**DESIGN AND IMPLEMENTATION OF AN ONLINE ASSIGNMENT SUBMISSION MANAGEMENT SYSTEM   
A CASE STUDY OF FEDERAL UNIVERSITY DUTSE (FUD).**

**BY**

**MUHAMMAD NURA ADAM FSC/CSC/16/1045**

**A PROJECT SUBMITTED TO THE DEPARTMENT OF**

**COMPUTER SCIENCE**

**FEDERAL UNIVERSITY DUTSE (FUD).**

**IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR**

**THE AWARD OF BACHELOR DEGREE IN COMPUTER SCIENCE**

**AUGUST, 2020**

**ABSTRACT**

The paper examines the design and implementation of an online assignment submission system. The challenges faced by students in meeting assignment deadlines and cost associated with printing hard copies of paper, necessitated the researcher to develop a user friendly system to tackle these challenges. The system allows students to submit assignments online to a particular course lecture, who have access to the system for grading purpose. The system was developed using PHP and MYSQL technologies.

**CHAPTER ONE**

**1.0 INTRODUCCTION**

An online assignment handling is a system contained within the Module virtual learning environment. The functionality of the standard assignment handling module has been extended to cater for all the Department's needs in terms of receiving assignments from students, making them available to tutors to mark, returning grades, comments and marked work to students and keeping Registry and course administrators informed at all stages of the process. Extension requests are an integral part of the system.

Universities, Polytechnics and colleges of education are considered the main provider of knowledge in various fields. Various courses of studies are taught in institutions, covering several fields including applied Sciences, Math, Computer, Human Resource, and Accounting. Most courses at polytechnic consist of theoretical as well as practical subject matter. To evaluate the level of understanding and degree of comprehension among students, assignments are often given.

Assignments are submitted by students either individually or in groups. Assignment management involves collecting, marking, and redistributing to students. Tregobov (1998) breaks the process down into four stages: submission, recording, marking, and return. Online assignment submission and management (OASM) involves the use of the World-Wide Web, the Internet and computers to aid this process (Jones, 2003). With traditional assignment submission system, lots of problems arise especially when the students have to submit the answers of the assignment to the lecturer. There may be problems due to distance, time, or format of the assignment (written or printed). Also every learning process requires administrative support. Much of this administrative support is to some degree transparent but if the latter is not well organized it could disrupt the flow of learning between students and the staff.

As the educational world is moving faster and becoming more competitive, almost every university and polytechnics started to use an online submission system, or newer technologies to facilitate their task, to have more time, and to be in pace with this fast moving IT world.

**1.2 STATEMENT OF THE PROBLEM**

Many reported failures in courses thought in the tertiary institutions can be attributed to the careless of the course representative or the teaching assistant who failed to submit an assignment paper to the lecturer for marking (Jones, 2003). Such careless could lead to a zero mark for the victim whose paper was declared missing or simply not found. The manual method of submitting assignments to the course lecturer or directly to the lecturer in most Nigerian Universities is simply not effective as these papers could get damaged or get missing due to the carelessness of the course representative or the lecturer.

Moreover, assignments that are large in terms of pages or volume could easily discourage a student from submitting due to financial constraints brought about by high cost of printing an assignment. All these problems highlighted are the main reasons the researcher is developing an electronic assignment submission system to curtail these challenges and make studying more enjoyable in our tertiary institutions.

1.3 **OBJECTIVES OF STUDY**

The main objective of the paper is to design and implement an online assignment submission system. Specific objectives of the study are:

i) To develop a system for proper documentation of students’ record in the in the school system.

ii) To Create a database that will manage each student assignment submission and allow access by lecturer to access those files submitted by the student.

iii) To Create a quick search and advance search that the student / lecturer when sorting a file.

iv) This proposed system is geared towards providing a system to assure equal opportunity and impartial review of student assignment submission

v) The system to be developed will provide a full school system by that will take cares of faculty registration, lecturer registration, course registration, department registration, users’ registration, student registration, perform scores /grades registration, test scores assessment and article entry.

**1.4 SIGNIFICANCE OF THE STUDY**

The study will aid lecturers to have a well-structured system for assignment submission and grading. This will eliminate paper work in their offices and improve on their efficiency in managing students’ assignments. Unlike attaching files in a mailing system like yahoo mail and Gmail and sending to a lecturer, this system systematically arranges assignments with respect to courses, departments and students ID, creating a more user friendly environment for both the students and the lecturers.

**1.5 ORGANIZATIONAL PROFILE**

Founded in 2011, the federal University Dutse has, for over some years, provided qualitative and research-oriented education to Nigerians and all those who have entered its domain in search of knowledge. The University has built a legacy of excellence and has been instrumental in the production of top range graduates and academia who have had tremendous impact, directly or indirectly, on growth and development in Nigeria.

**1.6 DEFINITION OF TERMS**

**Assignment:** a task or piece of work allocated to someone as part of a job or course of study

**Student:** A student or pupil is a learner, or someone who attends an educational institution.

**Online: online**" indicates a state of connectivity.

**Lecturer:** A lecture is an oral presentation intended to present information or teach people about a particular subject, for example by a university or college teacher.

**Submission:** the action of presenting a proposal, application, or other document for consideration or judgment.

**CHAPTER TWO**

**LITERATURE REVIEW**

Electronic submission of assignments has been widely used by information technology teachers for many years, taking the form of email submission, transferring files to a specified directory or providing marker access to directories and files on a unix system. A brief search of the web still reveals many examples where these practices are continued. They have had varying degrees of success but the following comment taken from an email list is quite revealing. “I was hoping to use an email-based submission system, but there are still a

For now, you will need to put your file in a directory on RCS (also accessible from CS workstations).”

A simple, modern form of electronic assignment marking is to use MS Word which is popular and affords some advantages over manual methods. Feedback comments can be provided through hidden text, the *Comment* facility or by using different fonts and colours. Changes can be examined through the *Track Changes* facility. This may be satisfactory for written word assignments but is totally inappropriate for computer programs, which are presented in pure text form so that they can be compiled and run.

There are many examples of more sophisticated uses of electronic-based assessment reported in the literature, ranging from online questionnaire generators to comprehensive systems which may include submission of assignments and Potatoes, 2002) includes six applications, enabling the user to create interactive multiple-choice, short-answer, jumbled-sentence, cross word, matching/ordering and gap-fill exe needed for programming assignments.

One of the most successful examples of electronic courseware in Information Technology is Course Master (Course Master, 2000), a client-sever system for delivering course-based programming. It provides functions for automatic assessment of students’ work, administration of the resulting marks, solutions and course materials. It is also able to detect plagiarism in students’ programs. A student is able to develop a program, submit it to the server for marking or evaluation and get instant feedback. Course Master is a complete reimplementation after 10 years of experience using the Ceilidh system (Foxley, 1999) in many institutions.

The BOSS system (Joy & Luck, 1998) is closer to OAS in terms of its intended use. This system allows programming assignments to be submitted online, tested against prescribed criteria and manually marked in a secure environment. Recently facilities have been added for electronic marking using ‘electronic mark sheets’. The project is driven by demands of increasing student numbers and is restricted to particular operating systems.

The OAS provides all of the functionality of BOSS and improves on this and other existing systems in a number of ways. Firstly, it was motivated to some extent by large class sizes but more so by the desire to provide more useful feedback to students, which has proved to be popular. Secondly, the OAS may be used across all units in a course of study and is not restricted to particular course materials like Course Master. Thirdly, OAS is web-based which represents a familiar and easy to use environment for students and is not constrained by certain operating systems. .Finally, lecturers find that the F2M2 marking program is easy to use and many have stated that it is time-saving compared to manual systems. F2M2 is a Java application, which means that it can run on any system with a Java Virtual Machine installed.

