GELAN BOY'S BOARDING SECONDARY SCHOOL

CTE ANNUAL ASSIGNMENT

Topic: Connect PHP with MySQL Database
Apply CRUD Operations

Grade 12 – CTE New Curriculum

Prepared by: Mira Tamiru

UNIT 4: INTEGRATE DATABASE

Unit Coverage

This unit provides the necessary information and practice regarding the following content coverage:

1. Connect to a MySQL database
2. Apply CRUD queries Operations

4.1. Connect PHP with MySQL Database

To connect to a MySQL database in PHP, you can use `mysqli_connect()` (procedural style) or create a `mysqli` object (object-oriented style). To integrate a MySQL database with PHP and perform CRUD (Create, Read, Update, and Delete) operations, you'll need to use PHP's MySQLi or PDO extension. MySQLi (MySQL Improved) is simpler and provides a good way to interact with MySQL databases.

Procedural Style Connection

```
<?php
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "school";

// Create connection
$conn = mysqli_connect($servername, $username, $password, $dbname);

// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}
```

```
echo "Connected successfully";
mysqli_close($conn);
?>
```

Object-Oriented Style Connection

```
<?php
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "school";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

echo "Connected successfully";
$conn->close();
?>
```

Databases, PHP, and Web Design

When working on a website that is either hosted or local on your computer, sometimes you'll need a database for it. Historically, a deep knowledge of SQL was required to run commands in a command prompt. Nowadays, tools like **phpMyAdmin** provide a graphical user interface for SQL. This allows users to create databases, tables, and manage permissions through a visual interface.

4.2. Apply CRUD Queries Operations

Creating PHP CRUD using Object-Oriented Programming (OOP) increases coding confidence for developing full-fledged web application projects. CRUD (Create, Read, Update, and Delete) represents the fundamental building

blocks of software. Using the MVC architecture for these implementations ensures better maintainability.

Every Entity in an application requires CRUD implementation. For instance, in project management software, the Project, Task, Owner, and Client are distinct entities that require separate CRUD functionalites for data manipulation.

Database Structure (crud_example.sql)

```
CREATE DATABASE IF NOT EXISTS school;
USE school;

CREATE TABLE IF NOT EXISTS tbl_student (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    roll_number VARCHAR(20) NOT NULL UNIQUE,
    dob DATE NOT NULL,
    class VARCHAR(20) NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

CREATE TABLE IF NOT EXISTS tbl_attendance (
    id INT AUTO_INCREMENT PRIMARY KEY,
    attendance_date DATE NOT NULL,
    student_id INT NOT NULL,
    present TINYINT DEFAULT 0,
    absent TINYINT DEFAULT 0,
    FOREIGN KEY (student_id) REFERENCES tbl_student(id) ON DELETE
CASCADE,
    UNIQUE KEY unique_attendance (attendance_date, student_id)
);
```

Database Controller (DBController.php)

The DBController class handles the core database connection and query execution logic.

```
<?php
class DBController {
```

```
    private $host = "localhost";
    private $user = "root";
    private $password = "";
    private $database = "school";
    private $conn;

    function __construct() {
        $this->conn = $this->connectDB();
    }

    function connectDB() {
        $conn = mysqli_connect($this->host, $this->user, $this-
>password, $this->database);
        if (!$conn) {
            die("Connection failed: " . mysqli_connect_error());
        }
        return $conn;
    }

    function runBaseQuery($query) {
        $result = mysqli_query($this->conn, $query);
        $resultset = array();
        if ($result) {
            while ($row = mysqli_fetch_assoc($result)) {
                $resultset[] = $row;
            }
        }
        return $resultset;
    }

    function runQuery($query, $param_type, $param_value_array) {
        $sql = $this->conn->prepare($query);
        if (!$sql) { return array(); }
        $this->bindQueryParams($sql, $param_type,
$param_value_array);
        $sql->execute();
        $result = $sql->get_result();
        $resultset = array();
        if ($result) {
            while ($row = $result->fetch_assoc()) {
                $resultset[] = $row;
            }
        }
```

```
        return $resultset;
    }

    function insert($query, $param_type, $param_value_array) {
        $sql = $this->conn->prepare($query);
        if (!$sql) { return 0; }
        $this->bindQueryParams($sql, $param_type,
$param_value_array);
        $sql->execute();
        return $this->conn->insert_id;
    }

    function update($query, $param_type, $param_value_array) {
        $sql = $this->conn->prepare($query);
        if (!$sql) { return 0; }
        $this->bindQueryParams($sql, $param_type,
$param_value_array);
        $sql->execute();
        return $sql->affected_rows;
    }

    function bindQueryParams($sql, $param_type, $param_value_array) {
        $param_value_reference[] = &$param_type;
        for ($i = 0; $i < count($param_value_array); $i++) {
            $param_value_reference[] = &$param_value_array[$i];
        }
        call_user_func_array(array($sql, 'bind_param'),
$param_value_reference);
    }
}
?>
```

Student Management Logic (Student.php)

```
<?php
require_once("class/DBController.php");

class Student {
    private $db_handle;
    function __construct() { $this->db_handle = new DBController(); }

    function addStudent($name, $roll_number, $dob, $class) {
```

```
        $query = "INSERT INTO tbl_student (name, roll_number, dob,
class) VALUES (?, ?, ?, ?)";
        $paramType = "ssss";
        $paramValue = array($name, $roll_number, $dob, $class);
        return $this->db_handle->insert($query, $paramType,
$paramValue);
    }

    function editStudent($name, $roll_number, $dob, $class,
$student_id) {
        $query = "UPDATE tbl_student SET name = ?, roll_number = ?,
dob = ?, class = ? WHERE id = ?";
        $paramType = "ssssi";
        $paramValue = array($name, $roll_number, $dob, $class,
$student_id);
        $this->db_handle->update($query, $paramType, $paramValue);
    }

    function deleteStudent($student_id) {
        $query = "DELETE FROM tbl_student WHERE id = ?";
        $paramType = "i";
        $paramValue = array($student_id);
        $this->db_handle->update($query, $paramType, $paramValue);
    }

    function getAllStudent() {
        $sql = "SELECT * FROM tbl_student ORDER BY name";
        return $this->db_handle->runBaseQuery($sql);
    }
}
?>
```

Attendance Logic (Attendance.php)

```
<?php
require_once("class/DBController.php");

class Attendance {
    private $db_handle;
    function __construct() { $this->db_handle = new DBController(); }

    function addAttendance($attendance_date, $student_id, $present,
```

```
$absent) {
        $query = "INSERT INTO tbl_attendance (attendance_date,
student_id, present, absent) VALUES (?, ?, ?, ?)";
        $paramType = "siii";
        $paramValue = array($attendance_date, $student_id, $present,
$absent);
        return $this->db_handle->insert($query, $paramType,
$paramValue);
    }

    function getAttendance() {
        $sql = "SELECT attendance_date, COUNT(*) as total_students,
SUM(present) as total_present, SUM(absent) as total_absent
                FROM tbl_attendance GROUP BY attendance_date ORDER BY
attendance_date DESC";
        return $this->db_handle->runBaseQuery($sql);
    }
}
?>
```

System Navigation and File Structure

The application is organized into a clean folder structure to separate logic (classes), presentation (web), and assets (css).

```
project-folder/
├── class/
│   ├── Attendance.php
│   ├── DBController.php
│   └── Student.php
├── web/
│   ├── css/style.css
│   ├── attendance-add.php
│   ├── attendance.php
│   ├── header.php
│   ├── footer.php
│   ├── student-add.php
│   └── student.php
├── crud_example.sql
└── index.php
```