1 Macromedia Dreamweaver

This software is used to help website developers build a quality website, with the help of built in functions, which allows the developer to pick out features for the website without having to code the features desired. Besides that, it also contains functions that helps make the website that is developed with this tool more dynamic, such as generating codes to link to databases in the back-end of the website interface. This software is best used for the construction of web applications that are large scaled and rely on the use of databases.

3.8 Hardware and Software Requirements

Table 3.1 lists the minimum hardware and software requirements needed to operate the Problem Based Learning information system.

Table 3.1: Minimum Hardware and Software Requirements

	Hardware	Software
Server	i. Pentium or AMD with 1 GHz processor or more	vi. Windows NT Server vii. MySQL

	ii. 256 RAM or more iii. 10 Gigabytes of Hard disk space iv. Network connection of 100Mbps or more v. Standard input/output peripherals	
Client	i. Pentium II or AMD 400 MHz or above ii. 256 RAM or above iii. Standard input/output peripherals	iv. Any graphic user interface platform or operating system v. Internet Explorer 4 or Netscape Navigator 4 or similar browsers

3.9 Summary

The methodology chosen for the development of the system is the Prototyping Waterfall model for the reasons below:

i. Its sequence of event is clear and easy to explain

ii. Easy to associate each milestone with its deliverables

iii. Flexibility of development of system

In the development of the PBL information system, a prototyping waterfall model is used, which offers flexibility for developers. The system encompasses these functional requirements:

i. Ability for administration and lecturers to create and view reports and statistics related to Problem Based Learning courses.
ii. Allow lecturers to register their PBL courses in to the system with a brief description.
iii. The system allows the upload and download of reports, triggers, lesson plans and thinking points.
iv. A template for course evaluation, peer evaluation and online test is available for lecturers to receive feedback.
v. System is able to automatically tally marks, for course evaluation, peer evaluation and can report who has or has not taken the online test.
vi. Students can fill in evaluation forms online and submit them.
vii. Online submission of student work or assignment is available.
viii. Journal entry for progress tracking is available.
ix. Discussion area for students to interact with other students or lecturer is offered.
x. Students are able to view their marks from the online test and from peer evaluation forms.

Non-functional requirements for the system:

i. User-friendly and Usability

ii. Reliability

iii. Maintainability and Expandability

iv. Modularity

v. Response Time

vi. Security

Software requirements for the system:

i. Macromedia Dreamweaver

ii. Windows NT Server

iii. MySQL

CHAPTER 4: SYSTEM DESIGN

4 System Design

4.1 System Design

Design is a creative process of transforming the problem into a solution; a description of the solution is also called design. The design of a system is split into two main categories, which are the conceptual design and the technical design. The conceptual design will give a graphical and textual explanation of what the proposed system will do when it is finally implemented, and the technical design allows the system builders to understand the actual hardware and software needed to solve the customers' problems. These two designs are what will describe the same system in two different ways, because of the different audiences in each module; the conceptual design would be important to the target user and the developer, while the technical design would interest the system builders.

To achieve a good design and quality software, three characteristics that serve as a guide for the evaluation are the following:

i. The design must implement all of the explicit requirements contained in the analysis mode, and it must accommodate all of the implicit requirements desired.
ii. The design must be easily understood and be readable to the parties involved in generating the code, test the system codes, and subsequently maintain the system; where the design would serve as a guide to these parties.
iii. The design should provide a complete picture of the system, addressing the data, functional and behavioral domains from an implementation perspective.

This chapter will mainly discuss the design of the Problem Based Learning information system in terms of:

i. Architectural design

ii. System Functionality design

iii. Database design

4.2 Architectural Design

The primary objective of architectural design is to develop a modular program structure and represent the control relationship between modules. This architectural design will link the system requirements specification with the system components that will fulfill these requirements and give the system such capabilities. Components are usually modules and the architectural design will give a graphical view of the interconnection of these modules. In addition, the architectural defines operations that create systems for subsystems. The overall Problem Based Learning information system is actually composed of 2 modules as shown in Figure 4.1.

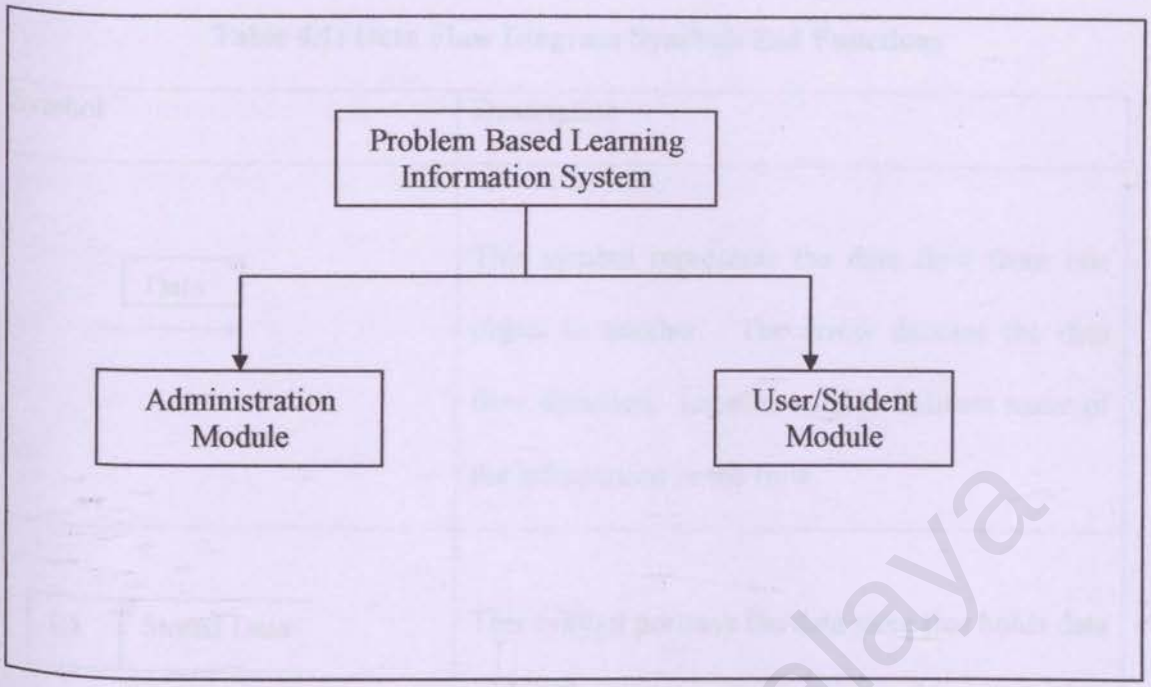
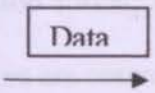
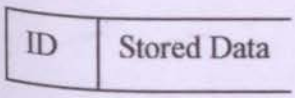
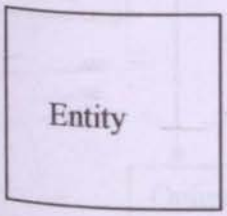


Figure 4.1: Problem Based Learning Information System Architecture

4.3 System Functionality Design

As shown in figure 4.1, the system is split into two main modules, which are the user module and the administration module. This section will discuss the functionality of each module and the Data Flow Diagram (DFD) of each module. DFD is a method to illustrate how data flows in a system. Table 4.1 shows the types of symbols used in a Data Flow Diagram and their functions.

Table 4.1: Data Flow Diagram Symbols and Functions

Symbol	Description
	This symbol represents the data flow from one object to another. The arrow denotes the data flow direction. Label is used to indicate name of the information in the flow.
	This symbol portrays the data store that holds data within the system. It comprises of tow part, thus the identifier and description of data stored.
	This symbol represents an entity such as a person, place or any object.
	This symbol represents a process. Process involves transformation of input data and output data. It consists of two portions. The top portion stores the identifier information. Description of the process is kept in the middle section.

4.3.1 User/Student Module

The user module offers a multitude of functions. Functions such as login into the system, submission of journals and assignments, peer submission forms, course evaluation forms, forum or discussion area and searching of Problem Based Learning resources are controlled by this module. Figure 4.2 below will be used to further illustrate the makings of the user module.

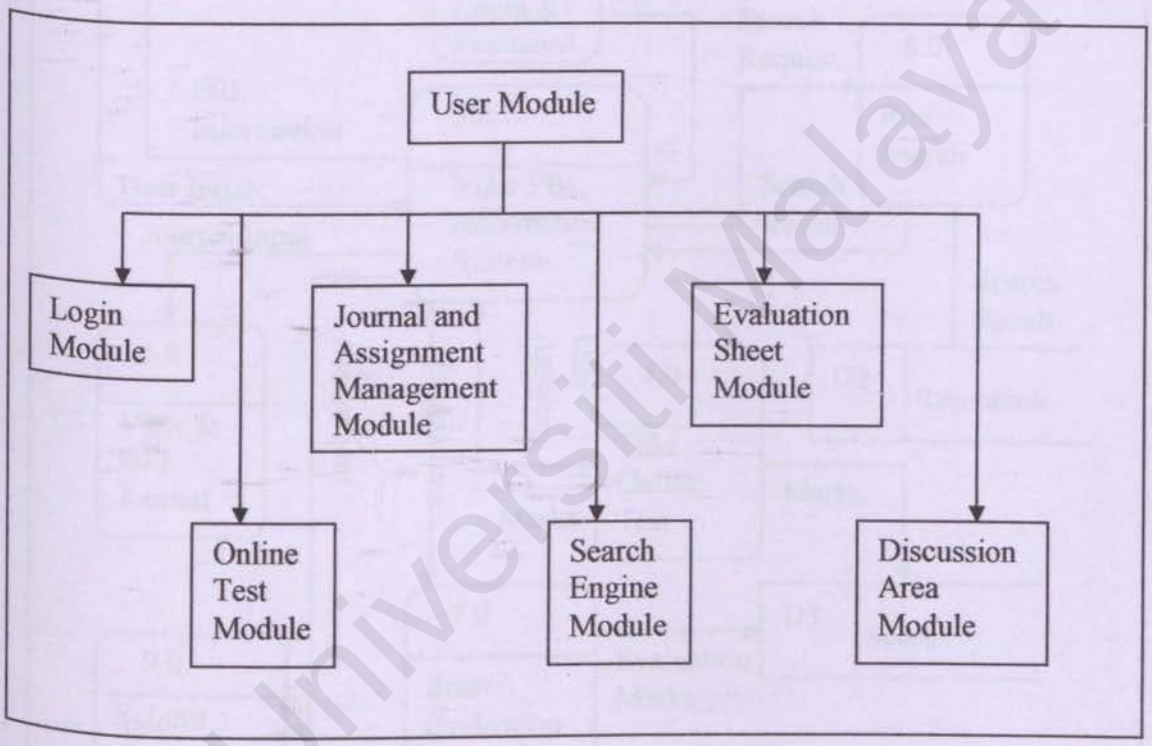


Figure 4.2: User Module Structure

The data flow is shown in Figure 4.3, which illustrates the data flow in the user module.

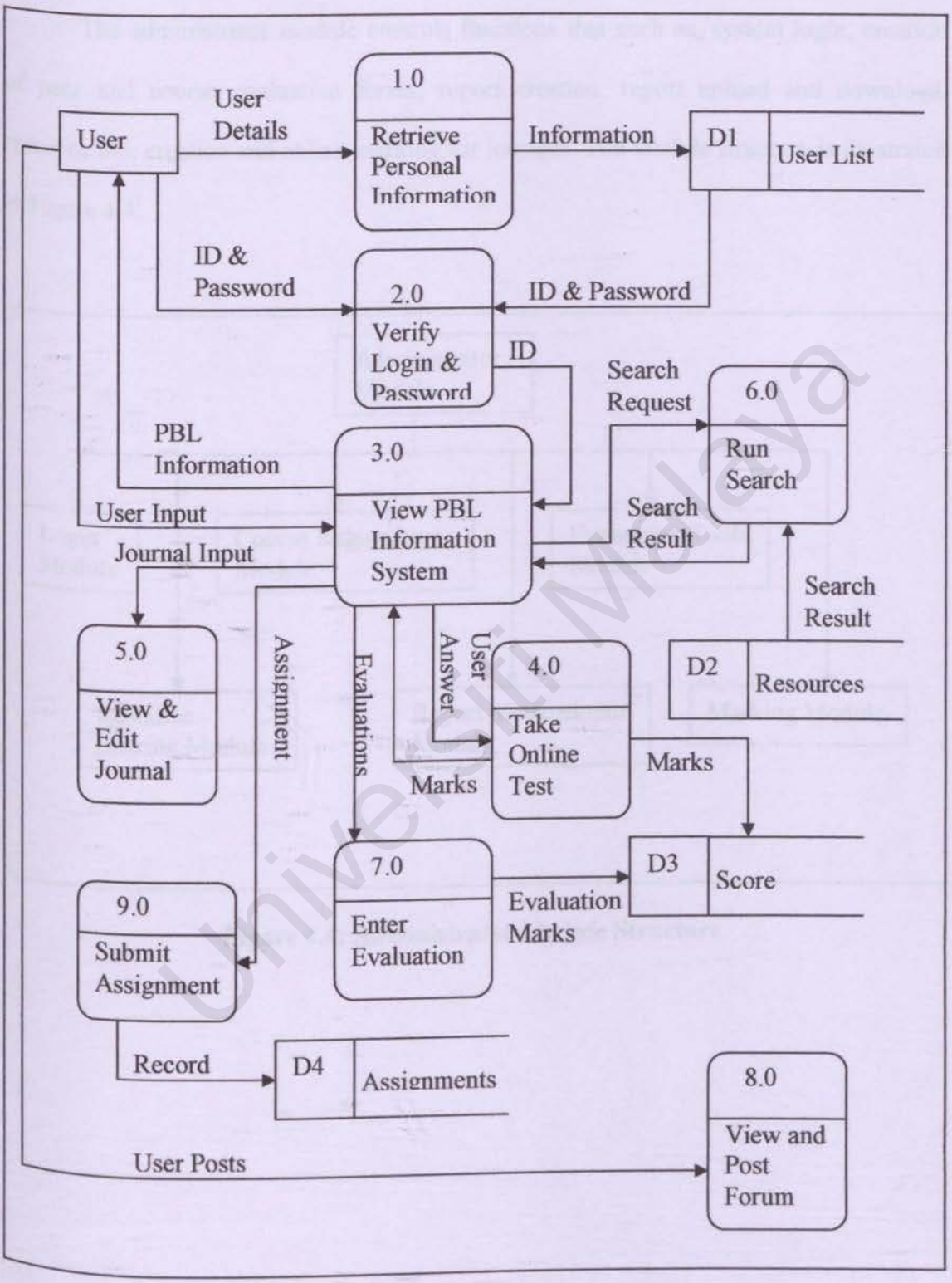


Figure 4.3: User Module Data Flow Diagram

4.3.2 Administrator Module

The administrator module controls functions that such as, system login, creation of peer and course evaluation forms, report creation, report upload and download, resource link creation and online marking for journals. The module structure is illustrated in Figure 4.4.

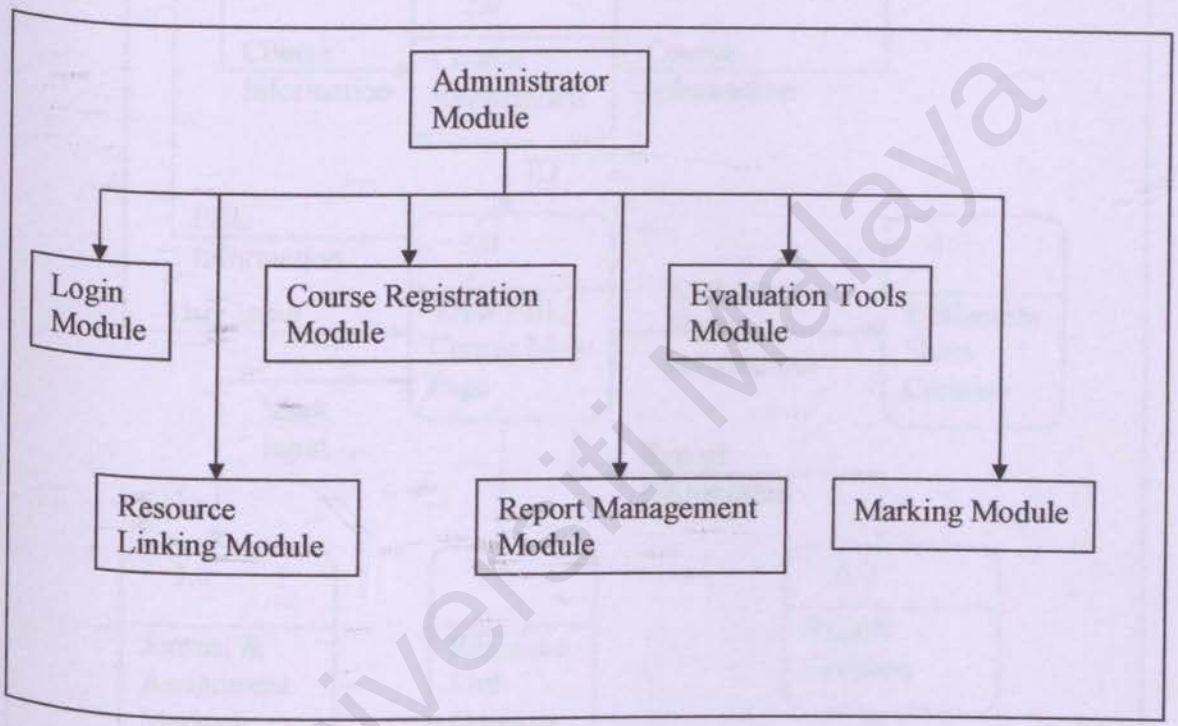


Figure 4.4: Administrator Module Structure

The data flow is shown in Figure 4.5, which illustrates the data flow in the administrator module.