The electronic delivery of programming assignments also means that, before the files are submitted to F2M2, they have already been virus checked, are already compiled and may easily have been run against a test suite. A further advantage of the OAS is that it is a single integrated system unlike BOSS which is a suite of related programs.

All teaching and learning requires administrative support (Jones and McCormack, 1997). Much of this administrative support is to some degree transparent but if performed inefficiently become immediately obvious and can distract students and staff from the learning process (Darbyshire, 2000). Online Assignment Submission and Management (OASM) is one administrative task that can consume much of a course coordinator’s time (Darbyshire, 2000).

Reducing the amount of time consumed, addressing other problems, and benefiting from new advantages are some of the reasons which have driven many staff to adopt and use OASM. Darbyshire (2000) reports finding over 6000 hits from web search engines for pages describing online assignment submission procedures to students. In addition there is a large amount of literature written on the topic. This section outlines some of the reasons driving the adoption of OASM and provides an overview of much of this previous work.

**CHAPTER THREE**

**ANALYSIS AND DESIGN**

**3.0 INTRODUCTION**

This chapter discusses the analysis and design of the proposed system. It focus on the system structure and interactions. It begins with a description of the system using software UML diagrams such as the Use case and Activity diagrams. A program specification showing the flow of commands and segmenting the system into subroutines is also used to further describe the system. The database structure is also discussed with entity relationship diagrams (ERDs) showing the layout of the database.

**3.1 DESCRIPTION OF THE SYSTEM**

The proposed system is geared towards providing a system to assure equal opportunity and impartial review of student assignment submission, in the University of Dutse. The online assignment submission system to be developed

allows for assignment submission by students, faculty registration, lecturer registration, course registration, department registration, users registration, student registration, perform scores /grades registration, test scores assessment and article entry. The rest of this section describes the components and modules of the proposed system in terms of software UML diagrams.

**3.1.1 Use Case Diagram**

One can visualize high level system functions of requirements by drawing use case diagrams, which contain primary actors and use cases. Actors are entities that interact with the system while use cases are system functions that actors involve in.

Submit assignment

Search assignment

Add student

user

Delete assignment

***Fig. 3.1 USE CASE diagram for the proposed system***

As seen in Fig. 3.1, there is one actor involved in the system. This actor is normally the admin / lecturer.

**3.1.2 Activity Diagram**

Activity diagrams are graphical representations of work flows of step wise activities and actor. In an activity diagram, the flow of execution is modeled as activity nodes connected by activity edges.

Display UI

Enter Command

Display UI

Enter Command

Search Database

Enter Search Criteria

Display results

Enter Student Password ID

Initiate file explorer then upload

Display uploaded UI

Display student Form

Enter student Info.

Store student info in database

[Add student]

Grade

[Manage]

Yes

[Search]

Upload file]

***Fig. 3.2 Activity Diagram for the Proposed System***

As seen in figure 3.2, the activity diagram of the proposed system represents the sequence of activities that are involved in the operation of the system. The activities shown are those to be carry out by the user (recruitment manager).

**3.2 PROGRAM SPECIFICATION**

Main program

Begin main program

Enter command

If command = manage

Call manage subprogram

Else if command = search

Call search subprogram

Else if command = match

Call match subprogram

End if,

Manage subprogram

Begin

Enter command

If command = Add student

Call Add student subprogram

Else if command = faculty

Call Faculty Subprogram

End if;

End sub,

Search subprogram

Begin

Enter search criteria

Trim keywords

Search database using keywords

Rank search results

Display search results

End sub,

Match Subprogram

Begin

Enter student ID

Select student profile using ID

Search database using student profile

Grade student and display search results

Display all scores results

End sub

Add assignment subprogram

Begin

Display assignment UI form

Enter assignment details

Store details in database

End sub

Student performance subprogram

Begin

Display Grade UI form

Select student to grade

Enter grade for selected option

Store grade in database

End sub

**3.2.1 FLOW CHART FOR THE PROPOSED SYSTEM**

Flowcharts are employed to help visualize and document the processes in a system and help the viewer understand these processes. Fig. 3.3 shows the diagrammatic representation of the system in terms of the process operations.

Is

grade?

Select student and grade

F

Enter ID

Upload/Search file to/in database

T

F

F

T

Display/download files

Enter search criteria

Search Database

Display Results

Is

search?

is assignment?

Start

Enter Command

Is

manage?

Display Job

is add

test?

Display test form

Enter test info.

Enter Command

Store test in database

F

Display all student test

Store grade for student

T

T

T

Fig. 3.3 Flowchart for the System

**3.3 INTERFACE DESIGN**

This section shows the aspect of the system design that defines the interfaces and data that satisfy specified requirements.

**LOGIN INTERFACE**

**LOGIN WINDOW x**

**User ID**

**PASSWORD**

**LOGIN**

**Fig 3.4 USER INTERFACE**

Fig. 3.4login window for the proposed system

Fig. 3.4 shows the login in interface and the required information for system access, the window is for both the system admin and student.

**ADD ASSIGNMENT INTERFACE**

**User Login**

**Name:**

**file:**

**Lecturer ID:**

**Faculty:**

**Title Of file:**

**SUBMIT**

**Fig. 3.4: Add Assignment Interfaces**

This interface allows student to add assignment to the system

**TEACHER WINDOW**

**User Login**

**NAME**

**Sex:**

**age**

**female**

**Male**

**Qualification**

**Salary**

**Cancel**

**Send Data**

**Figure 3.6: Teacher Interface**

The Teacher interface allows the administrator to register teachers into the system. The actual form contains more detailed information as seen in the application.

SEARCH **WINDOW**

**SEARCH**

**Search field**

**Continue**

**Figure 3.7: Search Interface**

The Search interface application search process in all the window, where admin, lecturer and student can query the database for result in respect to the page view.

The actual form contains more detailed information as seen in the application.

**Figure 3.8: STUDENT PERSONAL DETAILS**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| NAME | SEX | DISTRICT | PARENT | OFFERTYPE | CLASS | STATUS | ACTION |
|  |  |  |  |  |  |  |  |
|  | | | | | | | |

**Figure 3.8: Student personal Interface**

* 1. **DATABASE STRUCTURE**

**USERS**

|  |  |  |
| --- | --- | --- |
| FIELD | DATA TYPE | DESCRIPTION |
| Id | Int(11) | ID auto-increment |
| Username | Varchar(20) | Username |
| Password | Varchar(20) | Password |
| User group | Varchar(25) | User group |
| Status | Varchar(15) | Status |
| Date-registered | Date/time | Date of registration |

**ACADEMIC YEAR**

|  |  |  |
| --- | --- | --- |
| FIELD | DATA TYPE | DESCRIPTION |
| Academic (yr-id) | Int (11) | Academic year id |
| Academic (yr-name) | Varchar (20) | Academic year name |
| Academic (yr-date) | Date/time | Academic date |

**ADMISSION**

|  |  |  |
| --- | --- | --- |
| FIELD | DATA TYPE | DESCRIPTION |
| Admission-ID | Int (11) | Admission ID |
| Institution-ID | Int (11) | Institution ID |
| Faculty-ID | Int (11) | Faculty ID |
| Dept-ID | Int (11) | Department ID |
| Title-ID | Int(11) | Title ID |
| First name | Varchar(20) | First name |
| Surname | Varchar (20) | Surname |
| Nationality | Varchar (50) | Nationality |
| Student no | Varchar (15) | Student number |
| Reg-no | Varchar (20) | Registration number |
| Academic year ID | Int (11) | Academic year ID |
| Course-ID | Int (11) | Course ID |
| Program-ID | Int (11) | Program ID |
| Sponsor-ID | Int (11) | Sponsor |
| Year | Int (11) | Year |
| Sex | Enum (“M”, “F”) | Sex |
| DOB | Date | Date of birth |
| POB | Varchar (40) | Place of birth |
| M-status | Int (11) | Marital status |
| Admission date | Date | Admission date |
| Admission-time | Time | Time of admission |

