Green University of Bangladesh

Department of Computer Science and Engineering (CSE)

Semester: (Spring, Year: 2025), B.Sc. in CSE (Day)

Lab Report 04 - PET Management System

Course Title: Integrated Design Project II

Course Code : CSE-406

Section : 213 D7

Students Details

Name	ID
MD Dulal Hossain	213902116
MD Rabby Khan	213902037
Mostak Ahammed	213902126

Submission Date: 16 April 2025

Course Teacher's Name: Sharifur Rahman

Designation: Lecturer

Department of CSE , GUB .

[For teachers use only: Don't write anything inside this box]

Lab Project Status	
Marks:	**Signature:**
Comments:	**Date:**

Contents

Chapter 1

1.1 Title Of The Lab Report Experiment

- Create database using PHP .
- Perform various operation(Created, Read, Update, Delete) in server site using PHP.

1.2 Objectives / Aim

- To understand the implementation of CRUD operations (Create, Read, Update, and Delete) using PHP and MySQL.
- To develop a dynamic web application that can interact with a database on the server.
- To enhance the understanding of server-side scripting with PHP.
- To learn how to handle form data and perform database queries securely.

1.3 Procedure

- **Backend Language:** PHP
- **Database:** MySQL
- **Front-end:** HTML (with optional CSS for styling)
- **Operations Implemented:**
  - **Create:** Add a new record into the database.
  - **Read:** Display records from the database.
  - **Update:** Modify existing records.
  - **Delete:** Remove records from the database.

1.4 Implementation

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

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

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

Figure 1.1: Connection Database Source Code.

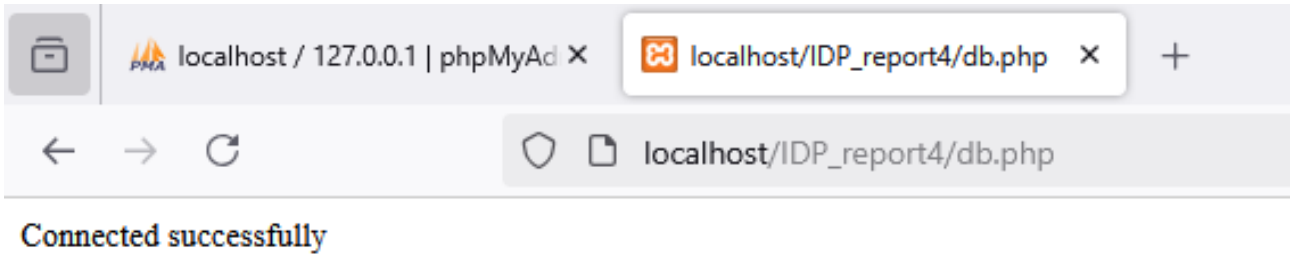

Figure 1.2: Connection Successful Message.

createtable.php > ...

```php
<?php

include("db.php");

$sql="CREATE TABLE student  (
    name varchar(20) NOT NULL,
    age varchar(20) NOT NULL)";

if($conn-> query(query: $sql)==true){
    echo "succesfull";

}
else{
    echo "unsuccessfull";
}

?>
```

Figure 1.3: Create Table Source Code.

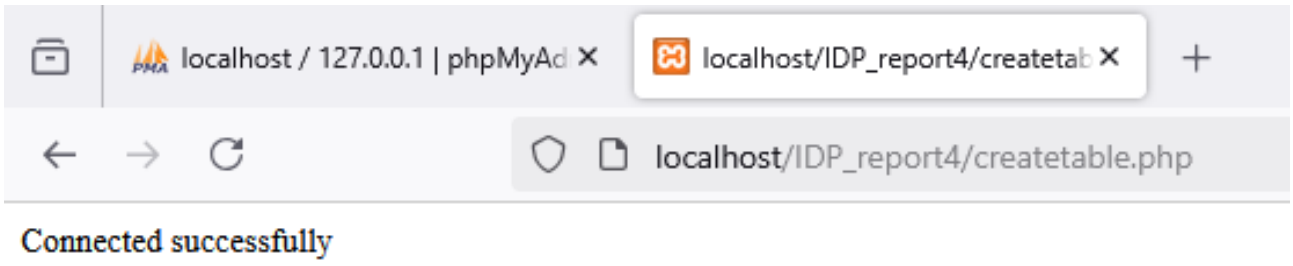

Figure 1.4: Create Table Successful Message.