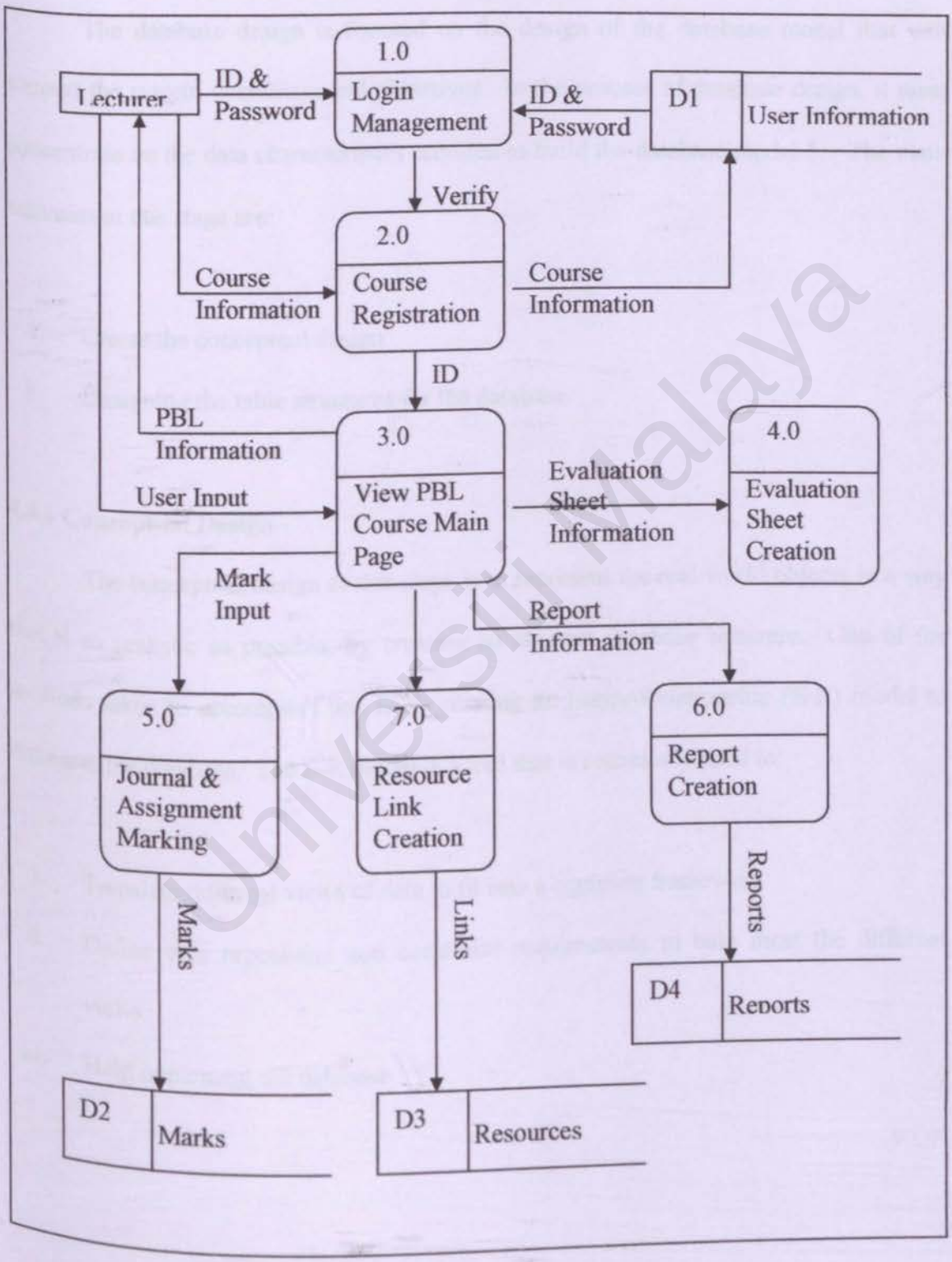


Figure 4.5: Administrator Module Data Flow Diagram

4.4 Database Design

The database design is focused on the design of the database model that will support the system operations and objectives. In the process of database design, it must concentrate on the data characteristics required to build the database model 5. The main activities at this stage are:

i. Create the conceptual design

ii. Designing the table structures for the database

4.4.1 Conceptual Design

The conceptual design at this stage is to represent the real world objects in a way that is as realistic as possible, by creating an abstract database structure. One of the methods taken to accomplish this is by creating an Entity-Relationship (E-R) model to represent the database. The E-R model is a tool that is commonly used to:

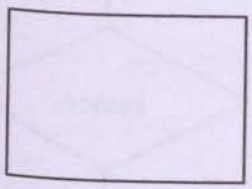
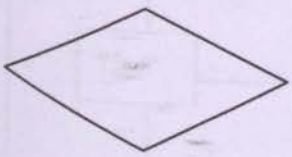
i. Translate different views of data to fit into a common framework

ii. Define data processing and constraint requirements to help meet the different views

iii. Help implement the database

Expressing relationships in a Entity Relationship diagram, certain symbols are used. These symbols and their meanings are depicted in Table 4.2.

Table 4.2: Entity-Relationship Symbols

Symbol	Description
	This symbol represents an entity such as a person, place or any objects.
	This diamond symbol portrays the relationship sets
	The line symbol represents link or relationship between the entities.

To show in greater detail, Figure 4.6 is used to depict the Entity-Relationship Model.

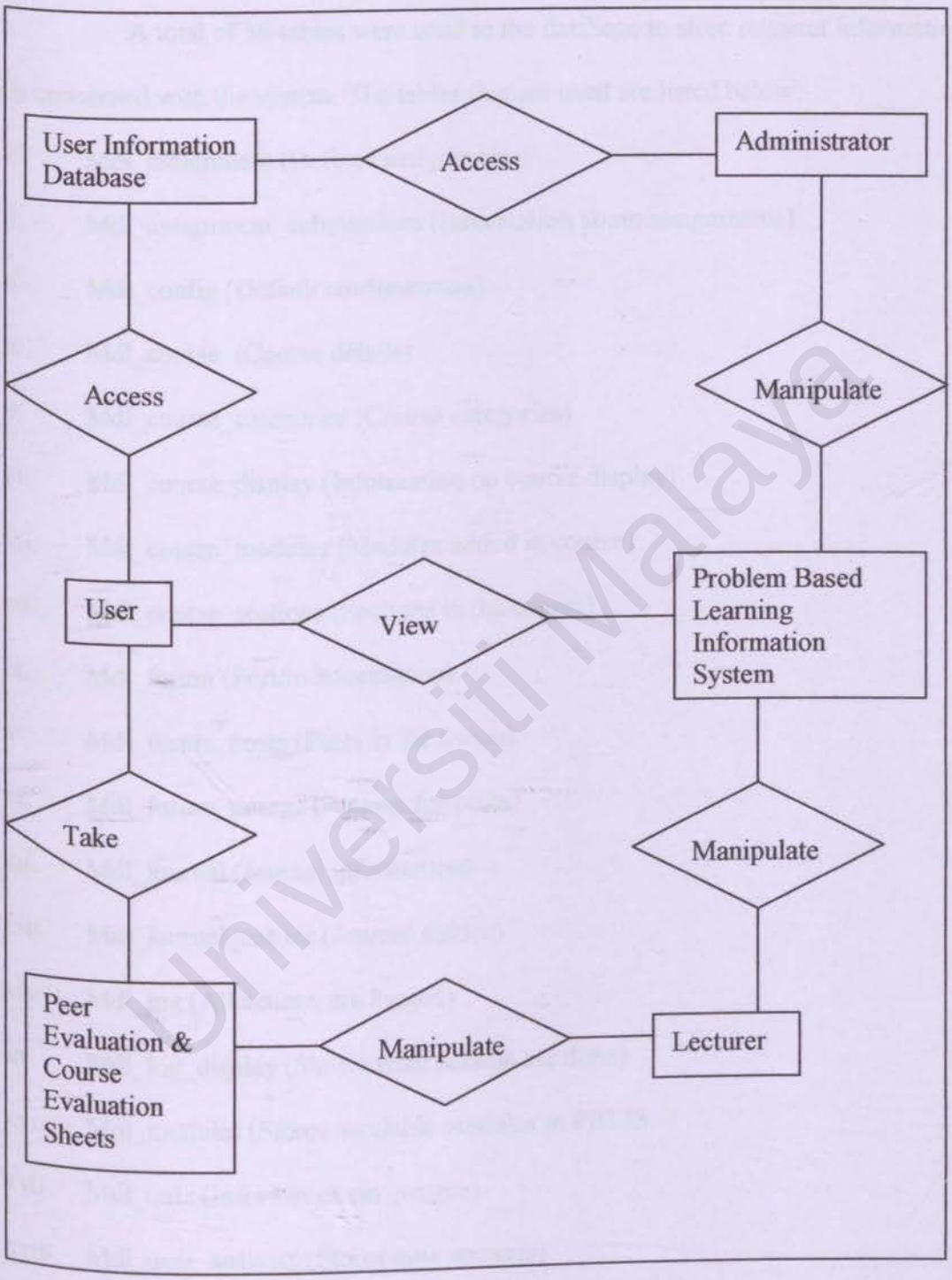


Figure 4.6: Entity-Relationship Model of Problem Based Learning Information System

4.4.2 Database Table Structure

A total of 36 tables were used in the database to store relevant information that is concerned with the system. The tables that are used are listed below:

i. Mdl_assignment (Defines assignments)

ii. Mdl_assignment_submissions (Information about assignments)

iii. Mdl_config (Default configuration)

iv. Mdl_course (Course details)

v. Mdl_course_categories (Course categories)

vi. Mdl_course_display (Information on course display)

vii. Mdl_course_modules (Modules added in course)

viii. Mdl_course_sections (Sections in the course)

ix. Mdl_forum (Forum information)

x. Mdl_forum_posts (Posts in the forum)

xi. Mdl_forum_ratings (Ratings for posts)

xii. Mdl_journal (Journal information)

xiii. Mdl_journal_entries (Journal entries)

xiv. Mdl_log (All actions are logged)

xv. Mdl_log_display (Shows what actions are done)

xvi. Mdl_modules (Stores available modules in PBLIS

xvii. Mdl_quiz (Information on quizzes)

xviii. Mdl_quiz_answers (Stores quiz answers)

xix. Mdl_quiz_attempts (Stores student quiz attempts)

xx. Mdl_quiz_categories (Quiz categories for question groups)

xxi. Mdl_quiz_grades (Quiz marks)

xxii. Mdl_quiz_match (Defines questions)

xxiii. Mdl_quiz_match_sub (Defines sub-questions)

xxiv. Mdl_quiz_multichoice (Stores options for MCQ's)

xxv. Mdl_quiz_question_grades (Marks for each question)

xxvi. Mdl_quiz_questions (Stores questions)

xxvii. Mdl_quiz_responses (Stores user responses to quizzes)

xxviii. Mdl_resource (Stores links)

xxix. Mdl_survey (Stores survey types)

xxx. Mdl_survey_answers (Stores survey answers)

xxxi. Mdl_survey_questions (Stores survey questions)

xxxii. Mdl_user (Stores user information)

xxxiii. Mdl_user_admins (Stores admin information)

xxxiv. Mdl_user_coursecreators (Stores course creator information)

xxxv. Mdl_user_students (Stores student information)

xxxvi. Mdl_user_teacher (Stores lecturer information)

4.4.2.1 Mdl_assignment Table

This table defines assignments that are posted or created in a course. The structure of this table is shown below in Table 4.3.

Table 4.3: Mdl_assignment Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	Assignment ID
Course	Int	10	Course ID
Name	Varchar	255	Assignment Name
Description	Text		Assignment Description
Format	TinyInt	2	File type
Resubmit	TinyInt	2	Allow or disallow resubmission
Type	Int	10	Hardcopy or upload softcopy
Maxbytes	Int	10	Max size of file
Timedue	Int	10	Deadline
Grade	Int	10	Mark for assignment
Timemodified	Int	10	Date uploaded

4.4.2.2 Mdl_assignment_submission Table

This table stores information on submitted assignments. The structure of this table is shown in Table 4.4.

Table 4.4: Mdl_assignment_submission Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	Assignment ID
Assignment	Int	10	Assignment description
UserID	Int	10	User who submitted assignment
Timecreated	Int	10	Time assignment was submitted
Timemodified	Int	10	Time assignment was resubmitted
Numfiles	Int	10	Amount of files in assignment
Grade	Int	10	Mark of assignment
Comment	Text		Comment placed by lecturer
Teacher	Int	10	Lecturer ID
Timemarked	Int	10	Time assignment was marked

4.4.2.3 Mdl_config Table

This table stores default configurations for the site Table 4.5 illustrates the table structure.

Table 4.5: Mdl_config Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	Default Identification
Name	varchar	255	Default descriptions

4.4.2.4 Mdl_course Table

This table stores course information. Table 4.6 shows the structure for the table.

Table 4.6: Mdl_course Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	Course ID
Category	Int	10	Course category
Password	Varchar	255	Password to allow access
Fullname	Varchar	255	Course fullname
Shortname	Varchar	255	Course shortname or course code
Summary	Text		Summary of course
Format	Int	2	Format of course
Modinfo	Text		Course detailed information

Newsitems	Int	10	Available site updates
Startdate	Int	10	Date course starts

4.4.2.5 Mdl_course_categories Table

The function of this table is to show course categories or departments. Shown in Table 4.7 is the structure for the table.

Table 4.7: Mdl_course_categories Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	Course ID
Name	Varchar	255	Category name

4.4.2.6 Mdl_course_display Table

This table holds the information on how the course is displayed to the users. The table structure is shown in Table 4.8.

Table 4.8: Mdl_course_display Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	Display ID
Course	Int	10	Course ID
UserID	Int	10	User ID

Display	Int	10	Type of display

4.4.2.7 Mdl_course_modules Table

This table represents information on modules added in a course. The structure is shown in Table 4.9.

Table 4.9: Mdl_course_modules Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	Course ID
Course	Int	10	Course name
Module	Int	10	Module ID
Instance	Int	10	Session ID
Section	Int	10	Sections available
Added	Int	10	Number of modules
Deleted	Tinyint	10	Delete option available
Score	Tinyint	10	Marks for the module

4.4.2.8 Mdl_course_sections Table

This table stores the sections available in a course. Shown below is Table 4.10 is the table structure for the Mdl_course_sections table.

Table 4.10: Mdl_course_sections Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	Course ID
Course	Int	10	Course name
Section	Int	10	Number of sections
Summary	Text		Summary of course
Sequence	Varchar	255	Amount of weeks
Visible	Tinyint	2	Visible to selected users

4.4.2.9 Mdl_forum Table

The Mdl_forum table stores forum information. Table 4.11 depicts the table structure for Mdl_forum.

Table 4.11: Mdl_forum Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	Forum ID
Course	Int	10	Course ID
Type	Int	10	Forum type
Name	Varchar	255	Forum name

Intro	Text		Forum introduction
Open	Int	10	Allow posts or not
Assessed	Int	10	Time created
Timemodified	Int	10	Time modified

4.4.2.10 Mdl_forum_discussion Table

This table stores the sections available in a course. Shown below is Table 4.12 is the table structure for the Mdl_forum_discussion table.

Table 4.12: Mdl_forum_discussion Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	Forum ID
Course	Int	10	Course ID
Forum	Int	10	Forum
Name	Varchar	255	Forum name
Firstpost	Int	10	Date of first post
Timemodified	Int	10	Time post modified

4.4.2.11 Mdl_forum_posts Table

This table stores the sections available in a course. Shown below is Table 4.13 is the table structure for the Mdl_forum_posts table.

Table 4.13: Mdl_forum_posts Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	Forum ID
Discussion	Int	10	Discussion ID
Parent	Int	10	Parent node ID
Userid	Int	10	User ID
Created	Int	10	Date of post
Modified	Int	10	Date post modified
Subject	Varchar	255	Subject of post
Message	Text		Message in post
Format	Tinyint	2	Post format
Attachment	Varchar	255	Location of attachment
Totalscore	Tinyint	6	Total mark

4.4.2.12 Mdl_forum_ratings Table

This table stores the sections available in a course. Shown below is Table 4.14 is the table structure for the Mdl_forum_ratings table.

Table 4.14: Mdl_forum_ratings Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	Forum ID
UserId	Int	10	User ID
Post	Int	10	Post ID

Time	Int	10	Time of post

4.4.2.13 Mdl_journal Table

This table stores the sections available in a course. Shown below is Table 4.15 is the table structure for the Mdl_journal table.

Table 4.15: Mdl_journal Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	Journal ID
Course	Int	10	Course ID
Name	Varchar	255	Journal name
Intro	Text		Journal intro
Timemodified	Int	10	Time journal modified

4.4.2.14 Mdl_journal_entries Table

This table stores the sections available in a course. Shown below is Table 4.16 is the table structure for the Mdl_journal_entries table.

Table 4.16: Mdl_journal_entries Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	ID
Journal	Int	10	Journal ID
UserId	Int	10	User ID

Modified	Int	10	Day modified
Text	Text		Journal entry
Format	Tinyint	2	Journal format
Comment	Text		Comment on journal
Teacher	Int	10	Lecturer ID
Timemarked	Int	10	Time journal marked

4.4.2.15 Mdl_log Table

This table stores the sections available in a course. Shown below is Table 4.17 is the table structure for the Mdl_log table.