**ASSIGNMENTTABLE**

|  |  |  |
| --- | --- | --- |
| FIELD | DATA TYPE | DESCRIPTION |
| ID | Int (11) | User ID |
| FILE\_NAME | Varchar (100) | File name |
| File\_tile | Varchar (10o) | File Tile |
| Faculties\_id | Varchar (100) | Faculty id |
| Lecturer\_id | Varchar (100) | Lecturer id |

**FACULTY**

|  |  |  |
| --- | --- | --- |
| FIELD | DATA TYPE | DESC RIPTION |
| Faculty-id | Int(11) | Faculty ID |
| Institution-id | Int (11) | Institution ID |
| Faculty-name | Varchar (200) | Faculty name |
| Faculty-date | Date/time | Faculty date |

**RESULT**

|  |  |  |
| --- | --- | --- |
| FIELD | DATA TYPE | DESCRIPTION |
| Result-id | Int(11) | Result ID |
| Staff-id | Varchar (10) | Staff ID |
| Course unit-id | Int (11) | Course unit ID |
| Course-work | Varchar (5) | Course work |
| Exam | Varchar (5) | Exam |
| Student no | Varchar (10) | Student number |
| Result-date | Date/time | Result date |

**ROLE**

|  |  |  |
| --- | --- | --- |
| FIELD | DATA TYPE | DESCRIPTION |
| Role-id | Int(11) | Role ID |
| Role-name | Varchar(100) | Role name |
| Role-date | Date/time | Role date |

**SEMESTER**

|  |  |  |
| --- | --- | --- |
| FIELD | DATA TYPE | DESCRIPTION |
| Sem-id | Int(11) | Semester ID |
| Sem-name | Varchar (20) | Semester name |
| Sem-date | Date/time | Semester date |

**STAFF**

|  |  |  |
| --- | --- | --- |
| FIELD | DATA TYPE | DESCRIPTION |
| Staff-id | Varchar(10) | Staff ID |
| Staff type-id | Int(11) | Staff type ID |
| Staff-name | Varchar(40) | Staff name |
| Status | Int(11) | Status |
| Staff-date | Date/time | Staff date |

**STAFF TYPE**

|  |  |  |
| --- | --- | --- |
| FIELD | DATA TYPE | DESCRIPTION |
| Staff type-id | Int(11) | Staff type ID |
| Staff type-name | Varchar (20) | Staff type name |
| Staff type-date | Date/time | Staff type date |

**STATUS**

|  |  |  |
| --- | --- | --- |
| FIELD | DATA TYPE | DESCRIPTION |
| ID | Int(11) | ID |
| Name | Varchar (15) | Name |
| Date | Date/time | Date |

**PASS MARK**

|  |  |  |
| --- | --- | --- |
| FIELD | DATA TYPE | DESCRIPTION |
| ID | Int(11) | ID |
| Pass mark | Int(11) | Pass mark |

**PROGRAM**

|  |  |  |
| --- | --- | --- |
| FIELD | DATE TYPE | DESCRIPTION |
| Program-id | Int(11) | Program ID |
| Program-name | Varchar(20) | Program name |
| Program-date | Date/time | Program date |

**REGISTRATION DEAD LINE**

|  |  |  |
| --- | --- | --- |
| FIELD | DATA TYPE | DESCRIPTION |
| ID | Int(11) | ID |
| Date | Date | Date |
| Date 2 | Date | Date 2 |
| Yearost-id | Int(11) | Yearost-id |
| Sem-id | Int(11) | Semester-id |
| Academic-yr | Varchar912) | Academic year |

**REGISTRATION**

|  |  |  |  |
| --- | --- | --- | --- |
| FIELD | DATA TYPE | | DESCRIPTION |
| Takes-id | Int(11) | | Takes ID |
| Academic-yr-id | Int (11) | | Academic year ID |
| Yearost-id | Int (11) | | Yearost ID |
| Sem-id | Int (11) | | Semester ID |
| Reg no | Varchar (20) | | Registration number |
| Student-no | Varchar (15) | | Student number |
| Course-id | Int(11) | | Course ID |
| Course unit-id | Int (11) | | Course unit ID |
| Takes-date | Date | | Takes date |
| Take-time | Time | Time taken | |

**COURSE**

|  |  |  |
| --- | --- | --- |
| FIELD | DATA TYPE | DESCRIPTION |
| Course-id | Int (11) | Course ID |
| Faculty-id | Int (11) | Faculty ID |
| Dept-id | Int (11) | Department ID |
| Course-code | Varchar(20) | Course code |
| Course-name | Varchar(200) | Course name |
| Duration-id | Int (11) | Duration ID |
| Tuition | Int (11) | Tuition |
| YU Course-date | Date/time | Course-date |

**GRADES**

|  |  |  |
| --- | --- | --- |
| FIELD | DATA TYPE | DESCRIPTION |
| Id | Int(11) | Id |
| Upper | Decimal(5,2) | Upper credit |
| Lower | Decimal(5,2) | Lower credit |
| Grade | Varchar(5) | Grades |
| GP | Decimal(5,3) | Grade point |

**ARTICLE ENTRY**

|  |  |  |
| --- | --- | --- |
| FIELD | DATA TYPE | DESCRIPTION |
| Id | Int(11) | Id |
| Student –no | Int(11) | Student number |
| Reg-no | Varchar(12) | Registration number |
| Academic-yr | Int(11) | Academic year |
| Yearost-id | Int(11) | Year of start id |
| Sem-id | Int(11) | Semester id |
| Date sent | Date | Date sent |
| Time sent | Time | Time sent |
| Status | Varchar(20) | Status |

**VIEW STAFF**

|  |  |  |
| --- | --- | --- |
| FIELD | DATA TYPE | DESCRIPTION |
| Staff-id | Varchar (10) | Staff-ID |
| Staff-type | Varchar (20) | Staff-type |
| Staff-name | Varchar (40) | Staff-name |
| Status | Varchar (15) | Status |
| Staff-date | Date/time | Staff date |

**VIEW TEACHERS**

|  |  |  |
| --- | --- | --- |
| FIELD | DATA TYPE | DESCRIPTION |
| ID | Int (11) | ID |
| Subject | Varchar (100) | Subject |
| Staff | Varchar (40) | Staff |
| Date | Date | Date |

**VIEW USERS**

|  |  |  |
| --- | --- | --- |
| FIELD | DATA TYPE | DESCRIPTION |
| Id | Int (11) | ID |
| Username | Varcahr (20) | Username |
| Password | Varchar (20) | Password |
| User group | Varchar (25) | User group |
| Status | Varchar (15) | Status |
| Date-registered | Date/time | Date registered |

**TEACHES**

|  |  |  |
| --- | --- | --- |
| FIELD | DATA TYPE | DESCRIPTION |
| Teaches-id | Int (11) | Teaches ID |
| Course unit-id | Int (11) | Course unit ID |
| Staff-id | Varchar (10) | Staff ID |
| Teaches-date | Date | Teaches date |

**CHAPTER FOUR**

**SYSTEM IMPLEMENTATION**

**4.1.0 INTRODUCTION**

This chapter will focus on the implementation of the system. It discusses the features and choice of the programming language used for the implementation of the system in this research. Also discussed in this chapter is the system testing strategies. The target computer requirements are explained as well as the maintenance issues that should be considered.