Create (insert.php): The newly added data appears when redirected to the listing (Read) page.

```php
<?php
include("db.php");

$name = $_POST['name'] ?? '';
$department = $_POST['department'] ?? '';

if (!empty($name) && !empty($department)) {
    $stmt = $conn->prepare(query: "INSERT INTO students (name, department) VALUES (?, ?)");
    $stmt->bind_param(types: "ss", var: &$name, vars: &$department);

    if ($stmt->execute()) {
        header(header: 'Location: index.php');
        exit;
    } else {
        echo "Error: " . $stmt->error;
    }

    $stmt->close();
} else {
    echo "Please provide both name and department.";
}

$conn->close();
?>
```

Figure 1.5: Read (index.php): Displays a dynamic table containing all records from the users table.

```php
<?php
// Database connection
$conn = new mysqli(hostname: 'localhost', username: 'root', password: '', database: 'school');
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

// Insert data if form is submitted
if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    $name = $_POST['name'];
    $age = $_POST['age'];

    $insert_sql = "INSERT INTO students (name, age) VALUES ('$name', $age)";
    if ($conn->query(query: $insert_sql) === TRUE) {
        echo "New record created successfully";
    } else {
        echo "Error: " . $insert_sql . "<br>" . $conn->error;
    }
}

// Retrieve data
$sql = "SELECT * FROM students";
$result = $conn->query(query: $sql);
?>
```

Figure 1.6: Update (edit.php and update.php): When clicking the update link, the user is taken to a form with pre-filled data of the selected record.

update.php > ...

```php
<?php
include("db.php");

if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    // Handle form submission
    $id = $_POST['id'] ?? null;
    $name = $_POST['name'] ?? '';
    $age = $_POST['age'] ?? '';

    if ($id !== null && is_numeric(value: $id) && $name !== '' && $age !== '') {
        $stmt = $conn->prepare(query: "UPDATE students SET name = ?, age = ? WHERE id = ?");
        $stmt->bind_param(types: "sii", var: &$name, vars: &$age, $id);

        if ($stmt->execute()) {
            $stmt->close();
            $conn->close();
            header(header: 'Location: index.php');
            exit;
        } else {
            echo " Error updating record: " . $stmt->error;
        }

        $stmt->close();
    } else {
        echo " Invalid input data.";
    }
```

Figure 1.7: Delete (delete.php):

- Clicking the Delete link sends the user ID to the server.
- The corresponding row is removed from the users table.

delete.php > ...

```php
<?php
include("db.php");

$id = $_GET['id'] ?? null;

if ($id !== null && is_numeric(value: $id)) {
    $stmt = $conn->prepare(query: "DELETE FROM students WHERE id = ?");
    $stmt->bind_param(types: "i", var: &$id);

    if ($stmt->execute()) {
        header(header: 'Location: index.php');
        exit;
    } else {
        echo "Error deleting record: " . $stmt->error;
    }

    $stmt->close();
} else {
    echo "Invalid or missing ID.";
}

$conn->close();
?>
```

Figure 1.8: delete.php

Overall, the output clearly demonstrates how server-side scripting using PHP can be used to manage database records efficiently through a web interface. All operations are performed in real-time and reflect immediately in the displayed data.