Table 4.17: Mdl_log Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	Log ID
Time	Int	10	Time of login
UserID	Int	10	User ID
IP	Varchar	255	IP address
Course	Int	10	Course ID
Module	Varchar	255	Module ID
Action	Varchar	255	Action performed
URL	Varchar	255	URL text
Info	Varchar	255	Log information

4.4.2.16 Mdl_log_display Table

This table stores the sections available in a course. Shown below is Table 4.18 is the table structure for the Mdl_log_display table.

Table 4.18: Mdl_log_display Table Structure

Field Name	Field Type	Field Size	Description
Module	Varchar	255	Module ID
Action	Varchar	255	Action performed

4.4.2.17 Mdl_modules Table

This table stores the sections available in a course. Shown below is Table 4.19 is the table structure for the Mdl_modules table.

Table 4.19: Mdl_modules Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	Module ID
Name	Varchar	255	Module name
Search	Int	10	Available for search
Visible	Tinyint	2	Visible or invisible

4.4.2.18 Mdl_quiz Table

This table stores the sections available in a course. Shown below is Table 4.20 is the table structure for the Mdl_quiz table.

Table 4.20: Mdl_quiz Table Structure

Field Name	Field Type	Field Size	Description
Id	Int	10	Quiz ID
Course	Int	10	Course ID
Name	Varchar	255	Quiz name
Intro	Text		Quiz intro
Timeopen	Int	10	Time quiz opens
Timeclose	Int	10	Time quiz closes
Attempts	Smallint	6	Attempts on quiz
Questions	Text		Questions
Sumgrades	Int	10	Mark total
Grades	Int	10	Mark
Timecreated	Int	10	Time quiz created
Timemodified	Int	10	Time quiz modified

4.4.2.19 Mdl_quiz_answers Table

This table stores the sections available in a course. Shown below is Table 4.21 is the table structure for the Mdl_quiz_answers table.

Table 4.21: Mdl_quiz_answers Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	Quiz ID
Question	Int	10	Question ID
Answer	Varchar	255	Question answer
Fraction	Varchar	255	Mark
Feedback	Text		Comment on question

4.4.2.20 Mdl_quiz_attempts Table

This table stores the sections available in a course. Shown below is Table 4.22 is the table structure for the Mdl_quiz_attempts table.

Table 4.22: Mdl_quiz_attempts Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	Attempt ID
Quiz	Int	10	Quiz ID
UserID	Int	10	User ID
Attempt	Smallint	6	Number of attempts
Sumgrades	Varchar	255	Mark total

Timestart	Int	10	Time quiz started
Timefinish	Int	10	Time quiz finished
Timemodified	Int	10	Time quiz modified

4.4.2.21 Mdl_quiz_categories Table

This table stores the sections available in a course. Shown below is Table 4.23 is the table structure for the Mdl_quiz_categories table.

Table 4.23: Mdl_quiz_categories Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	Category ID
Course	Int	10	Course ID
Name	Varchar	255	Category name
Info	Text		Information on category

4.4.2.22 Mdl_quiz_grades Table

This table stores the sections available in a course. Shown below is Table 4.24 is the table structure for the Mdl_quiz_grades table.

Table 4.24: Mdl_quiz_grades Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	Grade ID
Quiz	Int	10	Quiz ID

UserID	Int	10	User ID
Grade	Varchar	255	Mark
Timemodified	Int	10	Time modified

4.4.2.23 Mdl_quiz_match Table

This table stores the sections available in a course. Shown below is Table 4.25 is the table structure for the Mdl_quiz_match table.

Table 4.25: Mdl_quiz_match Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	Quiz ID
Questions	Int	10	Question ID
Subquestions	Varchar	255	Subquestions

4.4.2.24 Mdl_quiz_match_sub Table

This table stores the sections available in a course. Shown below is Table 4.26 is the table structure for the Mdl_quiz_match_sub table.

Table 4.26: Mdl_quiz_match_sub Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	Subquestion ID
Question	Int	10	Question ID
Questiontext	Text		Question

Answertext	Varchar	255	Answer

4.4.2.25 Mdl_quiz_multichoice Table

This table stores the sections available in a course. Shown below is Table 4.27 is the table structure for the Mdl_quiz_multichoice table.

Table 4.27: Mdl_quiz_multichoice Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	MCQ ID
Question	Int	10	Question ID
Answers	Varchar	255	Answers

4.4.2.26 Mdl_quiz_question_grades Table

This table stores the sections available in a course. Shown below is Table 4.28 is the table structure for the Mdl_quiz_question_grades table.

Table 4.28: Mdl_quiz_question_grades Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	ID
Quiz	Int	10	Quiz ID
Question	Int	10	Question ID
Grade	Smallint	6	Mark allocated

4.4.2.27 Mdl_quiz_questions Table

This table stores the sections available in a course. Shown below is Table 4.29 is the table structure for the Mdl_quiz_questions table.

Table 4.29: Mdl_quiz_questions Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	Quiz ID
Category	Int	10	Category ID
Name	Varchar	255	Quiz name
Questiontext	Text		Question
Defaultgrade	Int	10	Default mark

4.4.2.28 Mdl_quiz_responses Table

This table stores the sections available in a course. Shown below is Table 4.30 is the table structure for the Mdl_quiz_responses table.

Table 4.30: Mdl_quiz_responses Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	UserID
Attempt	Int	10	Number of attempts
Question	Int	10	Question ID
Answer	Varchar	255	Answer
Grade	Varchar	255	Marks

4.4.2.29 Mdl_resource Table

This table stores the sections available in a course. Shown below is Table 4.31 is the table structure for the Mdl_resource table.

Table 4.31: Mdl_resource Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	Resource ID
Course	Int	10	Course ID
Name	Varchar	255	Resource name
Type	Tinyint	2	Resource type
Reference	Varchar	255	Reference text
Summary	Text		Summary of resource
Alltext	Text		Type of resource
Timemodified	Int	10	Time modified

4.4.2.30 Mdl_survey Table

This table stores the sections available in a course. Shown below is Table 4.32 is the table structure for the Mdl_survey table.

Table 4.32: Mdl_survey Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	Survey ID
Course	Int	10	Course ID

Template	Int	10	Survey template
Days	Smallint	6	Days survey is open
Timecreated	Int	10	Time survey created
Timemodified	Int	10	Time survey modified
Name	Varchar	255	Survey name
Intro	Text		Survey introduction
Questions	Varchar	255	Survey question

4.4.2.31 Mdl_survey_answers Table

This table stores the sections available in a course. Shown below is Table 4.33 is the table structure for the Mdl_survey_answers table.

Table 4.33: Mdl_survey_answers Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	Answer ID
UserID	Int	10	User ID
Survey	Int	10	Survey ID
Question	Int	10	Question ID
Time	Int	10	Time created
Answer1	Char	255	Answer 1
Answer2	Char	255	Answer 2

4.4.2.32 Mdl_survey_questions Table

This table stores the sections available in a course. Shown below is Table 4.34 is the table structure for the Mdl_survey_questions table.

Table 4.34: Mdl_survey_questions Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	Survey ID
Text	Varchar	255	Survey text
Shorttext	Varchar	255	Survey short question
Intro	Varchar	255	Survey introduction
Type	Tinyint	2	Survey type

4.4.2.33 Mdl_user Table

This table stores the sections available in a course. Shown below is Table 4.35 is the table structure for the Mdl_user table.

Table 4.35: Mdl_user Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	User ID
Username	Varchar	255	User name
Password	Varchar	255	Password
Idnumber	Varchar	255	Matric number
Firstname	Varchar	255	First name

Lastname	Varchar	255	Last name
Email	Varchar	255	User email
Instituition	Varchar	255	User instituition
Department	Varchar	255	User department
City	Varchar	255	City
Firstaccess	Int	10	First time accessed
Lastlogin	Int	10	User's last login

4.4.2.34 Mdl_user_admins Table

This table stores the sections available in a course. Shown below is Table 4.36 is the table structure for the Mdl_user_admins table.

Table 4.36: Mdl_user_admins Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	Admin ID
UserID	Int	10	User ID

4.4.2.35 Mdl_user_coursecreators Table

This table stores the sections available in a course. Shown below is Table 4.37 is the table structure for the Mdl_user_coursecreators table.

Table 4.37: Mdl_user_coursecreators Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	Course creator ID
UserID	Int	10	User ID

4.4.2.36 Mdl_user_students Table

This table stores the sections available in a course. Shown below is Table 4.38 is the table structure for the Mdl_user_students table.

Table 4.38: Mdl_user_students Table Structure

Field Name	Field Type	Field Size	Description
ID	Int	10	Student ID
UserId	Int	10	User ID
Course	Int	10	Course ID
Timestart	Int	10	Time started
Timeend	Int	10	Time ended
Time	Int	10	Time user created

4.5 User Interface Design

Graphic User Interfaces (GUIs) are used to determine the best possible user interface for the system. Considering the fact that the system is meant to be set online or is a web-based system, the interface must be attractive to attract the users' attention, but at the same time must be practical and easy to use, as it is a Problem Based Learning information system for users without very technical backgrounds. The easier the user interface is to understand, the better it is to relay information to the student. The Human Computer Interface (HCI) general principles of designing an interactive system are shown in Table 4.39 below:

Table 4.39: Human Computer Interface General Principles

Principle	Description
Consistency	Consistent format for command input, data display, menu selection, and placing of the control objects
Confirmation and Verification Message	Ask for verification for any non-trivial destruction such as delete a record
Recoverability	Ability of the user to take corrective action once an error has been recognized
Forgive Mistake	The system should protect itself from user error that might cause it to fail
Reverse Action	Allow user to return to the previous state (before change)
Functions Grouping	Categorize activities by function and organize screen geography accordingly

Simple Command Name	Use short and meaningful code to name commands. Short name is easy to memorize and reduce typing mistake
Responsiveness	How the user perceives the rate of command with the system. For example the mouse pointer changes to hourglass or displays a wait message when the system is processing data
Context-Sensitive Help	Provide relevant help topic for current state, when user needs the Help system

4.6 Input Form Design

There are seven main input forms available in the student module. These input forms are the login form, assignment upload form, journal entry form, peer evaluation form, course evaluation form, report template form and the discussion area form. The designs of these forms take in to consideration a few of the Human Computer Interface (HCI) principles.

4.6.1 Login Form

The login form is created in a simple manner. The login form is the first form view by all users of the system; students and lecturers. This form has the normal layout of a user ID, user password and a submit button. The layout of this form can be seen in Figure 4.7.

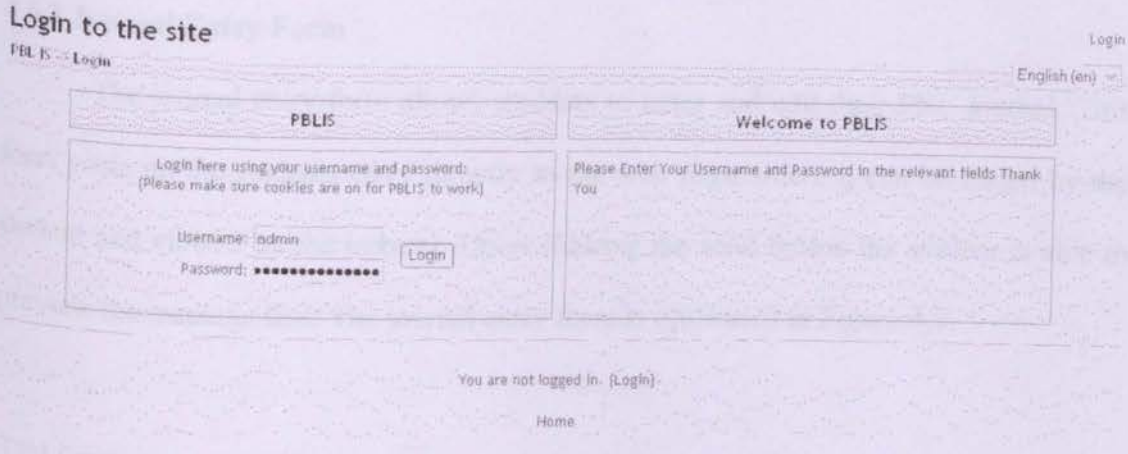


Figure 4.7: Login Form

4.6.2 Assignment Upload Form

The assignment upload form is for use by individual assignment submission or group submission. This form offers students the choice to pick between individual submission and group submission. The fields to be filled in by the student are matric number and assignment filename. The screenshot of this form can be seen in Figure 4.8 and shows a very simple interface.

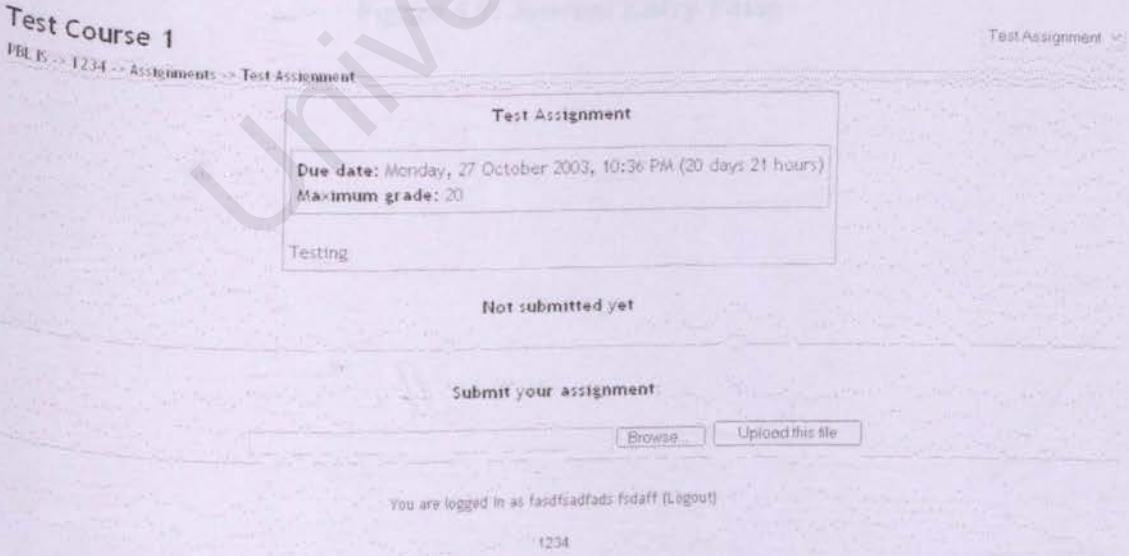


Figure 4.8: Assignment Submission Form

4.6.3 Journal Entry Form

The journal entry form allows students to enter and edit their PBL journal. This form posts up the students entry directly to the web page where it can be edited by the student and viewed by the lecturer. Upon clicking the send button the student is able to preview the message first. The journal entry form is illustrated in Figure 4.9.

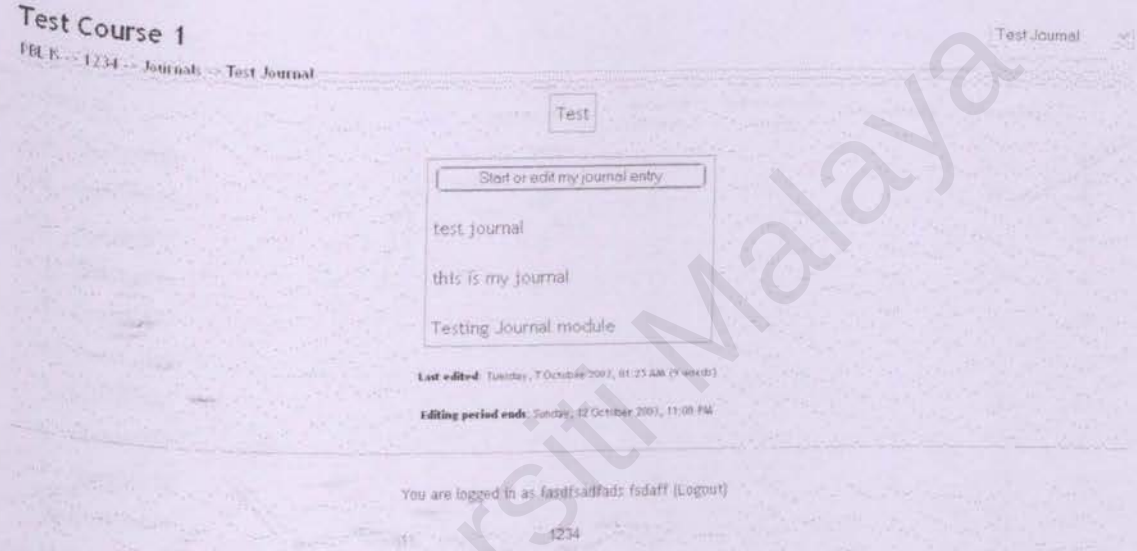


Figure 4.9: Journal Entry Form

4.6.4 Peer Evaluation Form

The peer evaluation form allows students to input marks for each of their group members. The group member list is automatically matched with the student's matric number from a database for the specific course. Available to students are a set of drop down boxes with a selection of marks for members in the group. Figure 4.10 is used to better depict the layout of the peer evaluation form.