**4.2.0 FEATURES AND CHOICE IMPLEMENTATION**

**LANGUAGE**

The programming language used in the implementation of this system is PHP (Hypertext Preprocessor), while the back end database engine adopted was MYSQL.

PHP is a general purpose server-side scripting language originally designed for web development to produce dynamic web pages. It has also evolved to include a command line interface capability and can be used in stand-a lone graphical applications.

The following features make PHP a preferred implementation language for this project:

PHP is an open source language released under the PHP license and includes hundreds of classes and methods such as language support classes for advanced language features like Stings, Arrays, Threads, and Exception Handling.

1. The PHP programming language delivers software-as-a-service (SAAS), and it is run on a web server, making its execution dependent on a web browser. This feature allows for software created with PHP (as that in this research) to be platform independent because it functions independent of the type of operating system running on the target machine.
2. PHP has evolved to include a command-line interface capability and can be used in standalone graphical applications. PHP can be deployed on most web server and also on a standalone shell of almost every operating system and platform free of charge.
3. PHP5 as used in this research includes new features such as improved support for object – oriented programming.

MYSQL is a relational database management system written in C and C++, that runs as a server providing multiuser access to a number of databases. MYSQL is used basically to create a relational database structure on a server in order to store data or automate procedure. The following features make MYSQL suitable for this research.

1. MYSQL is platform independent and it can run under a variety of operating systems.
2. It contains multiple storage engines that allow the programmer to choose what is most effective for each table in the application.
3. MYSQL is considered to be a very fast database management system delivering a multiuser and robust QL database server.

**4.3.0 SYSTEM TESTING**

This section is concerned with testing and debugging of the programs and general processes involved in achieving the objectives of the system requirement. System testing is conducted on a complete integrated system to evaluate the system’s compliance with its specified requirements. So we can also refer to the system testing phase as the investigatory testing phase of the software development life cycle. The system testing strategies used in this system include the unit test and integration test.

**4.3.1 Unit Test**

The primary goal of unit testing is to take the smallest piece of testable software in the application, isolate it from the remainder of the code and determine whether it behaves exactly as it is expected to. Developers who seek to understand the functionality provided by a unit and how to use it can look at the unit test to gain a basic understanding of the unit’s function. In this research, units test were carried out on the various parts of the system and necessary corrections were implemented on the resulting errors.

**4.3.2 Integration Test**

Integration testing is a logical extension of unit testing. In its simplest form, the units that have already been tested and are combined into a component and the interface between them is tested. In a realistic scenario, many units are combined into the components which are in turn aggregated into larger parts of the system. The idea is to test combination of pieces and eventually expand the process to test the modules with those of other groups. The integration test result for the system in this research was satisfactory.

**4.4.0 System Requirements**

This refers to the basic requirements that must be met by the target system to allow for optimal functionality. It puts Q check on the configuration of the target system to allow for compatibility. The target system will be discussed in two areas; Hardware and software requirements.

**4.4.1 Software Requirements**

The table shows a list of the software requirements for the proposed system to function.

|  |  |
| --- | --- |
| **Component** | **Requirement** |
| Operating system  PHP version  Database Engine  Web Server  Browsers | Windows 2000, XP, VISTA, 7  PHP 4.2.1 or later  MYSQL ‘5  Apache web server  Firefox, I.E, Opera |

***Table 4.1: Software requirements for the target system***

**4.4.2 Hardware Requirements**

Table 4.2 shows a list of the hardware requirements for the proposed system to function.

|  |  |
| --- | --- |
| **Component** | **Requirement** |
| RAM  Harddisk  Processor | 250MB or higher  10GB or higher  33H~~z~~ or higher |

Table 4.2: Hardware requirements for the target system.

**4.5.0 USER MANUAL**

The following is a brief guide for users of the application:

1. A user should start the server application by double clicking on the icon provided on the windows desktop. The server application will be started.
2. To start the client application, the user should use the Uniform Resource Locator (URL) provided for the system. The user page would be displayed.
3. When this page is displayed, the user can perform any of the following operations:
4. View all jobs to manage
5. Search using criteria
6. Perform an optional quick match by specifying a “job id”
7. Manage a selected job by clicking on the “manage” link on the user page.
8. Add a test to a job by clicking on the “add test” link on the job view page.
9. Grade an application by clicking on the “Grade” link provided on the job view page.
10. View all applications ordered by the applicants grade in descending order by clicking on the “view applications by grade” link provided on the “job view” page.
11. View applicant information by clicking on the “view application” link provided on the job view page.
12. View on the applicants resume by clicking on the “resume” page provided on the applicants page.

**CHAPTER FIVE**

**5.1.0 INTRODUCTION**

Base on the new system that is developed, several points has been carefully observed for further research in other to further improve our school system activities.

In summary of this project the researcher has read through school management system, which is quite large to be encapsulated in just one project work, so this project is reduced to focus on the online assignment submission management system as a whole.

The online assignment submission management system has been sub-divided into modules in other to carefully handle the processes involved in this system. The online assignment submission management system is sub-divided into (4) four major modules which are:

1. Description of the system
2. Program Specification
3. Interface Design
4. Structure of the Database

This module is also broken down, so as to capture all the necessary segments that are required to make this project successful.

**5.2.0 CONCLUSION**

In deciding what a school management system should look like for a school before abstracting to the research domain for any faculty or department. They should be validated in the fields. It’s important to look closely at systems that have successfully been deployed in challenging environment and any available evaluation data.

In creating an online assignment submission management system, out of school management system, it is essential that the assignment submission management system is designed with goal of allowing register student to login in with their valid ID access to access the assignment page and being able to upload his / her assignment in the system.

We are now in the fortunate situation of falling information technology costs, improving computer literacy combined with preliminary evidence of school management successes in resource poor areas. The critical challenge is to create well- designed, effective, low cost systems by sharing resources, learning from each other experiences and evaluating our work.

**5.3.0 RECOMMENDATION**

The software developed for the implementation of this research can be used by any School. The following recommendations are proposed:

* System backup should be carry out day-to-day if place on a central server on the school building.
* A cloud system backup is highly recommended to ensure safety of system in terms of theft or fire outbreak.
* Every staff granted access to the system is require to sign an undertaken to protect the integrity of the system.
* Users of the system should ensure to print out payment transactions for future references.
* Administrator of the system should create passwords with long characters so as to make password hacking difficult.
* User with certain privilege should ensure safekeeping of password since it provides access to the system.

**REFERENCES**

Ben-Ari, M. (2001). Constructivism in computer science education. *Journal of Computers in Mathematics &Science Teaching, 20*(1), 24-73.

Commonwealth of Australia. (2000, Thursday, 29th November 2001). Overseas student statistics. Retrievedfrom http://www.dest.gov.au/highered/statistics/tables/students2000.htm

CourseMaster. (2000). School of Computer Science & IT, The University of Nottingham, UK. Retrieved fromhttp://www.cs.nott.ac.uk/CourseMaster/cm\_com/index.html

Daconta, M. (Ed.), Monk, E. & Saganich, A. (1998). Java 1.2 and Javascript for C and C++ Programmers. JohnWiley & Sons.

Foxley, E. (1999). Ceilidh on the World Wide Web. Retrieved from http://www.cs.nott.ac.uk/~ceilidh/

Hot Potatoes (2003). Retrieved from http://web.uvic.ca/hrd/halfbaked/

Joy, M. & Luck, M. (1995). On-line submission and testing of programming assignments. In J. Hart (Ed.)*Innovations in computing teaching*. London: SEDA.