1.5 Test Result / Output

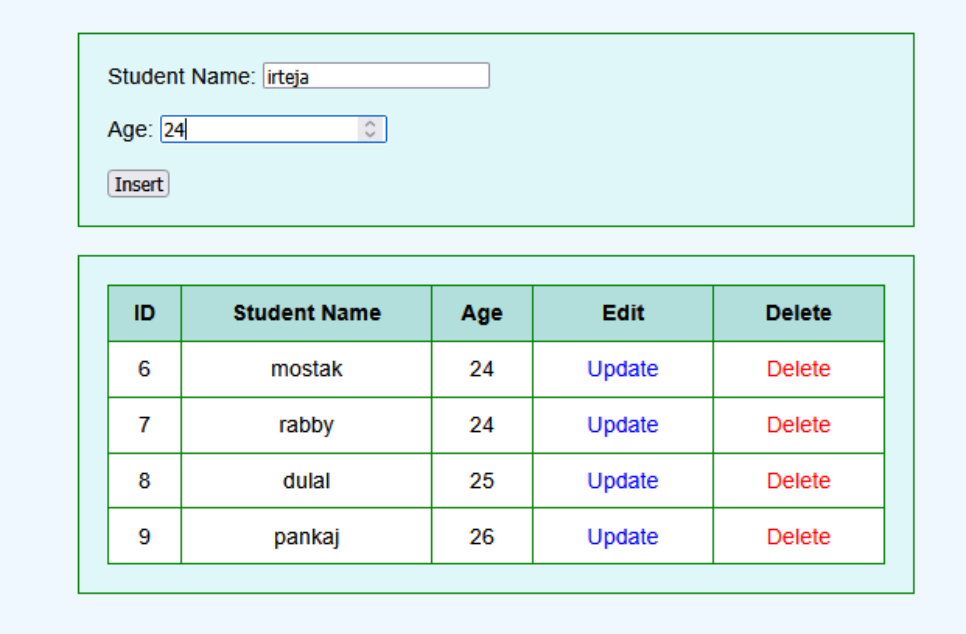

Figure 1.9: Add Student Information.

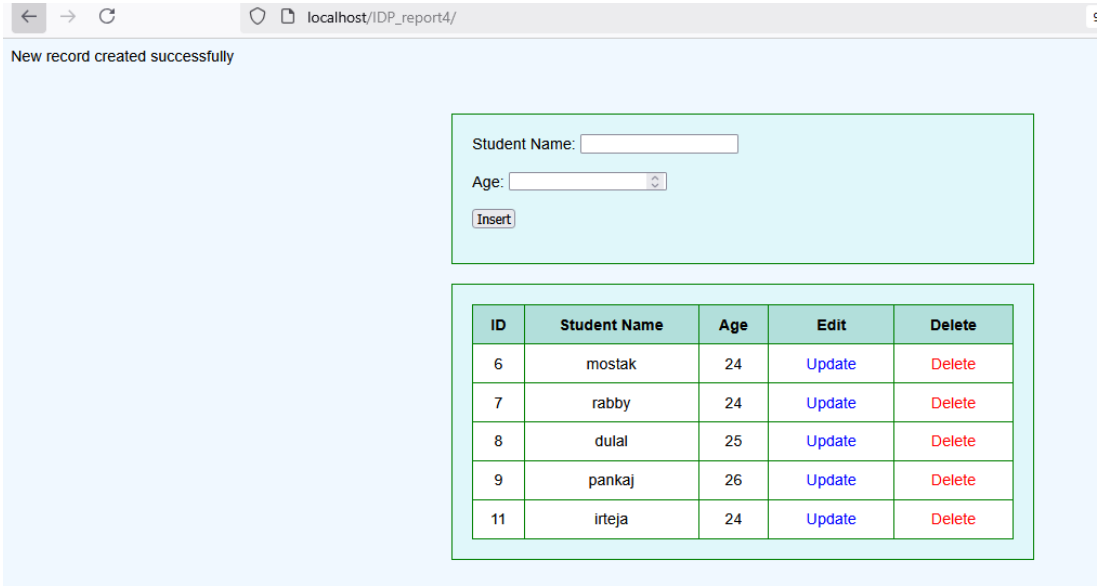

Figure 1.10: Show student information.