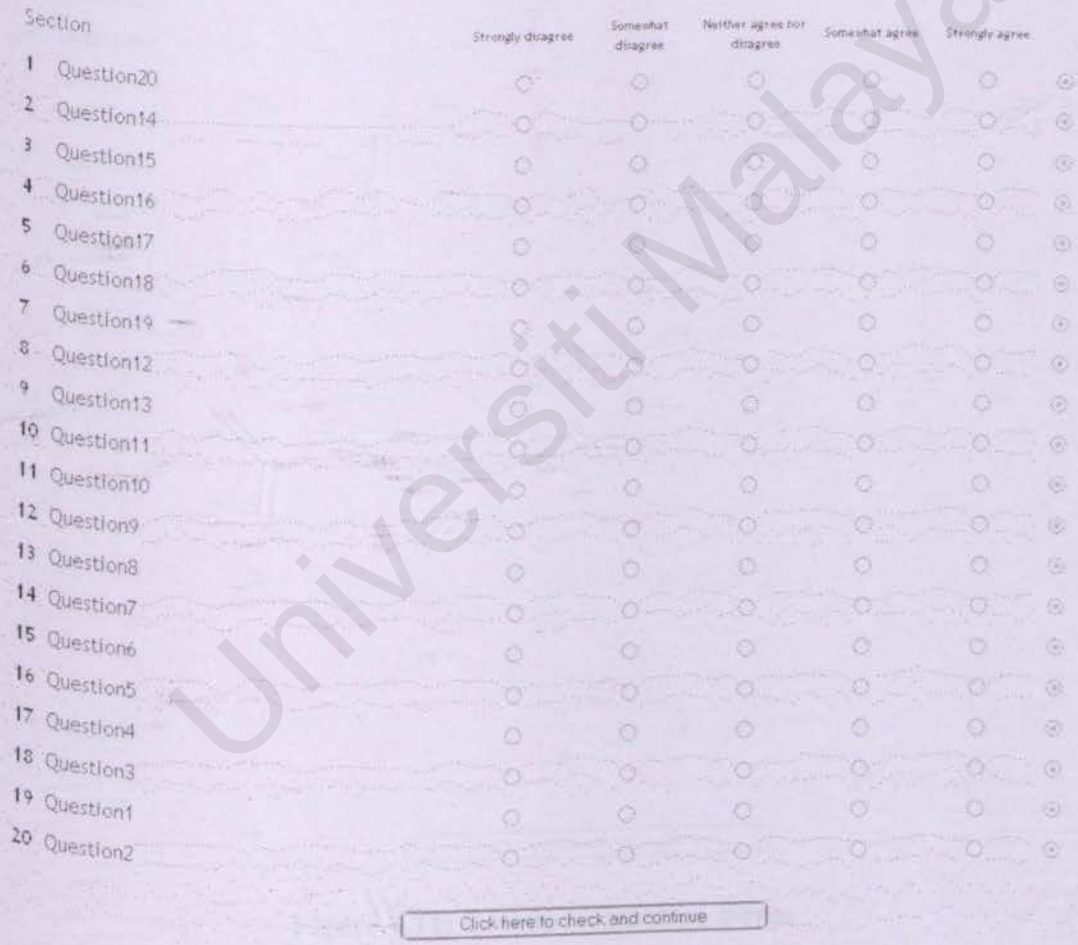


Figure 4.10: Peer Evaluation Form

4.6.5 Course Evaluation Form

This form allows students to carry out a course evaluation at the end of the course. Online evaluation is done to automate this process without taking time from teaching periods. This form will have questions followed by drop-down boxes so the layout of the form looks neat and consistent. Figure 4.11 is used to illustrate the layout of the course evaluation form.

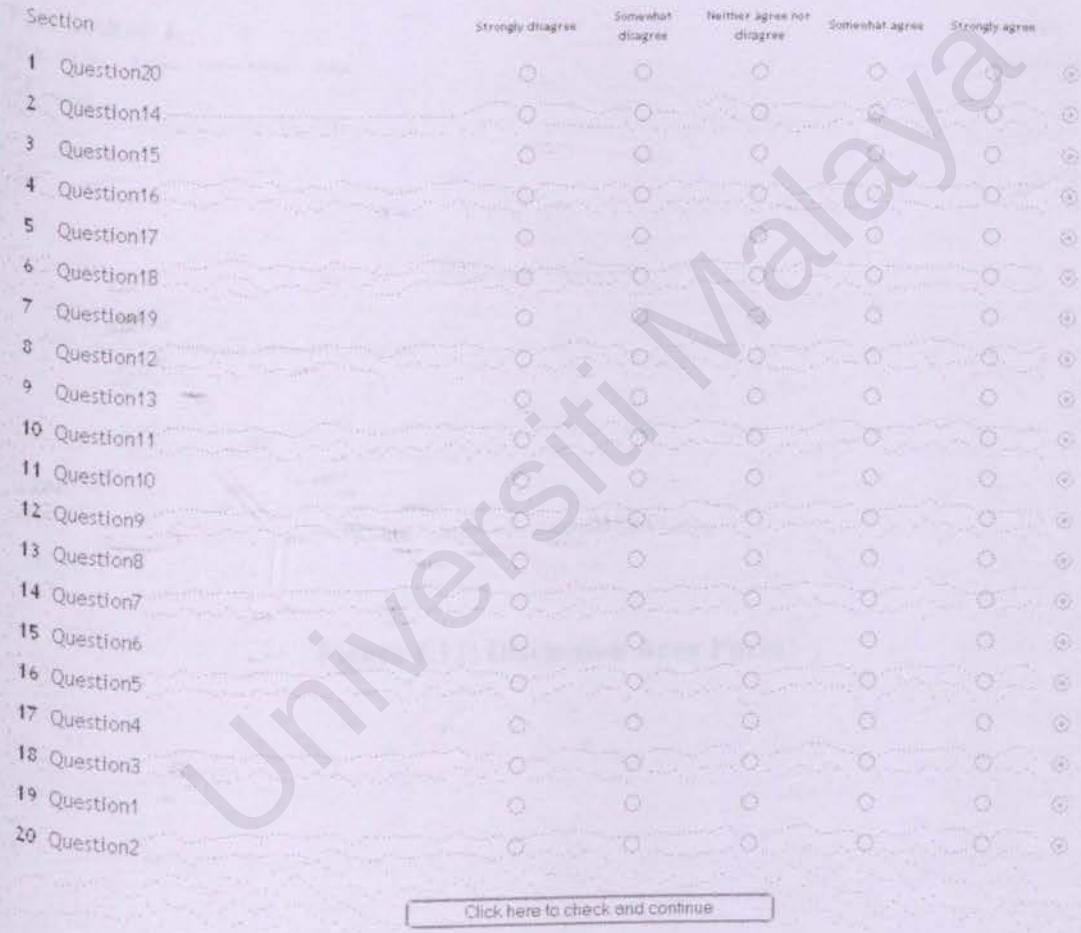


Figure 4.11: Course Evaluation Form

4.6.6 Discussion Area Form

The discussion area form is created to allow students and lecturers to easily create threads and post messages. The form will allow the person who posted their message to delete it if necessary, hence giving the option of removing irrelevant or wrongly posted messages. The layout of this form is shown in Figure 4.12.

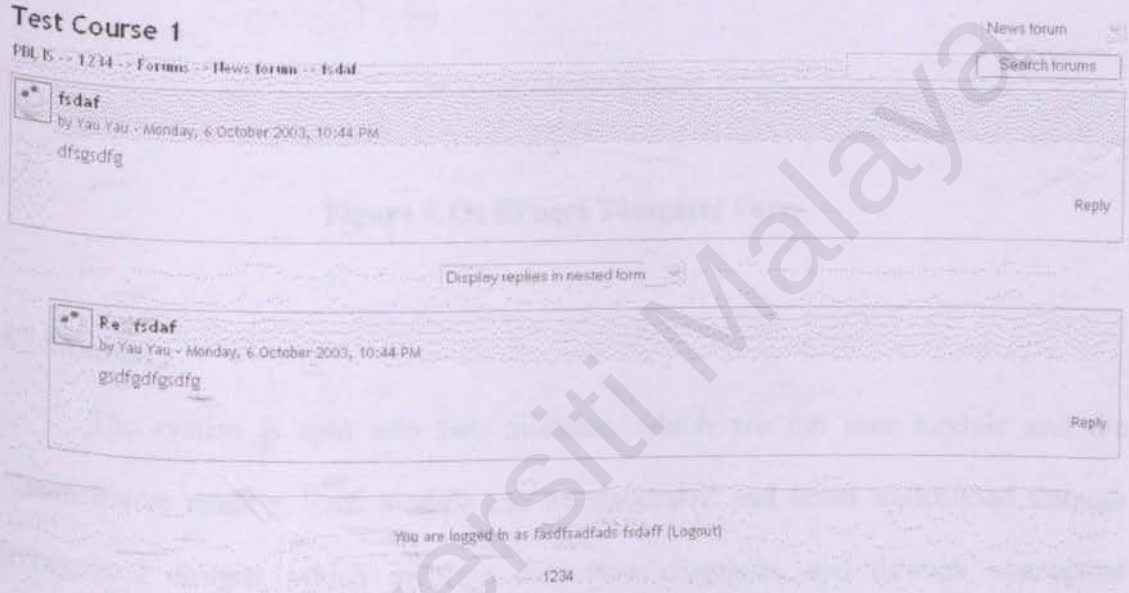


Figure 4.12: Discussion Area Form

4.6.7 Report Template Form

This form is used by lecturers to upload their reports online. Figure 4.13 is used to show the sample screenshot.

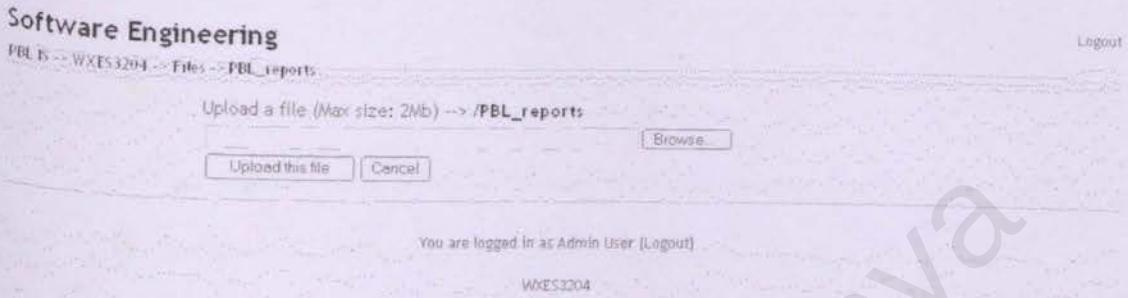


Figure 4.13: Report Template Form

4.7 Summary

The system is split into two modules, which are the user module and the administrative module. Each module can be elaborated and better understood through architectural designs, which involves data flow diagrams, and through conceptual designs, which includes the creation of an Entity-Relationship Diagram. Establishing the structure of the database that will be used for system will also make the development process more organized and better prepared for. Establishing and taking into account all the criteria that make a good GUI is also important especially since the system needs to be easily understood by the users.

CHAPTER 5: SYSTEM IMPLEMENTATION

5 System Implementation

System implementation involves the translation of the software representation produced during the design phase into a computer readable form. This is the phase that involves coding and writing of the program using several development tools.

5.1 Development Environment

There are a few strategies can be applied in this proposed system such as top-down design and modular development. Development strategies are very important because it determines how the coding process begins, debugs, testing and maintains. Thus, determines the right and suitable strategies from the beginning are vital. Any development strategies changes in the middle of system development will increase the cost and development time of the entire system.

5.1.1 Hardware Configuration

In order to implement the system, the following minimum requirements must be met :

i. Pentium or AMD with 1 GHz processor or more

ii. 256 RAM or more

iii. 10 Gigabytes of Hard disk space

iv. Network connection of 100Mbps or more

v. Standard input/output peripherals

5.1.2 Tools Used for System Development

The right kind of system tools is vital in the development of the system. Table 5.1 below lists the software used for the development of the Problem Based Learning Information System:

Table 5.1: Tools Used for System Development

Software	Module	Description
Macromedia Dreamweaver	System Requirement and Interface Design	Web page coding and layout
Zend Development Environment	System Requirement and Interface Design	Web page PHP coding development environment
Apache HTTP Server 1.3	System Requirement	Web Server
Internet Explorer 6.0	System Development	Web page design
Microsoft Windows XP Professional Edition	System Development	Operating System
MySQL	System Development	Database design

5.2 Implementation of PBLIS

The PBLIS administration module is developed to monitor and maintain the database through the website. This module is used by the lecturers to update or maintain their own class material. Client server architecture is used to develop PBLIS using MySQL and Apache as the backend development tool and Zend Studio as the front-end programming language for the designing of easy-to-use and easy to learn user interfaces. Basically, the implementation of PBLIS is divided into several stages, database creation, form creation and interface creation.

5.3 Database Creation

There are 5 steps involved in creating a database from scratch; creating tables, grouping the fields, building relationships, normalizing the database and testing the database with test data. First a list of fields is identified and created, the data type for each field is then selected and grouped logically and each table has a primary key and a foreign key. Tables that are used in PBLIS can be found in Chapter 4.

A total of 36 tables were used in the database to store relevant information that is concerned with the system. The tables that are used are listed below:

i. Mdl_assignment (Defines assignments)

ii. Mdl_assignment_submissions (Information about assignments)

iii. Mdl_config (Default configuration)

iv. Mdl_course (Course details)

v. Mdl_course_categories (Course categories)

vi. Mdl_course_display (Information on course display)

vii. Mdl_course_modules (Modules added in course)

viii. Mdl_course_sections (Sections in the course)

ix. Mdl_forum (Forum information)

x. Mdl_forum_posts (Posts in the forum)

xi. Mdl_forum_ratings (Ratings for posts)

xii. Mdl_journal (Journal information)

xiii. Mdl_journal_entries (Journal entries)

xiv. Mdl_log (All actions are logged)

xv. Mdl_log_display (Shows what actions are done)

xvi. Mdl_modules (Stores available modules in PBLIS

xvii. Mdl_quiz (Information on quizzes)

xviii. Mdl_quiz_answers (Stores quiz answers)

xix. Mdl_quiz_attempts (Stores student quiz attempts)

xx. Mdl_quiz_categories (Quiz categories for question groups)

xxi. Mdl_quiz_grades (Quiz marks)

xxii. Mdl_quiz_match (Defines questions)

xxiii. Mdl_quiz_match_sub (Defines sub-questions)

xxiv. Mdl_quiz_multichoice (Stores options for MCQ's)

xxv. Mdl_quiz_question_grades (Marks for each question)

xxvi. Mdl_quiz_questions (Stores questions)

xxvii. Mdl_quiz_responses (Stores user responses to quizzes)

xxviii. Mdl_resource (Stores links)

xxix. Mdl_survey (Stores survey types)

xxx. Mdl_survey_answers (Stores survey answers)

xxxi. Mdl_survey_questions (Stores survey questions)

xxxii. Mdl_user (Stores user information)

xxxiii. Mdl_user_admins (Stores admin information)

xxxiv. Mdl_user_coursecreators (Stores course creator information)

xxxv. Mdl_user_students (Stores student information)

xxxvi. Mdl_user_teacher (Stores lecturer information)

5.3.1 Adding Tables

Based on the groups of fields selected to represent the identity of each data field of PBLIS, each of the groups is then added into the table. One group represents one table. Adding and creating the table is made easy by using MySQL Control Centre. In each table, all fields and their data types are entered and saved to create and update the table into the database.

5.3.2 Building Relationships between Tables

The next task is to build relationships among the created tables. The pre-planned version of the relational database model is now converted into the actual database by drawing and linking the relationships between those tables. When setting up the relationships between tables, all the tables in PBLIS are tested for referential integrity, which is done to ensure related tables are valid and data cannot be accidentally changed or deleted.

5.3.3 Normalize the Database

Normalization consists of set of standard rules that help to prevent design flaws that cause repeated and inconsistent data. There are the 3 base steps in the normalization process. The first step is to form the first normal form where repeating groups are broken down into a new table. The second normal form is done by making sure all non-key fields are dependant on the primary key. Finally in the third normal form, no fields should be dependant on other non-key fields.

5.3.4 Testing the database

Before declaring the database is in full working order, self-generated sample data is put into the database. Queries done on MySQL tests the relationships and structure of the database by building a few queries to retrieve some data from the database.

5.4 Forms Creation

At this stage, the interface for PBLIS is converted to an actual form for users to input or retrieve data. Zend Studio and Macromedia Dreamweaver were used to help created the forms and interface for PBLIS.

A form is like a window through which users view the fields of a document; forms are the only way to see documents. A form may display all or a subset of the fields in the document. When a form is designed, components to show and how they should look to users are decided upon.

5.5 Creating Views

Standard-style views display document summaries in rows and columns so users can find the documents they want to read. To read documents in a database, a user opens a view then opens the document, which is represented by one row. At least one view is created for all PBLIS databases. All these views are associated with a form in such a way that dynamic web pages are created to form elements such as stylized text, graphics and tables which adds visual interest to the user.

5.6 Creating Links

As PBLIS is a multi-database web application, URL links are required to allow users to navigate through documents, web pages and databases. These links are placed at the navigation bar on the left of the webpage.