Joy, M. & Luck, M. (1998). Effective electronic marking for on-line assessment. In *Proceedings of the 6thAnnual Conference on the Teaching of Computing/3rd Annual Conference on Integrating Technology intoComputer Science Education*, 134-138.

Nulty, D., Bancroft, P., Brewster, S., & Smith, D. (1998). Web online feedback system. Retrieved fromhttp://www.fit.qut.edu.au/wolf/

QUT (2000) http://www.talss.qut.edu.au/talss/stafftlgrants/gen/

QUT. (2003-2007). *Teaching and learning plan*. QUT. Retrieved from

http://www.qut.edu.au/chan/pr/planning/StratPlan/stratplan.htm.

Thomas, P. G., & Paine, C. B. (2000). *Tools for observing study behaviour*. Paper presented at the Psychology of Programming Interest Group, April, Cosenza, Italy.

A. Tregobov, “*The Web-Based Assignment Submission Systems*”,

presented at NAWeb’98, University of New Brunswick, Canada, 1998.

David Jones, Sandy Behrens, “*Online Assignment Management: An*

*Evolutionary Tale,*” Hawaii International Conference on System

Sciences, vol. 5, no. 5, pp. 156c, Track 5, 2003.

APPENDIX A: SOURCE CODE

**Index.php**

<?php

session\_start();

require("conection/connects.php");

$msg="";

if(isset($\_POST['btn\_log'])){

$uname=$\_POST['unametxt'];

$pwd=$\_POST['pwdtxt'];

$sql=mysqli\_query($db,"SELECT \* FROM users\_tbl

WHERE username='$uname' AND password='$pwd'

");

$cout=mysqli\_num\_rows($sql);

if($cout>0){

$row=mysqli\_fetch\_array($sql);

$\_SESSION['uname'] =$row['username'];

$\_SESSION['pass'] =$row['password'];

if($row['type']=='admin')

$msg="UNILAG School Admin!.....";

else

header("location: everyone.php");

}

else

$msg="Login Username and Password Wrong......";

}

?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

<head>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

<title>::. Springlight School Manager.::</title>

<link rel="stylesheet" type="text/css" href="css/login.css" />

</head>

<body>

<form method="post">

<fieldset>

<fieldset></fieldset>

<div id="login\_back">

<div id="msg">

</div>

<div id="login\_form">

<label for="login">Username:</label>

<input type="text" class="fields" name="unametxt" title="Enter username here" />

<div class="clear"></div>

<label for="login">Password:</label>

<input type="password" class="fields" name="pwdtxt" title="Enter Password here" autocomplete="off"/>

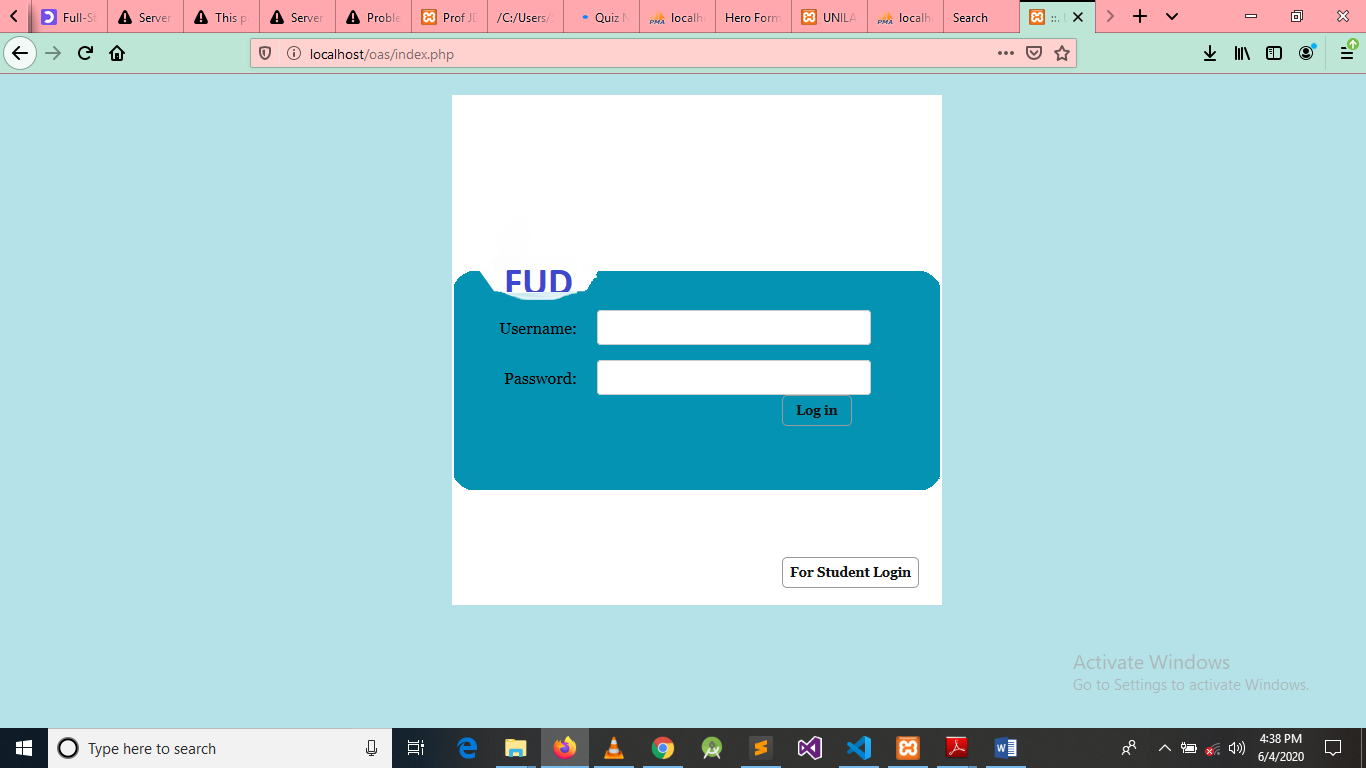
<div class="clear"></div>

<input type="submit" class="button" name="btn\_log" value="Log in" />

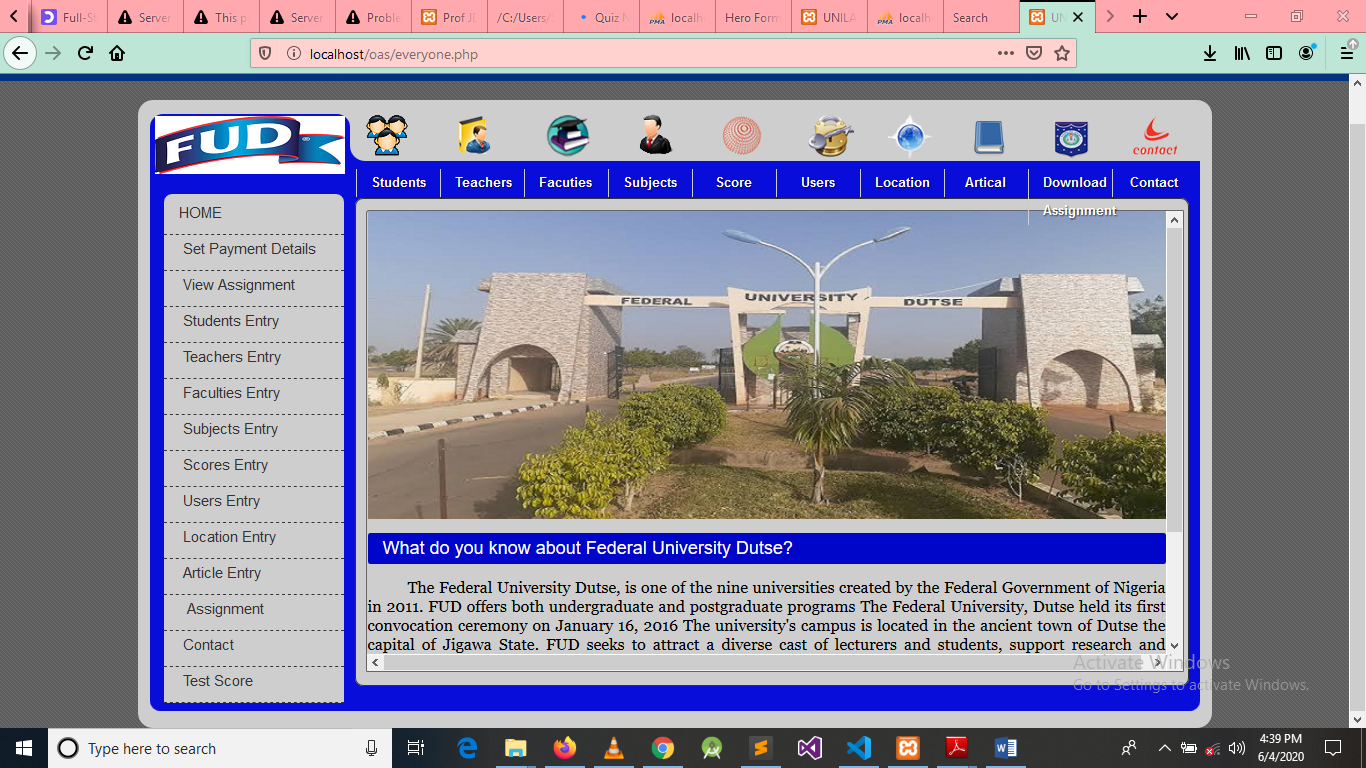
</div