5.7 Securing the System

MySQL and Apache server offers a variety of security mechanisms that allow certain privilege levels for different users. In PBLIS, there are two groups of access levels that are required, one for the students and another for the lecturers. To control the users privilege, the following point of views are taken into consideration:

i) To allow anonymous access to PBLIS, a table entry containing anonymous is entered

ii) To allow registered users to access PBLIS, a username and password is created in the table

iii) To control display of database elements within forms and hide-when capabilities for field, action and section are used. As an example, the 'Add New Announcement' are visible for lecturers but hidden when students access PBLIS

iv) To restrict user access to specific document, certain fields in a document are created. For example, students field on a document only contains registered matric numbers for that particular course and none other

v) To control access to the PBLIS database. For example, anonymous users are unable to view or access the tables and database of PBLIS.

5.8 Coding

Coding is a process that translates a system design into a programming language realization, which is a machine-readable form. The coding methodology used for PBLIS is a bottom-up approach. Bottom-up coding begins with some complete lower level modules. The complete lower level modules are integrated with the newly completed high-level modules. This allows the lower level modules to be created first. The next few pages will show some sample coding from the system. Figure 5.1 shows the coding for the index page that the user will view upon accessing the PBLIS.

```
<?PHP
   require_once("config.php");
   require_once("course/lib.php");
   require_once("mod/resource/lib.php");
   require_once("mod/forum/lib.php");
   if (! $site = get_site()) {
      redirect("$CFG->wwwroot/admin/index.php");
   }
   if (empty($USER->id)) {
      $loginstring = "<font size=2><a href=\"$CFG-
>wwwroot/login/index.php\">".get_string("login")."</a></font>";
   } else {
      $loginstring = "<font size=1>".user_login_string($site)."</font>";
   }
   if (empty($CFG->langmenu)) {
      $langmenu = "";
   } else {
      $currlang = current_language();
      $langs = get_list_of_languages();
      $langmenu = popup_form ("$CFG->wwwroot/?lang=", $langs, "chooselang",
$currlang, "", "", "", true);
   }
   print_header("$site->fullname", "$site->fullname", "home", "",
            "<meta name=\"description\" content=\"".s(strip_tags($site->summary))."\">",
            true, "", "<div align=right>$loginstring$langmenu</div>");
   $firstcolumn = false;
   $side = 175;
?>
<table width="100%" border="0" cellspacing="5" cellpadding="5">
 <tr>
 <?PHP
   $sections = get_all_sections($site->id);
   if ($site->newsitems > 0 or $sections[0]->sequence or isediting($site->id) or
isadmin()) {
      echo "<td width=\"$side\" valign=top nowrap>";
      $firstcolumn=true;
      if ($sections[0]->sequence or isediting($site->id)) {
         get_all_mods($site->id, $mods, $modnames, $modnamesplural,
$modnamesused);
         print_section_block(get_string("mainmenu"), $site, $sections[0],
                     $mods, $modnames, $modnamesused, true, $side, isediting($site-
>id));
      }
      if (isadmin()) {
         echo "<div align=\"center\">".update_course_icon($site->id)."</div>";
         echo "<br />";
```

```
    }
    if ($site->newsitems > 0 ) {
        $categories = get_categories();
        if (count($categories) > 1) {
            print_course_categories($categories, "none", $side);
        } else {
            $category = array_shift($categories);
            print_all_courses($category->id, "minimal", 10, $side);
        }
    }
    print_spacer(1,$side);
}
if (iscreator()) {
    if (!$firstcolumn) {
        echo "<td width=\"$side\" valign=top nowrap>";
        $firstcolumn=true;
    }
    print_admin_links($site->id, $side);
}
if ($firstcolumn) {
    echo "</td>";
}
echo "<td width=\"70%\" valign=\"top\">";
if ($site->newsitems == 0 ) {
    print_heading_block(get_string("availablecourses"));
    print_spacer(8,1);
    $categories = get_categories();
    if (count($categories) > 1) {
        print_course_categories($categories, "index");
    } else {
        print_all_courses("all");
    }
} else {
    if (! $newsforum = forum_get_course_forum($site->id, "news")) {
        error("Unable to create main forum");
    }
    if (isset($USER->id)) {
        $SESSION->fromdiscussion = "$CFG->wwwroot";
        if (forum_is_subscribed($USER->id, $newsforum->id)) {
            $subtext = get_string("unsubscribe", "forum");
        } else {
            $subtext = get_string("subscribe", "forum");
        }
        $headertext = "<table border=0 width=100% cellpadding=0 cellspacing=0><tr>
                <td>$newsforum->name</td>
                <td align=right><font size=1>
```

```
                <a href=\"mod/forum/subscribe.php?id=$newsforum-
>id\">$subtext</a>
                </td></tr></table>";
        } else {
            $headertext = $newsforum->name;
        }
        print_heading_block($headertext);
        print_spacer(8,1);
        forum_print_latest_discussions($newsforum->id, $site->newsitems);
    }
    echo "</td>";
    echo "<td width=30% valign=top>";
    if (isediting($site->id)) {
        $site->summary .= "<br><center><a href=\"admin/site.php\"><img
src=\"pix/i/edit.gif\" border=0></a>";
    }
    print_simple_box($site->summary, "", "100%", $THEME->cellcontent2, 5,
"siteinfo");
    print_spacer(1,$side);
    echo "</td>";
 ?>
 </tr>
</table>
<?PHP print_footer("home");     ?>
```

Figure 5.1 : Coding for index.php

Another sample of coding is shown in Figure 5.2 for the administrators page view.

```
<?PHP
  if (!file_exists("../config.php")) {
      echo "<H2 align=center>Config.php not found.<BR></H2>";
      die;
  }
  require_once("../config.php");
  if (!check_php_version("4.1.0")) {
      $version = phpversion();
      print_heading("Sorry, PBLIS requires PHP 4.1.0");
      die;
  }
  if (ini_get_bool('session.auto_start')) {
      error("The PHP server variable 'session.auto_start' should be Off");
  }
```

```
if (ini_get_bool('magic_quotes_runtime')) {
   error("The PHP server variable 'magic_quotes_runtime' should be Off");
}
if (!ini_get_bool('magic_quotes_gpc')) {
   error("The PHP server variable 'magic_quotes_gpc' is not turned On");
}
if (!ini_get_bool('file_uploads')) {
   error("The PHP server variable 'file_uploads' is not turned On");
}
if (!ini_get_bool('short_open_tag')) {
   error("The PHP server variable 'short_open_tag' is not turned On");
}
if (!is_readable(ini_get('session.save_path')) and !ini_get_bool('safe_mode')) {
   $sessionpath = ini_get('session.save_path');
   notify("Sessions not supported (session.save_path = '$sessionpath')");
}
if ($CFG->wwwroot == "http://pblis.com/pblis") {
   error("Error in PBLIS");
}
$dirroot = dirname(realpath("../config.php"));
if ($dirroot != $CFG->dirroot) {
   error("Please fix your settings in config.php:
       <P>You have:
       <P>\$CFG->dirroot = \"".addslashes($CFG->dirroot)."\";
       <P>but it should be:
       <P>\$CFG->dirroot = \"".addslashes($dirroot)."\";",
       "./");
}
if (!isset($CFG->framename)) {
   $CFG->framename = "_top";
}
if (!isset($CFG->release)) {
   $CFG->release = "";
}
if (!isset($CFG->version)) {
   $CFG->version = "";
}
if (! $tables = $db->Metatables() ) {
   $maintables = false;
} else {
   $maintables = true;
   $mtables = array("config", "course", "course_categories", "course_modules",
              "course_sections", "log", "log_display", "modules",
              "user", "user_admins", "user_students", "user_teachers");
   foreach ($mtables as $mtable) {
      if (!in_array($CFG->prefix.$mtable, $tables)) {
```

```
            $maintables = false;
            break;
        }
    }
}
if (! $maintables) {
    if (empty($agreelicence)) {
        $strlicense = get_string("license");
        print_header($strlicense, $strlicense, $strlicense, "", "", false, " ",
" ");
        print_heading("<A HREF=\"http://pblis.com\">PBLIS</A> - Problem Based
Learning Informatin System,");
        print_simple_box_start("center");
        echo text_to_html(get_string("gpl"));
        print_simple_box_end();
        echo "<br />";
        exit;
    }
    $strdatabasesetup   = get_string("databasesetup");
    $strdatabasesuccess  = get_string("databasesuccess");
    print_header($strdatabasesetup, $strdatabasesetup, $strdatabasesetup, "", "", false,
" ", " ");
    if (file_exists("$CFG->libdir/db/$CFG->dbtype.sql")) {
        $db->debug = true;
        set_time_limit(0);
        if (modify_database("$CFG->libdir/db/$CFG->dbtype.sql")) {
            $db->debug = false;
            notify($strdatabasesuccess, "green");
        } else {
            $db->debug = false;
            error("Main database error");
        }
    } else {
        error("Error");
    }
    print_continue("index.php");
    die;
}
include_once("$CFG->dirroot/version.php");
include_once("$CFG->dirroot/lib/db/$CFG->dbtype.php");
if ($CFG->version) {
    if ($version > $CFG->version) {
        $a->oldversion = $CFG->version;
        $a->newversion = $version;
        $strdatabasechecking = get_string("databasechecking", "", $a);
        $strdatabasesuccess  = get_string("databasesuccess");
```

```
        print_header($strdatabasechecking, $strdatabasechecking, $strdatabasechecking,
                     "", "", false, " ", " ");
        print_heading($strdatabasechecking);
        $db->debug=true;
        if (main_upgrade($CFG->version)) {
            $db->debug=false;
            if (set_config("version", $version)) {
                notify($strdatabasesuccess, "green");
                print_continue("index.php");
                die;
            } else {
                notify("Failed");
            }
        } else {
            $db->debug=false;
            notify("Failed");
        }
    } else if ($version < $CFG->version) {
        notify("Failed");
    }
} else {
    $strcurrentversion = get_string("currentversion");
    print_header($strcurrentversion, $strcurrentversion, $strcurrentversion,
                 "", "", false, " ", " ");
    if (set_config("version", $version)) {
        print_heading("PBLIS");
        print_continue("index.php");
        die;
    } else {
        $db->debug=true;
        if (main_upgrade(0)) {
            print_continue("index.php");
        } else {
            error("ERROR");
        }
        $db->debug=false;
    }
}
if ($release <> $CFG->release) {
    $strcurrentrelease = get_string("currentrelease");
    print_header($strcurrentrelease, $strcurrentrelease, $strcurrentrelease, "", "", false,
" ", " ");
    print_heading("PBLIS $release");
    if (!set_config("release", $release)) {
        notify("ERROR");
    }
```

```
    print_continue("index.php");
    print_simple_box_start("CENTER");
    include("$CFG->dirroot/lang/en/docs/release.html");
    print_simple_box_end();
    print_continue("index.php");
    exit;
}
if (!$mods = get_list_of_plugins("mod") ) {
    error("No modules available");
}
foreach ($mods as $mod) {
    if ($mod == "NEWMODULE") {
        continue;
    }
    $fullmod = "$CFG->dirroot/mod/$mod";
    unset($module);
    if ( is_readable("$fullmod/version.php")) {
        include_once("$fullmod/version.php");
    } else {
        notify("Module was not readable");
        continue;
    }
    if ( is_readable("$fullmod/db/$CFG->dbtype.php")) {
        include_once("$fullmod/db/$CFG->dbtype.php");
    } else {
        notify("Module was not readable");
        continue;
    }
    if (!isset($module)) {
        continue;
    }
    $module->name = $mod;
    if ($currmodule = get_record("modules", "name", $module->name)) {
        if ($currmodule->version == $module->version) {
        } else if ($currmodule->version < $module->version) {
            print_heading("$module->name module needs to be changed");
            $upgrade_function = $module->name."_upgrade";
            if (function_exists($upgrade_function)) {
                $db->debug=true;
                if ($upgrade_function($currmodule->version, $module)) {
                    $db->debug=false;
                    $module->id = $currmodule->id;
                    if (! update_record("modules", $module)) {
                        error("Could not update $module->name record in modules table");
                    }
                    notify(get_string("modulesuccess", "", $module->name), "green");
```

```
                echo "<HR>";
            } else {
                $db->debug=false;
                notify("Error");
            }
        }
        $updated_modules = true;
    } else {
        error("Error");
    }
} else {
    if (empty($updated_modules)) {
        $strmodulesetup   = get_string("modulesetup");
        print_header($strmodulesetup, $strmodulesetup, $strmodulesetup, "", "", false,
" ", " ");
    }
    print_heading($module->name);
    $updated_modules = true;
    $db->debug = true;
    set_time_limit(0);
    if (modify_database("$fullmod/db/$CFG->dbtype.sql")) {
        $db->debug = false;
        if ($module->id = insert_record("modules", $module)) {
            notify(get_string("modulesuccess", "", $module->name), "green");
            echo "<HR>";
        } else {
            error("$module->name FAILED");
        }
    } else {
        error("$module->name tables FAILED");
    }
  }
}
if (!empty($updated_modules)) {
  print_continue("index.php");
  die;
}
include_once("$CFG->dirroot/lib/defaults.php");
foreach ($defaults as $name => $value) {
  if (!isset($CFG->$name)) {
    $CFG->$name = $value;
    set_config($name, $value);
    $configchange = true;
  }
}
if (!empty($configchange)) {
```

```
    redirect("config.php");
  }
  if (! $site = get_site()) {
    redirect("site.php");
  }
  if (! record_exists("user_admins")) {
    redirect("user.php");
  }
  if (!iscreator()) {
    error("Admin or lecturer only", "$CFG->wwwroot/login/index.php");
  }
  $stradministration = get_string("administration");
  print_header("$site->shortname: $stradministration","$site->fullname:
$stradministration", "$stradministration");
  if (isadmin()) {
    $table->head  = array (get_string("site"), get_string("courses"), get_string("users"));
                $table->align = array ("CENTER", "CENTER", "CENTER");
                $table->data[0][0] = "<p><a
href=\"modules.php\">".get_string("managemodules")."</a></p>";
                if (file_exists("$CFG->dirroot/admin/$CFG->dbtype")) {
       $table->data[0][0] .= "<p><a href=\"$CFG-
>dbtype/frame.php\">".get_string("managedatabase")."</a></p>";
                }
                $table->data[0][1] = "<p><a
href=\"../course/edit.php\">".get_string("addnewcourse")."</a></p>".
                "<p><a href=\"teacher.php\">".get_string("assignteachers")."</a></p>".
                "<p><a
href=\"../course/delete.php\">".get_string("deletecourse")."</a></p>".
                "<p><a
href=\"../course/categories.php\">".get_string("categories")."</a></p>";
    if ($CFG->auth == "email" || $CFG->auth == "none" || $CFG->auth == "manual") {
                  $table->data[0][2] = "<p><a
href=\"user.php?newuser=true\">".get_string("addnewuser")."</a></p>";
    }
    $table->data[0][2] .=  "<p><a
href=\"user.php\">".get_string("edituser")."</a></p>".
                "<p><a href=\"admin.php\">".get_string("assignadmins")."</a></p>".
                "<p><a href=\"creators.php\">".get_string("assigncreators")."</a></p>";
  } else {
        $table->head  = array (get_string("courses"));
                $table->align = array ("CENTER");
                $table->data[0][1] = "<p><a
href=\"../course/edit.php\">".get_string("addnewcourse")."</a></p>".
                  "<p><a href=\"teacher.php\">".get_string("assignteachers")."</a></p>";
      }
  print_table($table);
```

```
    print_heading("PBLIS", "CENTER", 1);
    print_footer($site);
?>
```

Figure 5.2: Coding for admin.php

5.9 Integration

Modules that are coded separately are integrated to form the PBLIS system. As the modules are integrated, the system is tested. When integration has been completed, the entire system is then tested as a whole.

5.9.1 Module Integration

Integration methods employed are the incremental methods. In incremental development methods, some sections of the system are unit tested and integrated before other sections of the system are coded. Following that, the system is created in incremental steps. There are 3 major approaches to incremental integration; top-down, bottom-up and threads. The top-down method approach is used to integrate the modules of PBLIS.

In the top-down integration method, the command-level modules at the top of the system structure are coded and unit tested first. Then modules from level down are unit tested and added on until the entire system has been completed. Due to the integration of higher level modules first, the hierarchical structure of the system and the critical high level interfaces and functions are tested early.

5.10 Summary

In summary, the implementation of the Problem Based Learning Information System consists of many steps, such as interface creation and database creation but the most important phase of the implementation and development is the coding and linking of the system with the database.

CHAPTER 6: TESTING

6 Testing

6.1 Testing Techniques

Testing is an important process used in system development to discover a defect or a bug that is present in the system. Testing is the process of executing a program with the intent of finding a bug and identifying defects for subsequent correction and elimination. There are 2 testing technique namely, white box testing and black box testing.