**APPENDIX B: WINDOW SCREEN.**

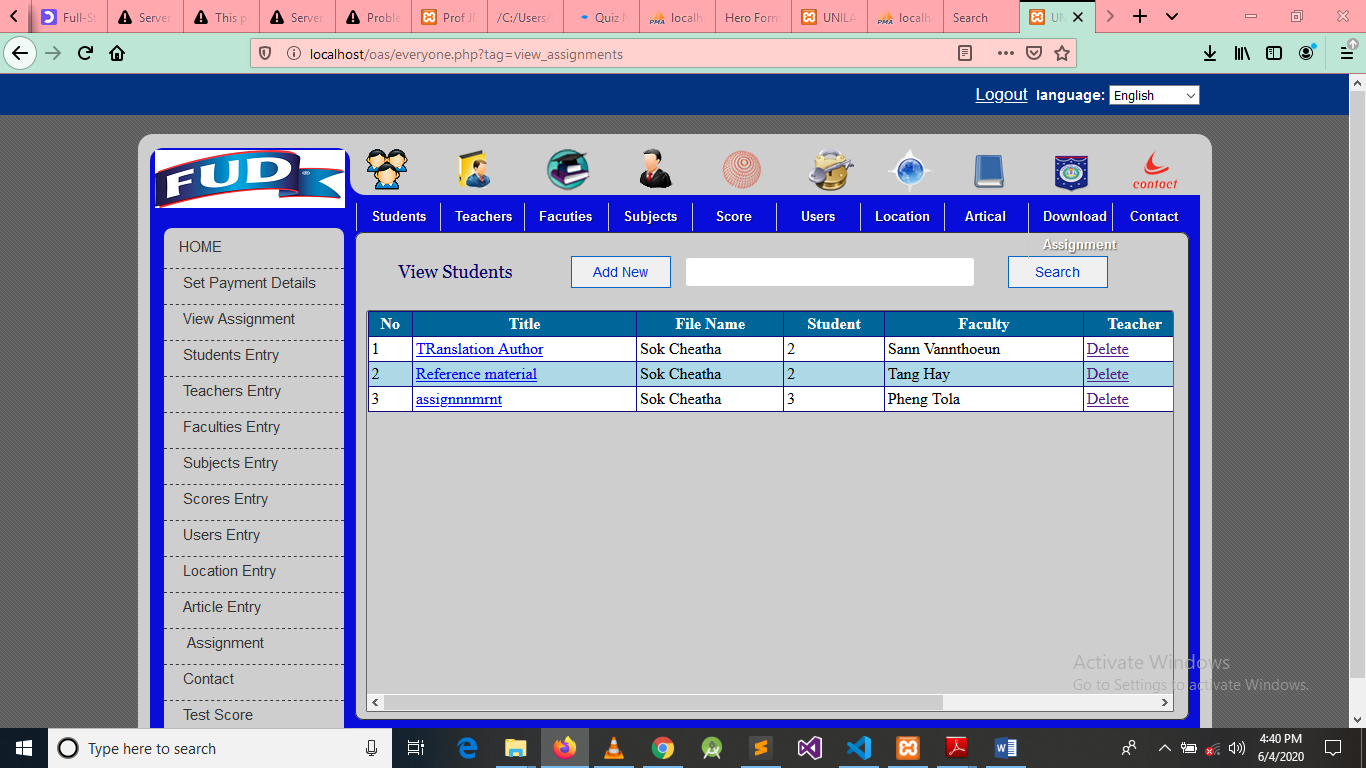
Login screen(index.php)



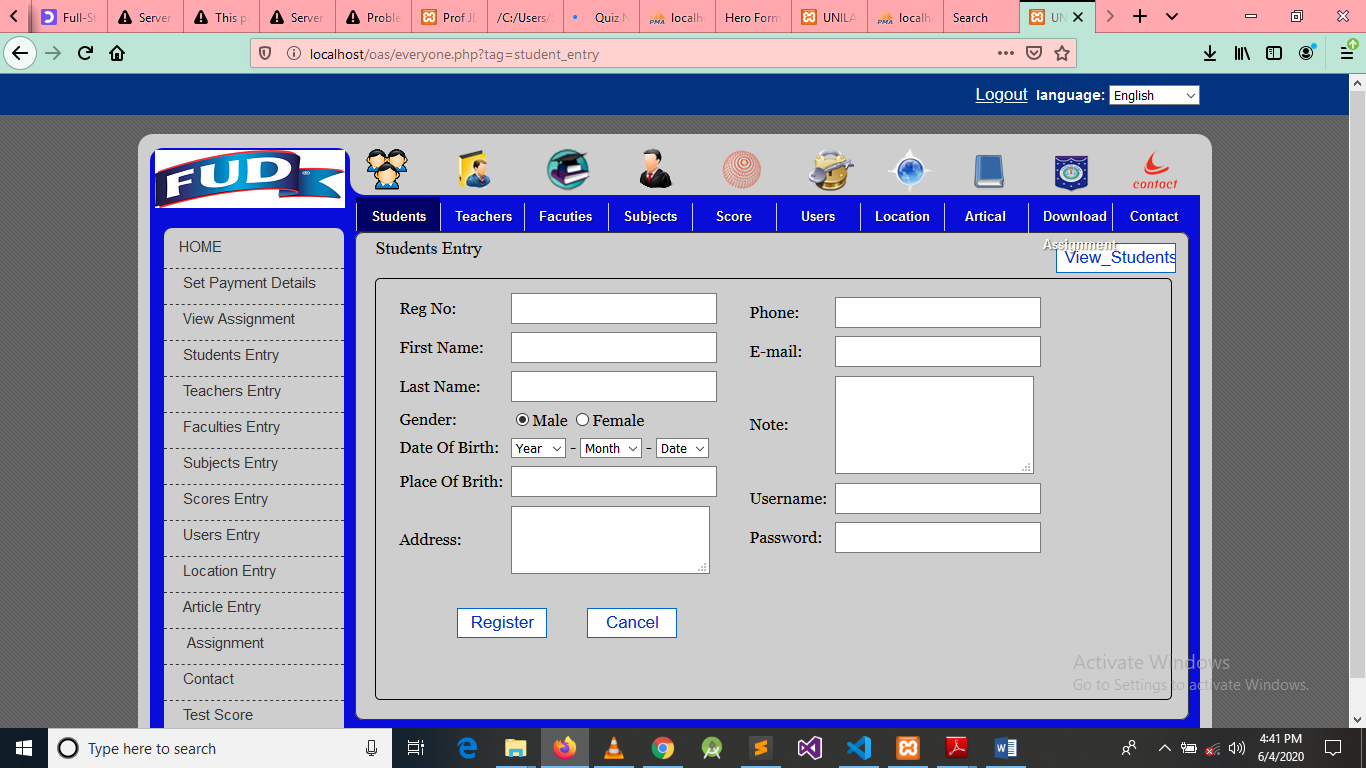
Welcome screen (everyone.php included home.php)