6.1.1 White Box Testing

White box testing is a test case design method that uses the control structure of the procedural design to derive test cases as shown in Figure 6.1:

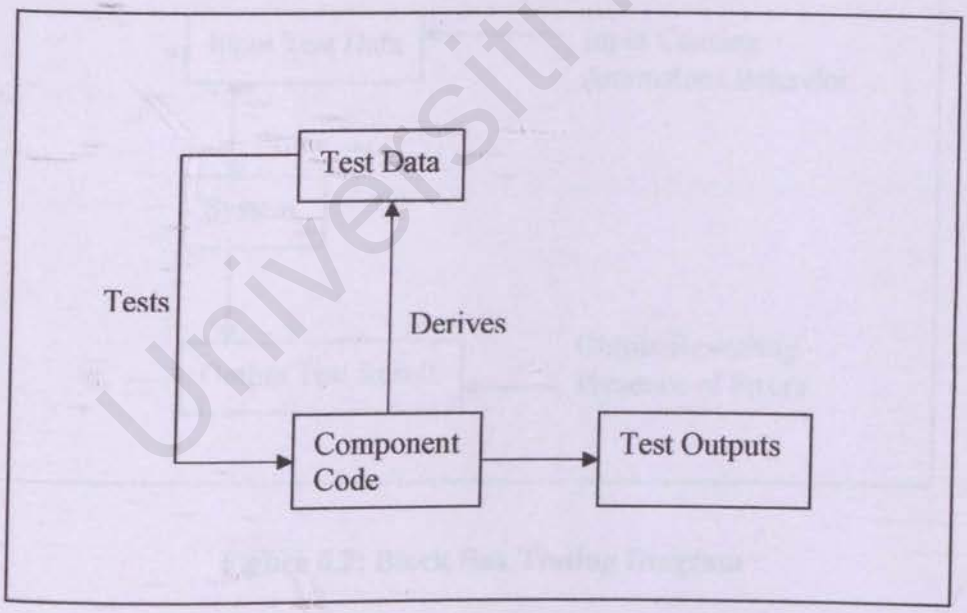


Figure 6.1: White Box Testing Diagram

It treats a product as a white box that is transparent with walls. Such testing allows detailed examination of the codes in order to create tests that will face execution at all possible branches of coding. It ensures that all possible paths and actions have been exercised while the system is under testing. In developing PBLIS, white box testing is implemented in module testing. Each code in the database is examined thoroughly using this technique.

6.1.2 Black Box Testing

Black box testing focuses on the functional requirements of the system as shown in Figure 6.2:

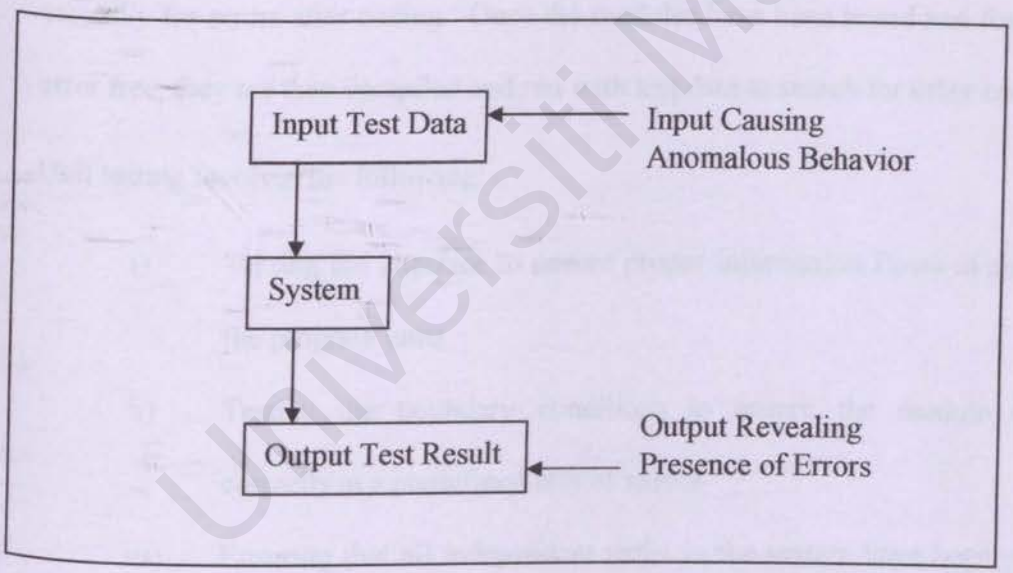


Figure 6.2: Black Box Testing Diagram

It treats the product as a black box that is an entity of which the tester is unable to see. This is considered a complementary approach to the white box testing to locate different errors. Black box testing is performed only at the system level to find malfunctions, interface errors, performance errors and initialization errors.

6.2 Testing Strategies

Testing was conducted during the entire development process of PBLIS. The testing consists of unit testing, integration testing and system testing.

6.2.1 Unit Testing

Unit testing focuses on the module, which is the smallest component of the software. Each module is tested individually, or independent of one another, to ensure their correctness. Functions and procedures in each module are examined carefully for errors after coding. Once the modules have been tested and found to be error free, they are then compiled and run with test data to search for other errors.

Unit testing involves the following:

i) Testing the interface to ensure proper information flows in and out of the program units

ii) Testing the boundary conditions to ensure the module operates correctly at a predefined sets of values

iii) Ensuring that all independent paths in the system have been executed at least once.

For the PBLIS, unit testing is done concurrently with the development phase. For example, the Lecturer module has a sub-module that manages PBL reports meant for management. Another example would be the student module which has a sub-

module that manages the course evaluation module. Each one of these functionalities is reviewed and checked separately. The sub-modules are then also tested to make sure that they function accurately. Once each section or modules have been tested individually the system will then be tested as a whole. In other works, the system will be reviewed as a complete functioning system. Table 6.1 shows examples of the methods and steps carried out to test the system and tabulate the system's responses to the test inputs.

Table 6.1: Test Input Steps and Results

Test Input	System Results
Input: Empty username and password. **Input:** Only empty password. **Input:** Only empty username. **Input:** Invalid username and password. **Input:** Valid username but invalid password.	**Output:** System informs user that the data is wrong and prompt the user to enter data again in order to login. **Result:** System responded correctly.
Input: Valid username and password.	**Output:** Administrator was able to access all modules, Lecturers were able to access and modify their respective class pages and Students were only granted access to the relevant courses they had signed up for. **Result:** System responded correctly.

6.2.2 Integration Testing

Integration testing is a systematic technique for constructing the program structure while at the same time conducting tests to uncover errors associated with interfacing. The objective is to take unit-tested sub-modules and build a module structure that has been dictated by design.

The integrated testing used in this PBLIS is the top-down approach. Each module at the highest level in the system hierarchy is tested individually. Using this approach the module that is next in line to be tested is the modules that called the previously tested module. During the process of testing, interfacing errors will be uncovered and fixed along the way. Considering that the system has been built by sections or modules, errors found can be corrected easily as its functions doesn't overlap significantly with the other modules. Therefore, any such changes to one module will not hamper or change the functionality of another module. Table 6.2 shows examples of the steps and methods performed to test the interface between all the modules.

Table 6.2: Test Input Steps and Results for Integration Testing

Test Input	System Results
Input: Multiple choice questions created is answered. **Input:** Marks for each student who has done the quiz is checked by the student and by the admin and lecturer	**Output:** Quiz results are stored in the database tables by the quiz module access by the students and marks automatically given by the system is able to be retrieved and viewed by the student, lecturer and administrator. **Result:** System responded correctly.

6.2.3 System Testing

System testing is a series of different tests whose primary purpose is to fully utilize all functions in the computer-based system. System testing ensures or verifies that the system solves the problem as defined by the requirements documents. Function testing comes after integrated testing to test or verify the integrated system to perform its function inline with its functionality requirements as specified in the requirements.

Once this segment of the testing is completed and each integrated system is found to function accordingly, a performance test is done to compare the integrated modules with the non-functional system requirements. These requirements include security, accuracy, speed and reliability.

The final step of system testing is the regression test. This test is done to the PBLIS to identify new errors that may be introduced as current ones are being corrected. A regression test is a test applied to a new version of a system to verify that it still performs the same functions in the same manner as the older version. For example, in the PBLIS, any changes made by the Lecturer Module should not hamper or influence the general function of the system calling in the main user software, such as the student course evaluation. Table 6.3 shows examples of tests to make sure no regression errors have occurred and to test the system as a whole.

Table 6.3: Test Input Steps and Results for System Testing

Test Input	System Result
Input: All buttons available in the system are clicked to check for correct functionality. **Input:** All forms are inserted with test text or test inputs and submitted.	**Output:** All buttons are functional and operate normally as is coded. **Result:** System responded correctly. **Output:** All forms were able to be submitted and relevant inputs are stored in the correct tables. **Result:** System responded correctly.

6.3 Debugging

Debugging is performed as a consequence of successful testing. When testing is done, and a test case is found with errors, the process or attempting to match the symptoms with the causes, which then leads to the correction of the error, is called debugging. The debugging approach employed was cause elimination. This form of debugging could identify the data related to the error occurrence and is organized and isolated to its potential causes. A list of all possible causes was developed and tests conducted to eliminate each. If initial tests indicate that a particular cause hypothesis shows promise, the data is refined in an attempt to isolate the error.

6.4 Summary

In summary, the testing of the system was relatively successful in weeding out errors. As each error occurred, appropriate error catching statements or coding were inserted to facilitate use for the user, if in the event the user inputs invalid text.

CHAPTER 7: SYSTEM EVALUATION

7 SYSTEM EVALUATION

7.1 Introduction

This is the final chapter of the report and details problems encountered and their solutions, the strengths and weaknesses of the system, possible future enhancements and experience gained from the development of the system.

7.2 Problems Encountered and Solutions

In the process of developing and implementing the system the following problems were encountered:

i. **Lack of experience and knowledge in system development** – Taking into consideration of the newness of PHP as a web coding language, mastery of the language was moderate but as more and more involvement in the development of the system surfaced, the familiarity of the language was easier to understand. Guidance, advice and criticism from individuals much more familiar in PHP were also received in good faith during the development of the system. References to technical books centered on the knowledge of coding PHP web pages were made and facilitated the learning and utilization of PHP as the main coding language. Furthermore, separate module development was dropped at the last minute due to inconsistencies in programming and coding. Due to these obstacles, the development of the Problem Based Learning Information System was hampered and delayed.

ii. **Difficulty in selecting the proper tools for system development** – With the unfamiliarity of available tools in coding PHP, a search for available PHP authoring tools was made and Zend Development Environment was available to download and use. The first tool that was selected to be used in the coding of PHP was Macromedia Dreamweaver but it proved to be much to troublesome in the compiling and previewing of PHP pages. In the end, Zend Development Environment was found and utilized. Macromedia Dreamweaver played a part in the designing of the forms and input buttons available in PBLIS. Once again, Zend Development Environment was recommended by individuals familiar in the coding of PHP.

7.3 System Strengths

Several key strength of the Problem Based Learning Information System has been identified.

7.3.1 User Friendly and Logical Flow of Interface

The interface of the PBLIS system is simple, clean and provides for easy navigation. With the consistency of each page and appropriate labeling and logical flow of information, the user can easily utilize the system.

The usage of appropriate colors and labels help the user navigate through the system. In simple terms, the system is built using Human-Computer Interaction concepts to ensure users are able to use the system easily.

7.3.2 Portability

Being a web based system; PBLIS boasts the advantage of being portable on most systems that utilize graphic enabled browsers. Usage of the system through the internet by way of the Apache Web-Server, users that utilize Internet Explorer, Netscape Navigator and other popular browsers are able to access and view the system.

7.3.3 Identification, Authentication and Security

PBLIS provides a strong security feature that allows only authorized users to access the system and database. The application requires user to login with password to access all the functions.

Multilevel access login is used in the PBLIS system, where the user ID and password are fixed in the database for Administrator, Lecturers and Students. Only authorized users with different levels of access can utilize the system. If a wrong combination of login or password is entered, the system will deny access. This is a part of authentication and security for the system to make sure no unauthorized persons to enter the system. Access to files by users with the inappropriate privileges is denied with the use of sessions and cookies.

7.3.4 Error Catching

Error catching or handling facilitates the usage of the system if a severe error has occurred. Error handing allows the system to display the error and attempt to recover from the server error rather and informing the user on what has happened. As PBLIS is a web based system, any errors that occur can be notified to the system administrator.

7.3.5 Internal Forum Search Engine

The search engine provides users with the ability to search through forums for particular information. The search engine is utilized to search for specific words or resource links contained in forums.

7.3.6 Plug-in or Module Addition Capability

A separate module that handles possibly future enhancement of the system is available. The current modules may not be sufficient in the future but this feature of adding modules and deleting modules allows the evolvement of the system by way of adding more functionality using additional modules.

7.4 Limitations of PBLIS

Although much effort has been put into making sure that the best possible system is developed, there still exist limitations that are unable to be avoided due to constraints of time and mastery of coding. The following are the limitations of PBLIS.

7.4.1 Database Backup Is Unavailable

PBLIS does not offer backup capability of its database. In the event of a severe error to the database, all information stored is lost. This is a severe limitation of PBLIS. If the backup function was to be implemented, PBLIS needs to terminate any connections to the database. Therefore additional coding is needed to perform this function. Unfortunately, the mastery of PHP is not sufficient to add this function.

7.4.2 Mailing Features Are Unavailable

While the system does store e-mail addresses of all users in the system, a mailing feature was not added. This is due to the unavailability of a POP (Post Office Protocol) server which is needed to implement the mailing feature. Ideally the system should mail updates to participants in a forum or in a designated class.

7.4.3 Full User Validation

The system lacks a full user validation, meaning that all students' courses and information can be added into the database manually. Therefore, the addition of extra users, be it student, lecturer or administrator can be done through the system. Therefore validation of users must be done manually, although users entering the system will be compared to the list of users in the database.

7.4.4 Graph and Chart Creation

Reports that should include graph bars or charts are unable to be done due to the lack of mastery in using the GD library function within PHP. The GD library function in PHP actually allows users to manipulate GIF or PNG graphic file formats. Therefore a graph chart or bar chart for tabulating surveys are unable to be created.

7.5 Future Enhancements

After studying the limitation of PBLIS, the following are some suggested potential improvements that can be applied to the system in order to enhance it.

7.5.1 Backup Feature

PBLIS requires a backup function to ensure all the database information can be kept safe and secure in the event of a severe database error. The current PBLIS lacks this feature. PBLIS should be able to backup all information stored within the database to a separate server or into some form of physical storage meaning disks or compact discs.

7.5.2 Mailing Feature

PBLIS can be improved by adding a mailing feature which is able to update relevant users, students or lecturer, to any new changes or updates found on the site. In order to add this new enhancement, a POP server should be available to route emails to

users or the Problem Based Learning Information System. This feature is very useful if there are any important announcements made on selected forums.

7.5.3 Additional Modules

Additional modules can be coded in the future to add more functionality such as graph creation or user grouping features. This can currently be done by creating modules that can be added in to the module directory contained within PBLIS. For instance, a module created to manipulate graphs can be inserted in the future by more experienced system or software developers.

7.6 Summary

As a whole the development of the Problem Based Learning Information System was a very eventful and successful experience, whereby new skills and mastery of the tools were learnt along the way. Although the system lacks in a few areas, the main functionality is available to be improved on.

7.7 Conclusion

In conclusion, the development and implementation of the Problem Based Learning Information System has achieved its objectives and requirements as a web-based application that facilitates the communication and collaboration among lecturers, among students and among lecturers and students. More users are relying on IT to solve their problems and reduce workloads. With the completion of the project, the facilitation

of work in FCSIT will be reduced. For example, users are able to manage their assignment submission, class evaluation and so forth with ease.

Much effort has been put into researching and studying other similar systems to adapt their functionality into PBLIS. Throughout the development of the system, knowledge of setting up the Apache Server, MySQL database server and coding in PHP has been gained. The experience and knowledge gained during the development of the project will no doubt prove to be a valuable asset in the future.

APPENDIX A

Sample Questionnaire

Our names are Tan Chee Haw WET 000294 and Nicholas Barau Eddy WET 000325, and we are conducting this survey to help us in the creation and implementation of a proposed Problem Based Learning Information System for the course WXES 3181, Latihan Ilmiah I. Any input is greatly appreciated. It would be greatly appreciated if this questionnaire could be left outside your door by the 30th April.Thank you for your cooperation.

1) Pick 3 most important features that you require in a Problem Based Learning Information System:

   - ☐ Journal
   - ☐ Problem Based Learning report management
   - ☐ Student work submission
   - ☐ Discussion Area / Forum
   - ☐ Online Peer & Course Evaluation
   - ☐ Others: _____

2) When grading group assignments, do you prefer to:-

   - ☐ Assign marks per group, or,
   - ☐ Assign marks per individual in a group, or,
   - ☐ Others: _____

3) Would you prefer to type a Problem Based Learning Report in

   - ☐ Document form (e.g. word processor), or,
   - ☐ Online Template, or,
   - ☐ Others: _____

4) For Course Evaluation and Peer Evaluation forms, would you prefer a

   - ☐ Customizable template / form, or,
   - ☐ Standard template / form, or,
   - ☐ Others: _____

5) When assigning groups, would you prefer

   - ☐ To assign groups based on a class list (e.g. alphabetically), or,
   - ☐ Students to group themselves, or,

Others: _____

For Questions 6 – 10, For 1 being not important at all and 5 being very important

6) Problem Based Learning report creation and management

1 2 3 4 5

7) Online student work submission

1 2 3 4 5

8) Student journals to track student progress

1 2 3 4 5

9) Online Peer & Course Evaluation

1 2 3 4 5

10) Discussion Area for Student – Student and Student – Lecturer interaction

1 2 3 4 5

11) Any additional comments please feel free to specify

_____

_____

Your help was greatly appreciated.
Thank You

USER MANUAL

Table of Contents

List of Figures

About the Manual

The user manual was created for facilitating the use of PBLIS amongst lecturers. As most of the functions in PBLIS are self explanatory, the core and basic functions of PBLIS will be documented highlighting functions by administrative and lecturer access privileges

The user manual is divided into 2 parts, the minimum requirements and instructions on using the administrative functions. Instructions accessing the system with student access will not be covered due to the simplicity of the user interface and logical flow of thought and that most functions in PBLIS are self-explanatory.

1 Hardware and Software Requirements

1.2 Hardware Requirements

This project requires the following components which are listed in Table 1.1.

Table 1.1: Hardware Requirements

Hardware Requirements	Minimum Requirements	Suggested Requirements
Processor	Pentium I (166 MHz)	Pentium III (500 MHz)
RAM	64MB	128MB
Disk Space	5GB	20GB
Operating System	Windows 95	Windows XP

1.3 Software Requirements

These are the following software required:

i. ASP

ii. Macromedia Dreamweaver

iii. MySQL

2 Using the System As Administrator

2.1 Logging In As Administrator

Figure 2.1 below shows the interface of the first screen seen when accessing the system.

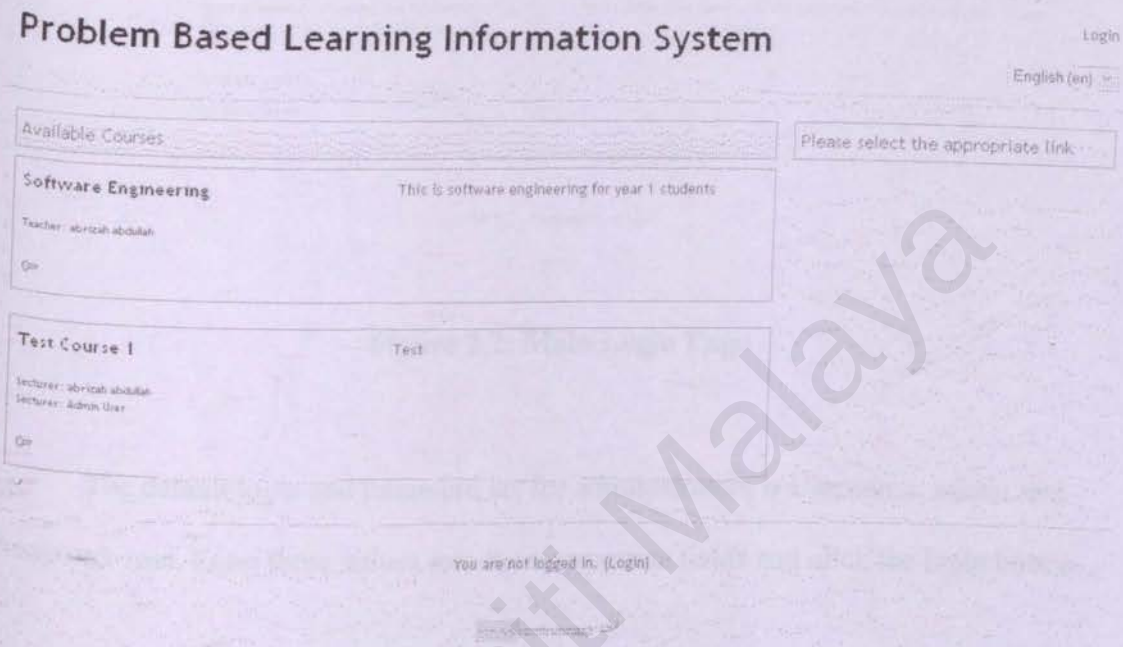


Figure 2.1: Starting Page

Select the appropriate course you would like to login to, or simply click the Login button located on the top right or at the bottom center of the screen.

Figure 2.2 illustrates the main login page of the PBLIS.

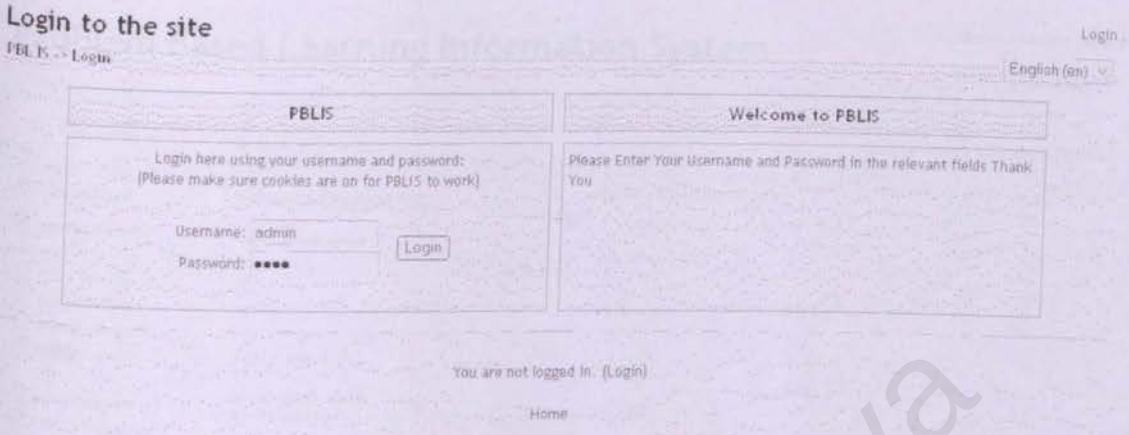


Figure 2.2: Main Login Page

The default login and password set for administrators is Username: admin and Password: root. Enter these values into the appropriate fields and click the login button.

Figure 2.3 shows the main administration page.

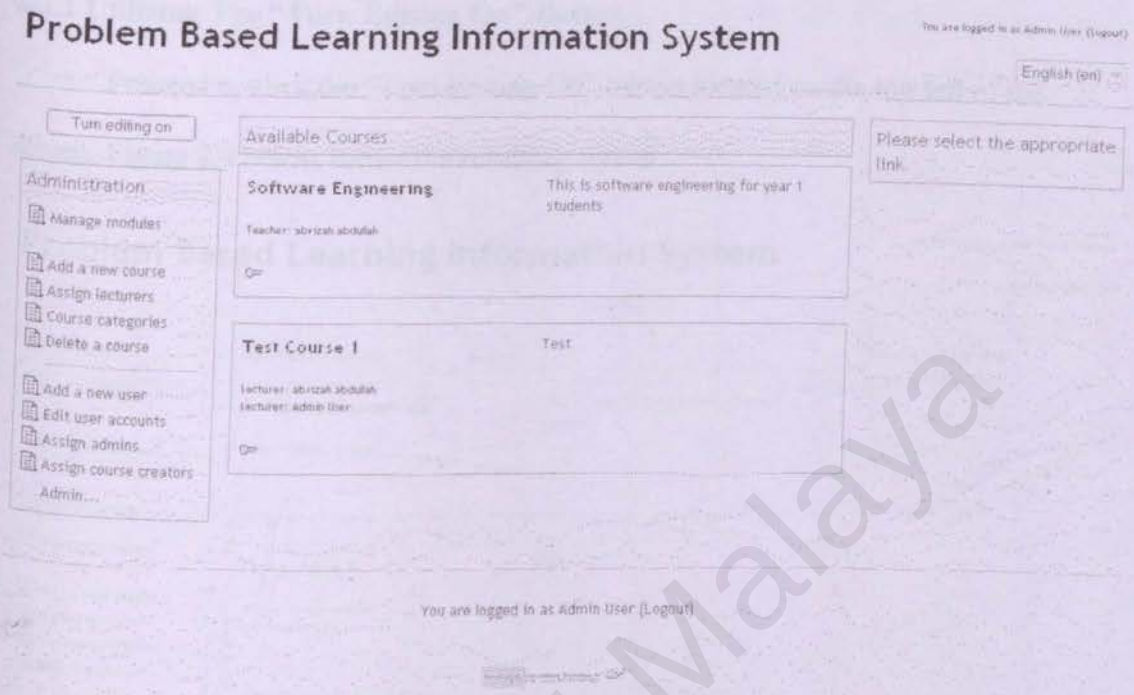


Figure 2.3: Main Administration Page

Select the appropriate link or functionality you wish to use as an administrator.

2.2 Utilizing Available Functions

2.2.1 Utilizing The "Turn Editing On" Button

Proceed to click the "Turn Editing On" button located on the top left of the screen. Figure 2.4 below shows the resulting screen.

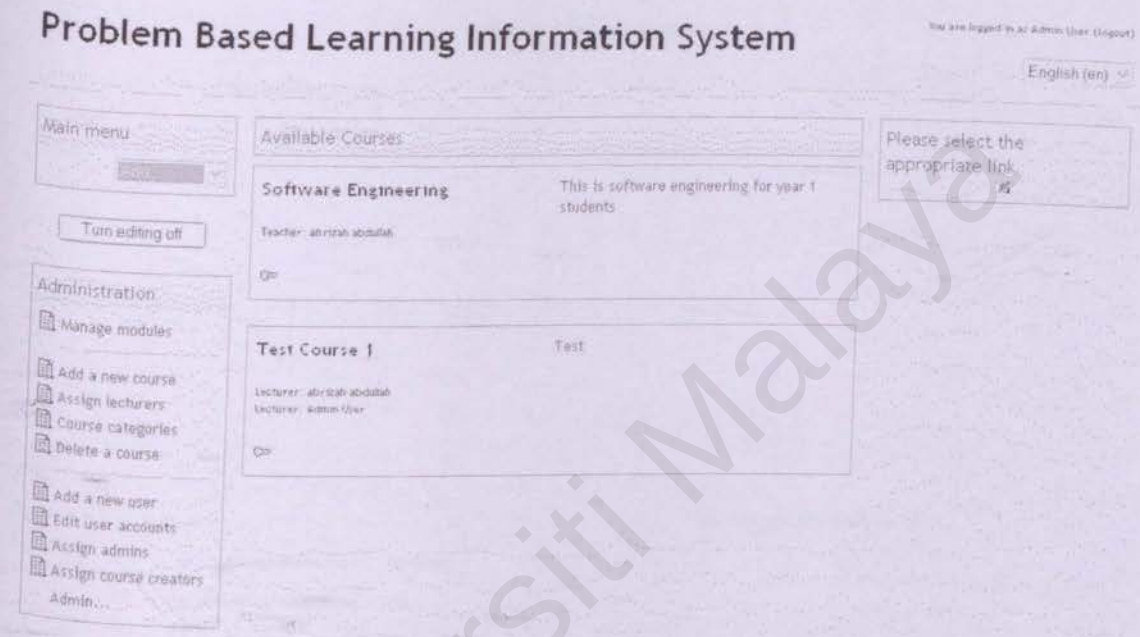


Figure 2.4: "Turn Editing On" Enabled

The administrator can now proceed to add functional modules to the main menu of PBLIS such as Assignment, Forum, Journal, Quiz, Resource and Survey from the drop down box at the top left of the screen.

2.2.2 Adding An Assignment

Make sure the "Turn Editing On" button has been clicked. Select "Assignment" from the drop down box. The following screen shown in Figure 2.5 will now appear.

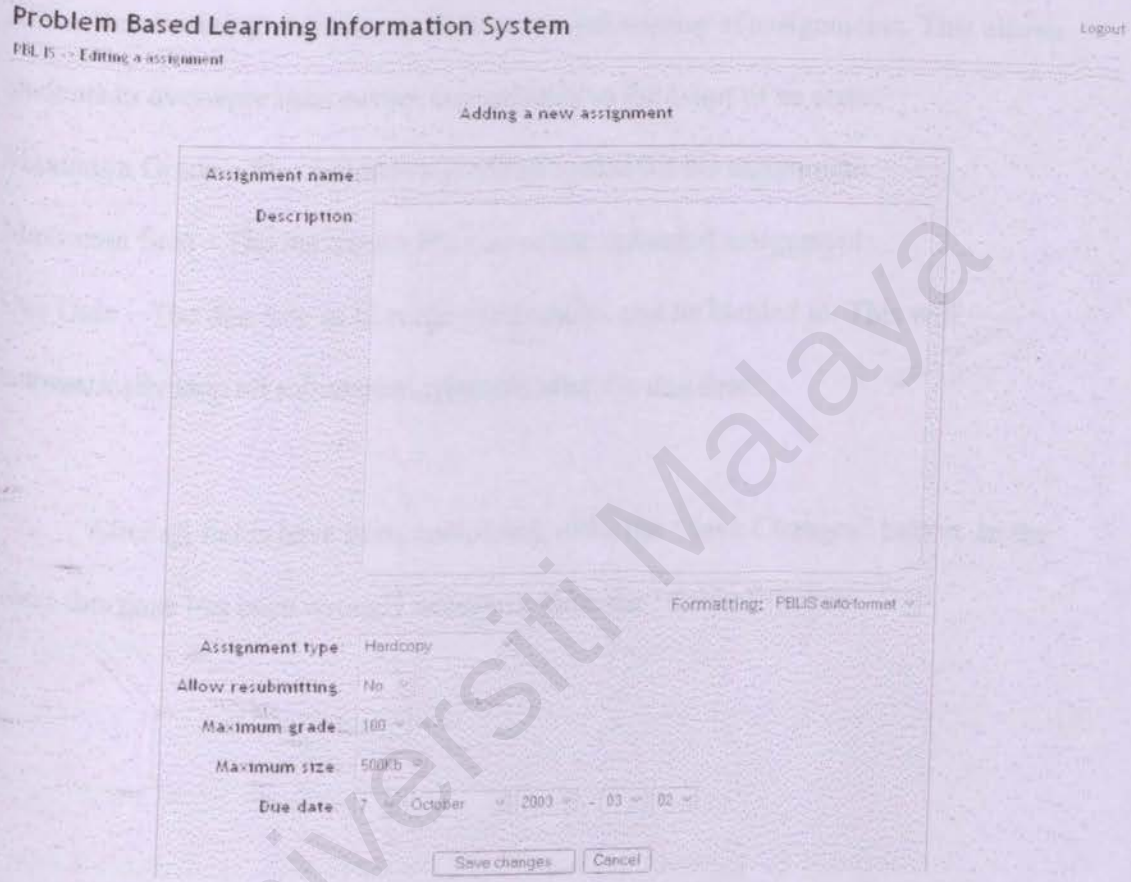


Figure 2.5: Adding An Assignment

Fill in and select the appropriate choices from all fields.

Assignment Name – The name of the assignment you wish to add

Description – The description of the assignment

Formatting – You are able to select from "PBLIS auto-format", "HTML format" and "Plain Text Format". Default is set to PBLIS auto-format. HTML allows addition of HTML tags. Plain Text format only allows regular text.

Assignment Type – You are able to select from "Hardcopy" and "Upload a Single File". "Hardcopy" informs students to physically hand in the assignment. "Upload a Single File" allows students to upload their assignment document online.

Allow Resubmitting – Allows or disallows resubmitting of assignments. This allows students to overwrite their earlier assignments in the event of an error.

Maximum Grade – The maximum grade allocated for the assignment.

Maximum Size – The maximum file size of the uploaded assignment.

Due Date – The due date as to when assignments can be handed in. This will automatically stop all submission attempts after the due date.

After all fields have been completed, click the "Save Changes" button. In the event this page has been wrongly accessed, click the "Cancel" button.

2.2.3 Adding A Forum

Make sure the "Turn Editing On" button has been clicked. Select "Forum" from the drop down box. The following screen shown in Figure 2.6 will now appear.

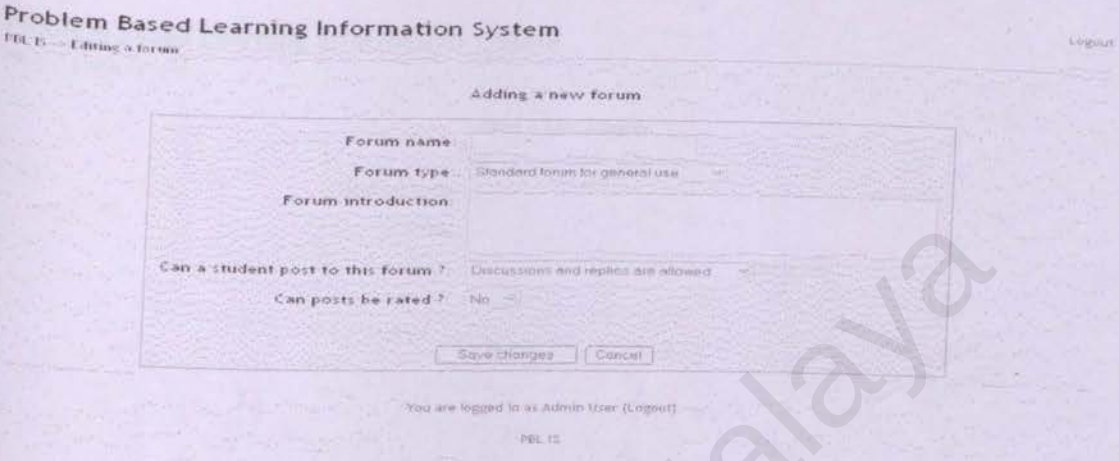


Figure 2.6: Adding A Forum

Fill in and select the appropriate choices from all fields.

Forum Name – The name of the forum about to be added

Forum Type – You are able to select from "Standard Forum for General Use", "A Single Simple Discussion" and "Each Person Posts One Discussion".

Forum Introduction – The introduction to what the forum should be about

Can A Student Post To This Forum – You are able to select from "Discussion and Replies Are Allowed", "No Discussions, but Replies Are Allowed" and "No Discussions, No Replies". "Discussion and Replies Are Allowed" lets users start discussions and post replies. "No Discussions, but Replies Are Allowed" does not let users start discussion but does let users reply to posts. "No Discussions, No Replies" does not let the user start discussions or reply to post; useful in creating announcements.

Can Posts Be Rated – Allows or disallows posts to be rated by users.

2.2.4 Adding A Journal

Make sure the "Turn Editing On" button has been clicked. Select "Journal" from the drop down box. The following screen shown in Figure 2.7 will now appear.