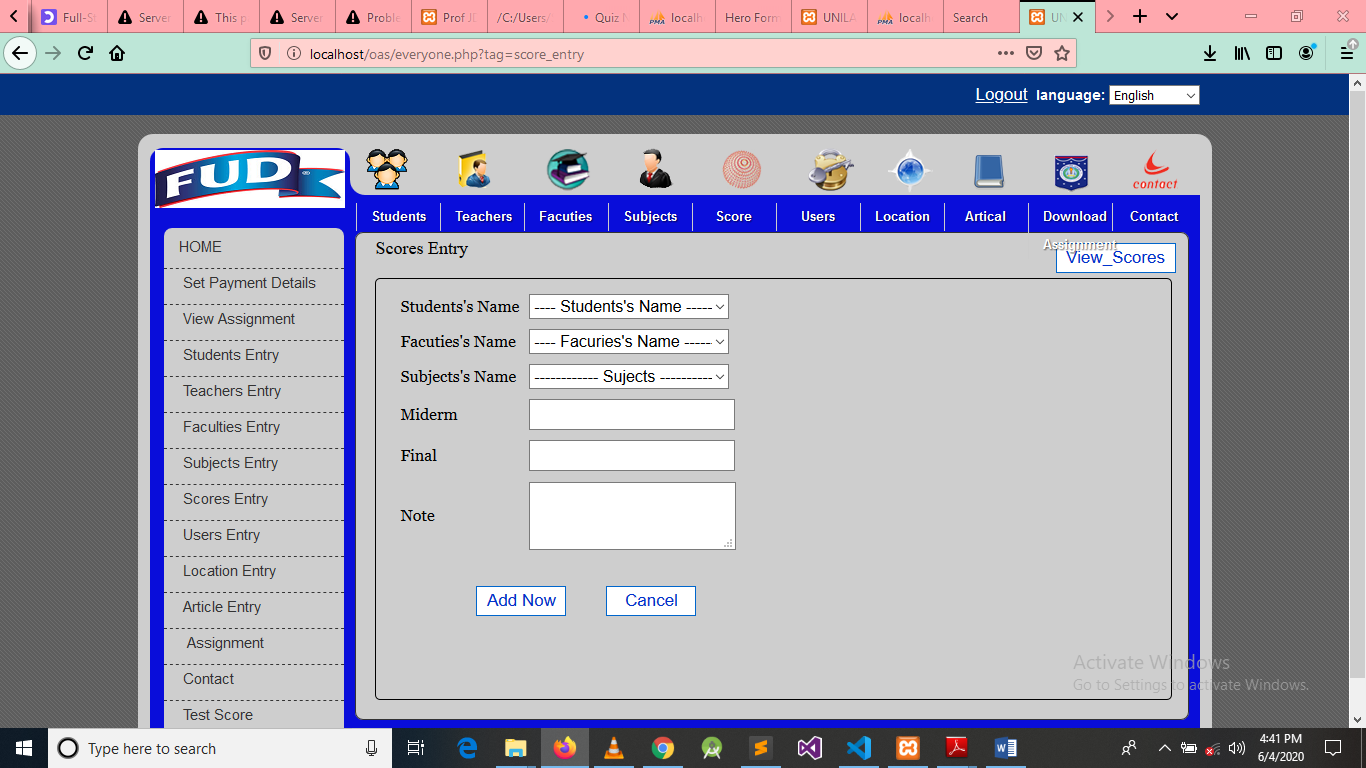
VIEW ASSIGNMENT



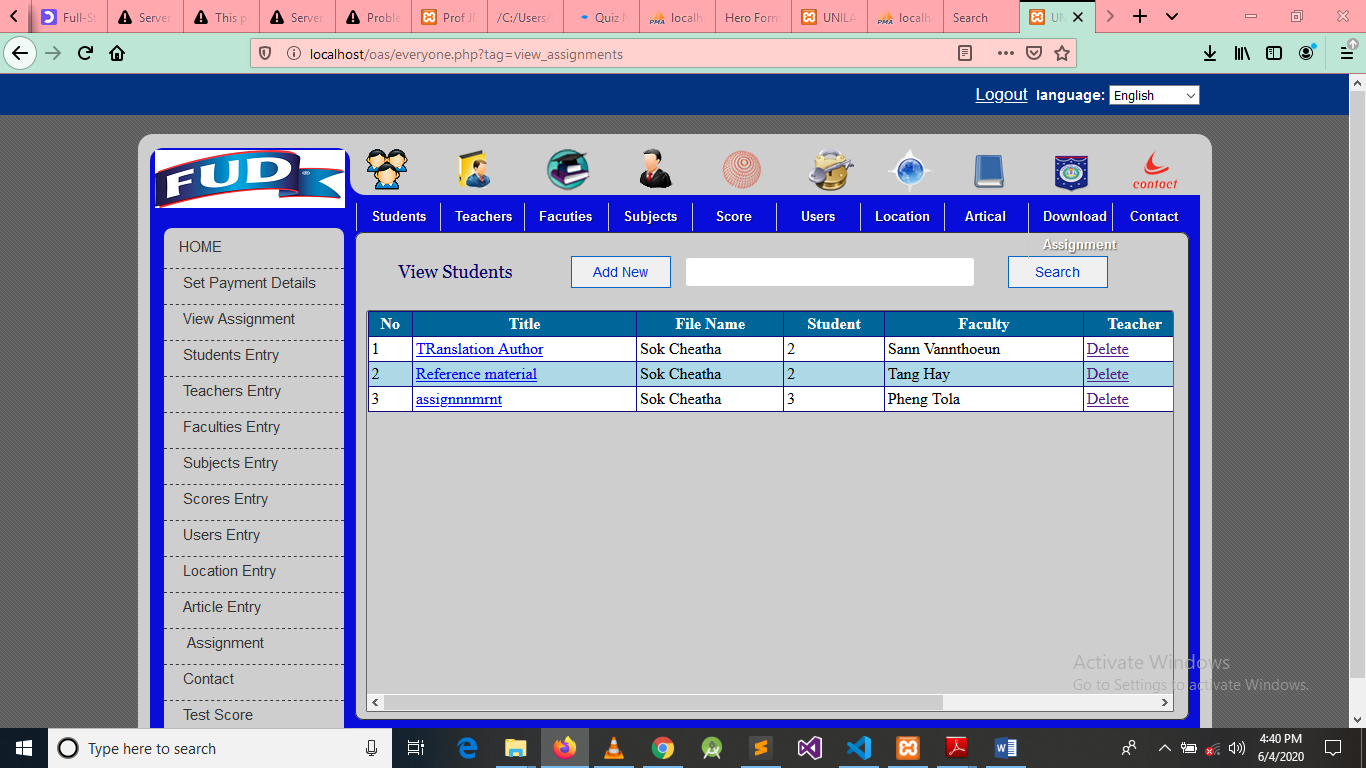
STUDENT ENTRY



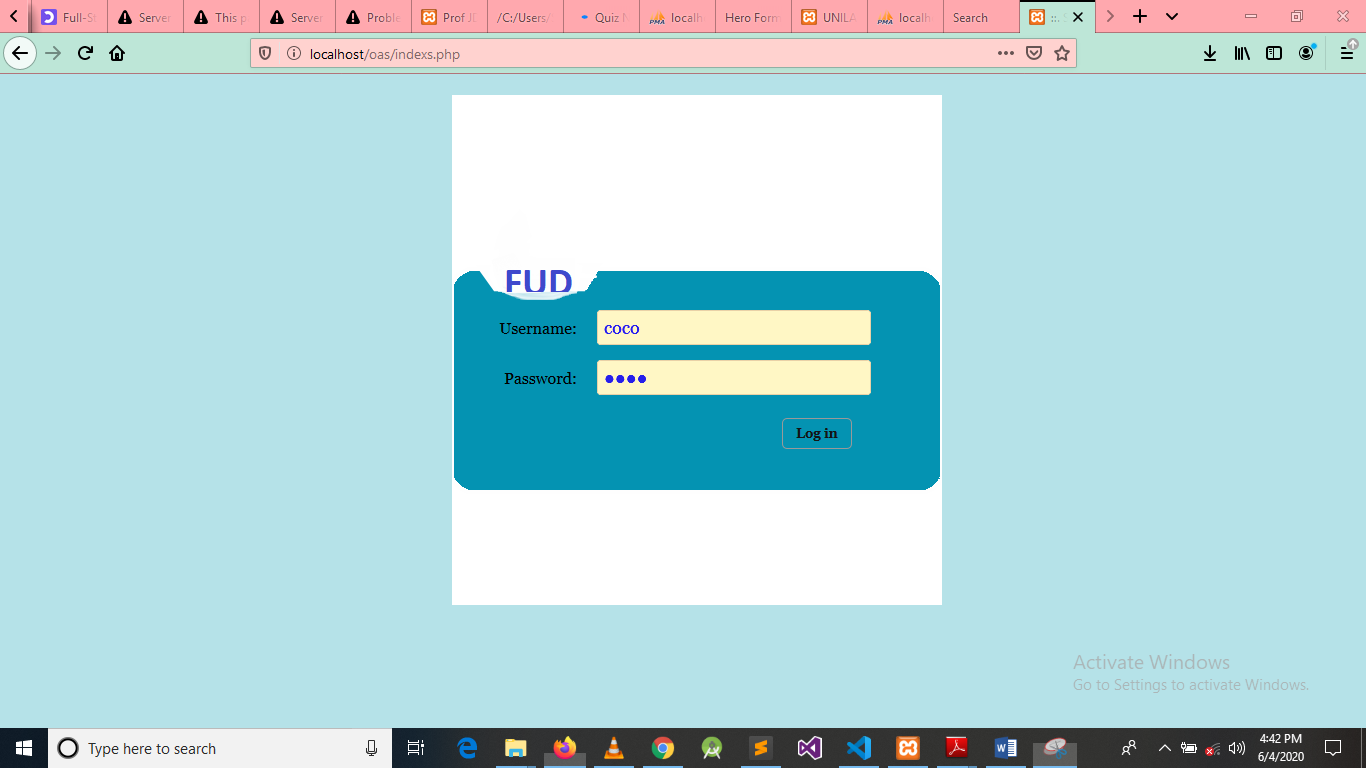
SCORE\_ENTRY



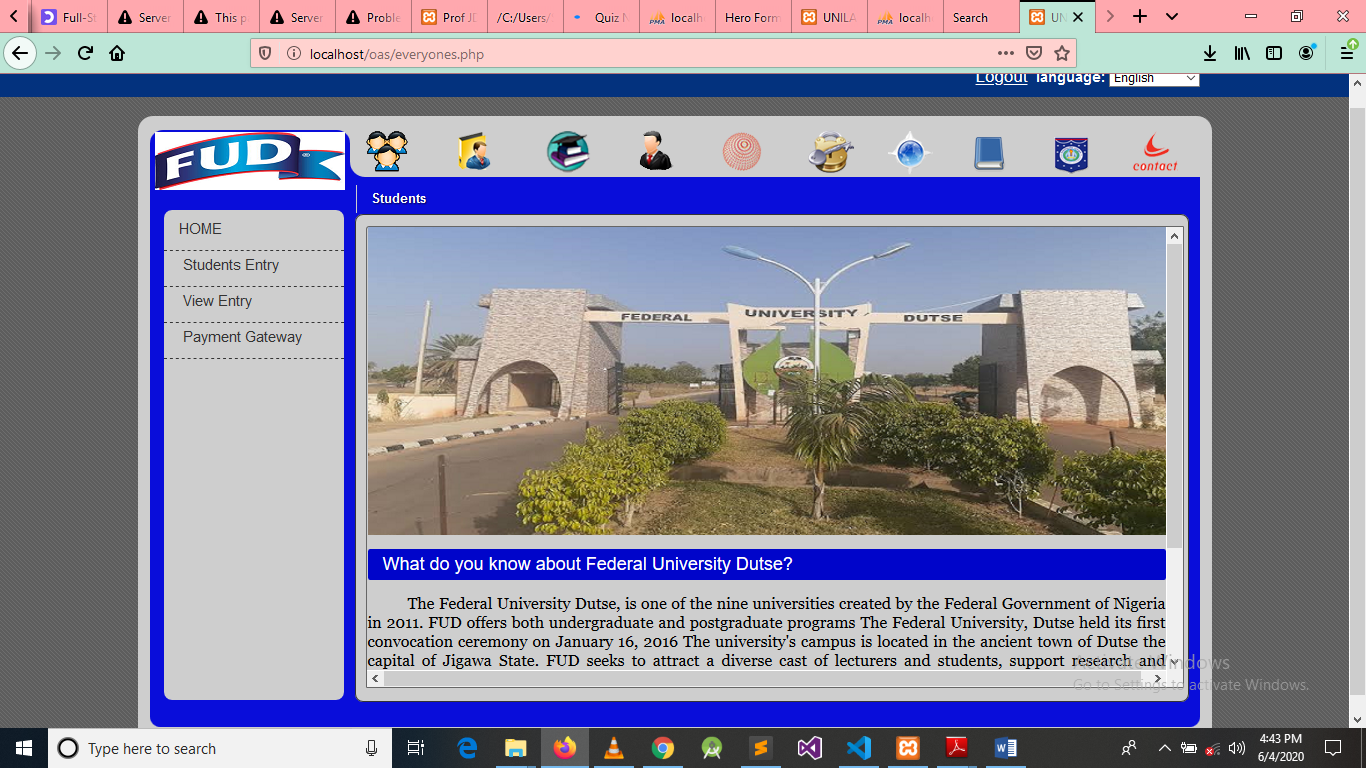
VIEW\_ASSIGNMENT



STUDENT LOGIN



STUDENT HOME SCREEN



FILE ENTRY