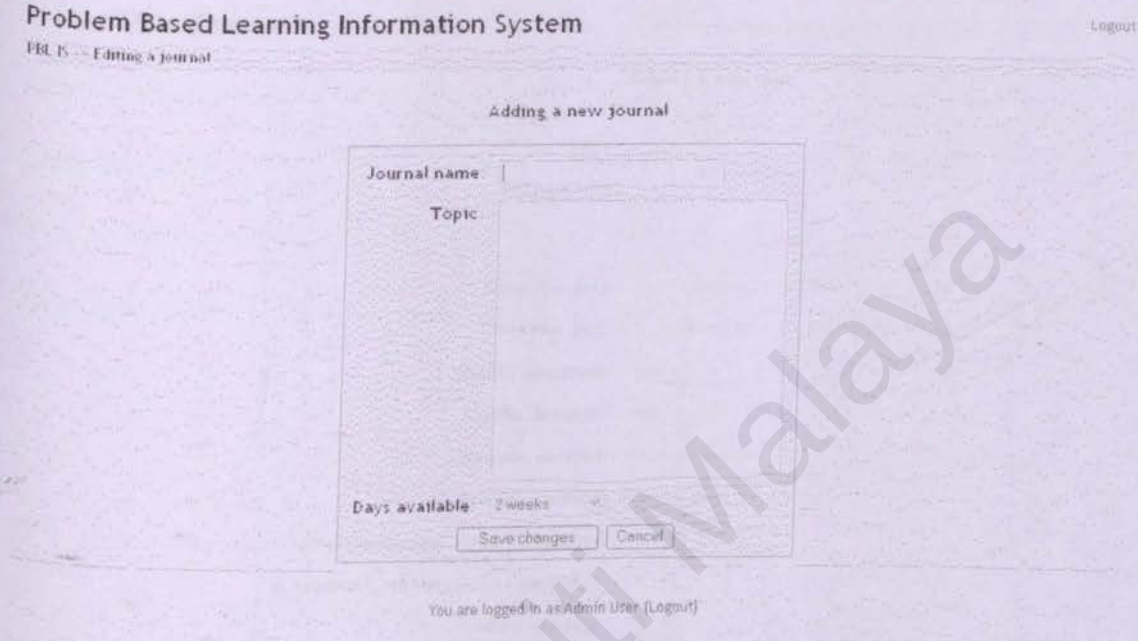


Figure 2.7: Adding A Journal

Fill in and select the appropriate choices from all fields.

Journal Name – The name of the journal.

Topic – Topic of what is to be written in the journal

Days available – The time frame of when the journal is available.

2.2.5 Adding A Quiz

Make sure the "Turn Editing On" button has been clicked. Select "Quiz" from the drop down box. The following screen shown in Figure 2.8 will now appear.

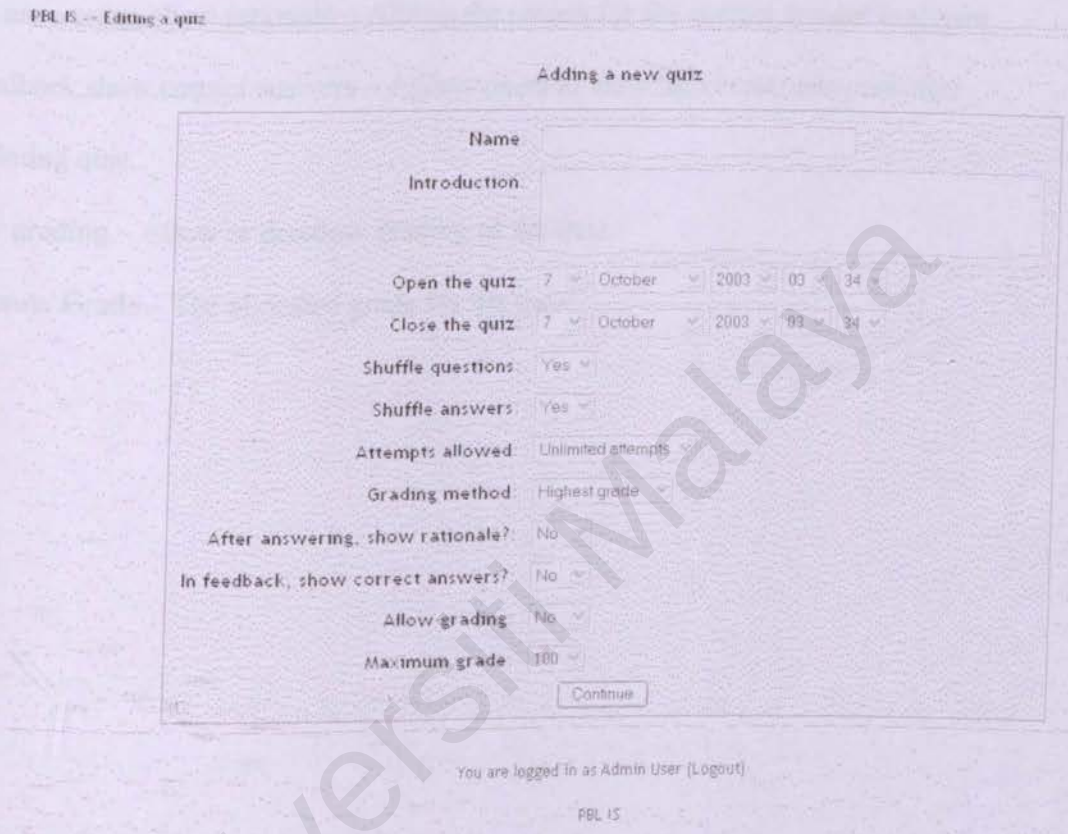


Figure 2.8: Adding A Quiz

Fill in and select the appropriate choices from all fields.

Name – Name of the quiz

Introduction – Introduction to the quiz

Open the quiz – The date the quiz will be available to be attempted

Close the quiz – The date the quiz will be unavailable

Shuffle Questions – Allow questions to be shuffled

Shuffle Answers – Allow answers in for MCQ's to be shuffled

Attempts Allowed – Specify number of attempts can be made on the quiz.

Grading Method – You are able to select from "Highest Grade", "Average Grade", "First Attempt" and "Last Attempt".

After answering, show rationale – Allows the reason for the correct answer is shown

In feedback show correct answers – Allows users to view the correct answers after completing quiz.

Allow grading – Allow or disallow grading of the quiz.

Maximum Grade – The allocated grade for the quiz

2.2.6 Adding A Resource

Make sure the "Turn Editing On" button has been clicked. Select "Resource" from the drop down box. The following screen shown in Figure 2.9 will now appear.

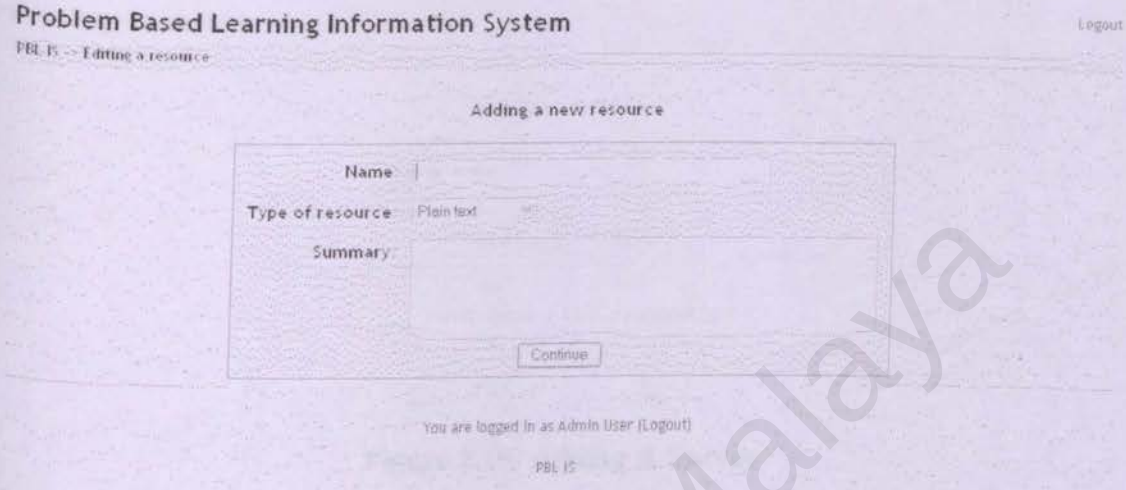


Figure 2.9: Adding A Resource

Name – Resource Name

Type – The type of resource either a web link, file, plain text and so forth

Summary – About the resource

2.2.7 Adding A Survey

Make sure the "Turn Editing On" button has been clicked. Select "Quiz" from the drop down box. The following screen shown in Figure 2.10 will now appear.

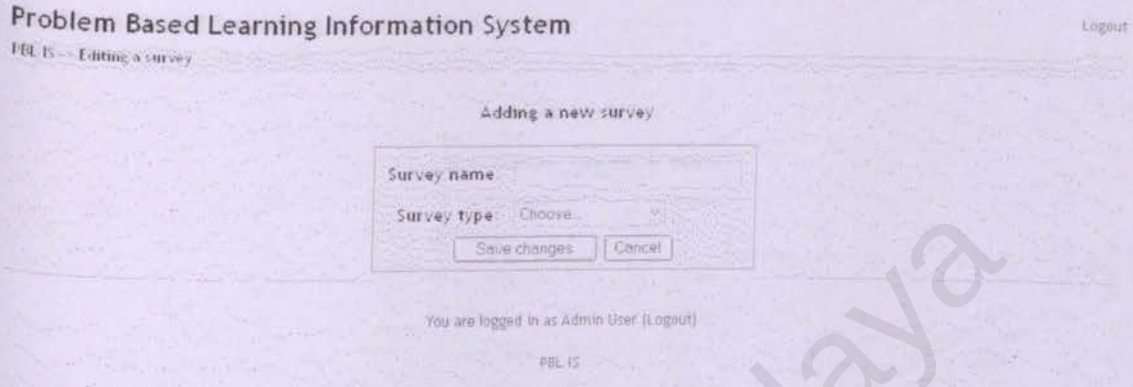


Figure 2.10: Adding A Survey

Survey name – The name of the survey

Survey type – Select the type of survey, e.g. class evaluation, peer evaluation

2.2.8 Managing Modules

Select "Manage Modules" from the main menu. The following screen shown in Figure 2.11 will now appear.

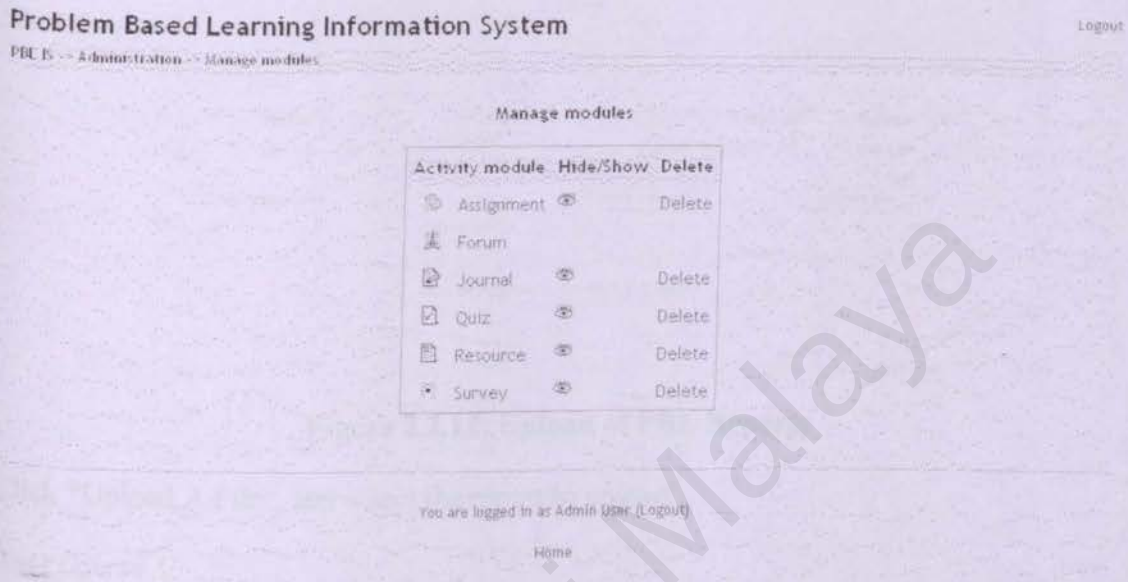


Figure 2.11: Manage Module Function

This allows the deletion of modules in the event they have outlived their usefulness.

2.2.9 Uploading PBL Reports

Select "Files" from the main menu. Proceed to "Upload File". The following screen shown in Figure 2.2.12 will now appear.

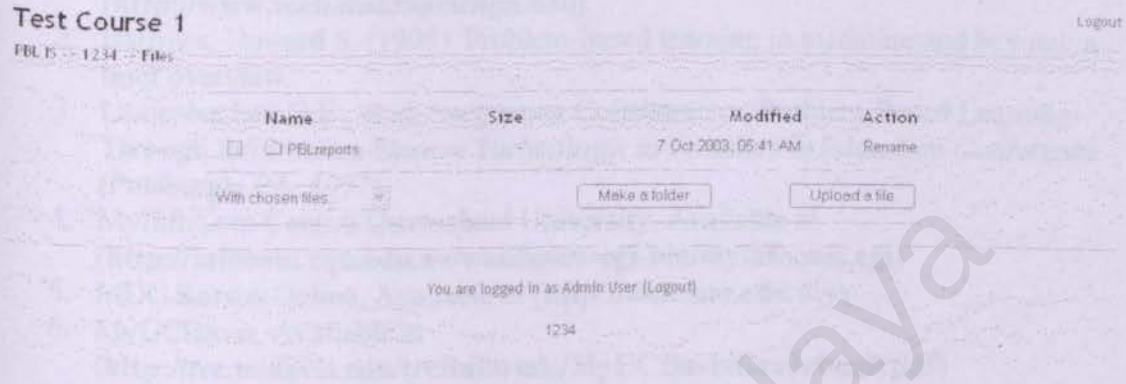


Figure 2.2.12: Upload of PBL Reports

Click "Upload A File", and select the report to upload.

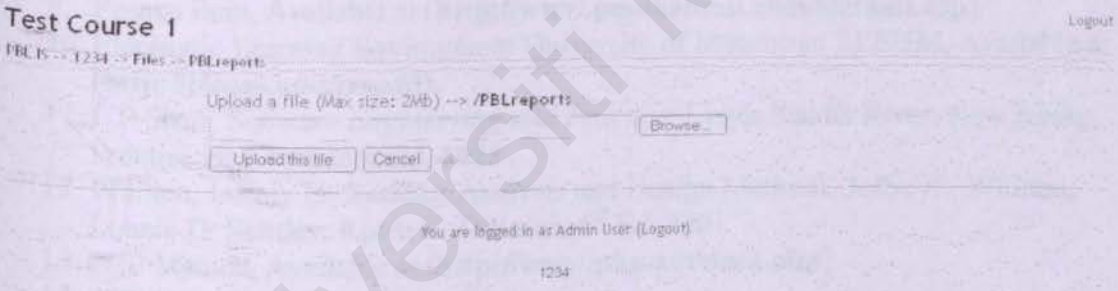


Figure 2.2.13: Upload of PBL Report

References

1. Maricopa Centre of Learning and Instruction, Available at **(http://www.mcli.dist.maricopa.edu)**
2. Barrows, Howard S. (1996). Problem-based learning in medicine and beyond: a brief overview.
3. Lautenbacher, G.E., et al. Supporting Collaborative, Problem-Based Learning Through Information System Technology, in Frontiers in Education Conference (Pittsburgh, PA, 1997)
4. MyInfoCom-Central Queensland University, Available at **(http://infocom.cqu.edu.au/webfuse/s-cgi-bin/myinfocom.cgi)**
5. MDC Kursus Online, Available at **(http://mdc.um.edu.my)**
6. MyUCDavis, Available at **(http://trc.ucdavis.edu/trc/tutorials/MyUCDavisGradebook.pdf)**
7. UC Berkeley E-Grades, Available at **(http://registrar.berkeley.edu/Acad/egradeinst.pdf)**
8. Precision Conference Solutions, Available at **(http://www.precisionconference.com)**
9. Pesaka Ilmu, Available at **(http://www.pesakailmu.com/Default.asp)**
10. Electronic Learning Environment University of Maastricht ELEUM, Available at **(http://eleum.unimaas.nl)**
11. L.P Shari, S*oftware Engineering and Practice.* Upper Saddle River, New Jersey, Prentice Hall International, 1998
12. Whitten, Jeffrey L., Systems Analysis and Design Methods. JeffreyL. Whitten, Lonnie D. Bentley, Kevin C. Dittman. 4th Ed. 2001
13. PHP Manual, Available at **(http://www.php.net/docs.php)**
14. ASP Overview, Available at **(http://msdn.microsoft.com/library/default.asp?URL=/library/en-us/dnasp/html/ASPover.asp)**
15. JSP Information, Available at **(http://java.sun.com/products/jsp/)**
16. Microsoft SQL Server 7.0 Overview, Available at **(http://www.wsbtech.com/mssql.htm)**
17. About Microsoft Products, Available at **(http://www.microsoft.com/products/)**
18. MySQL, Available at **(http://www.mysql.net)**
19. PC Review, Available at **(http://www.pcreview.com)**