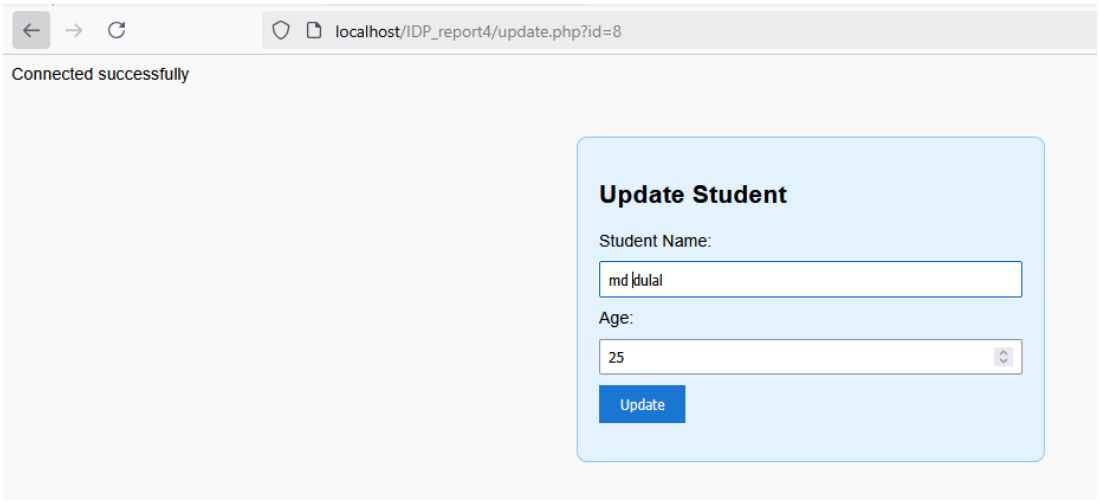

Figure 1.11: Update Student Information.

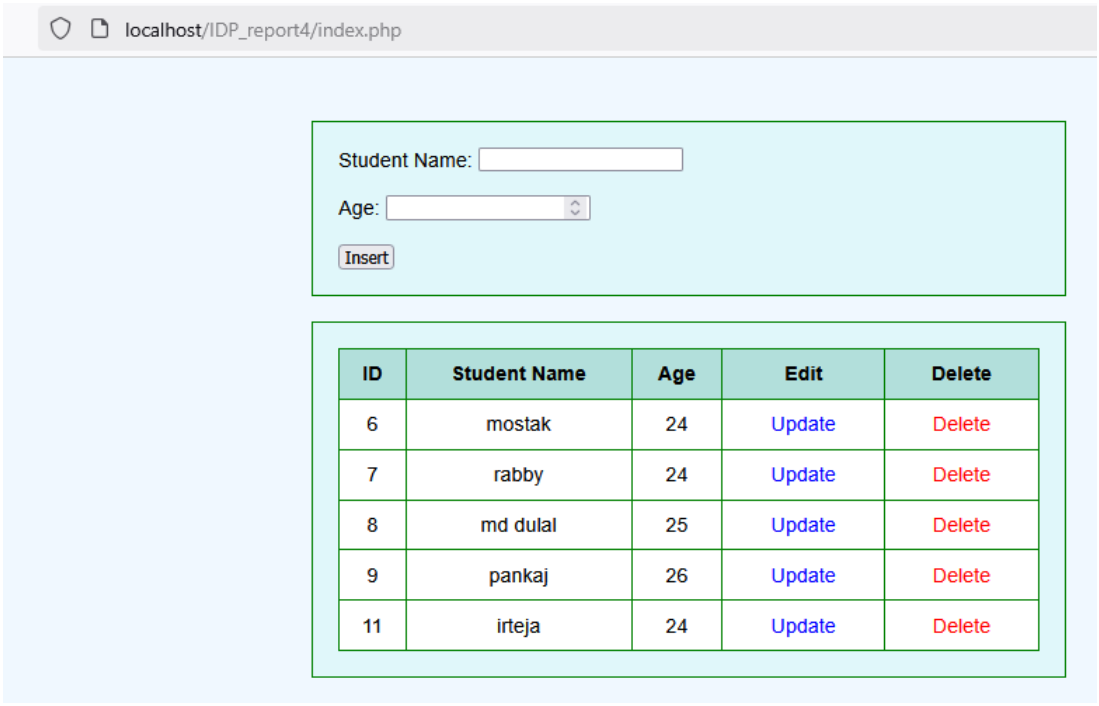

Figure 1.12: After Update show student information.

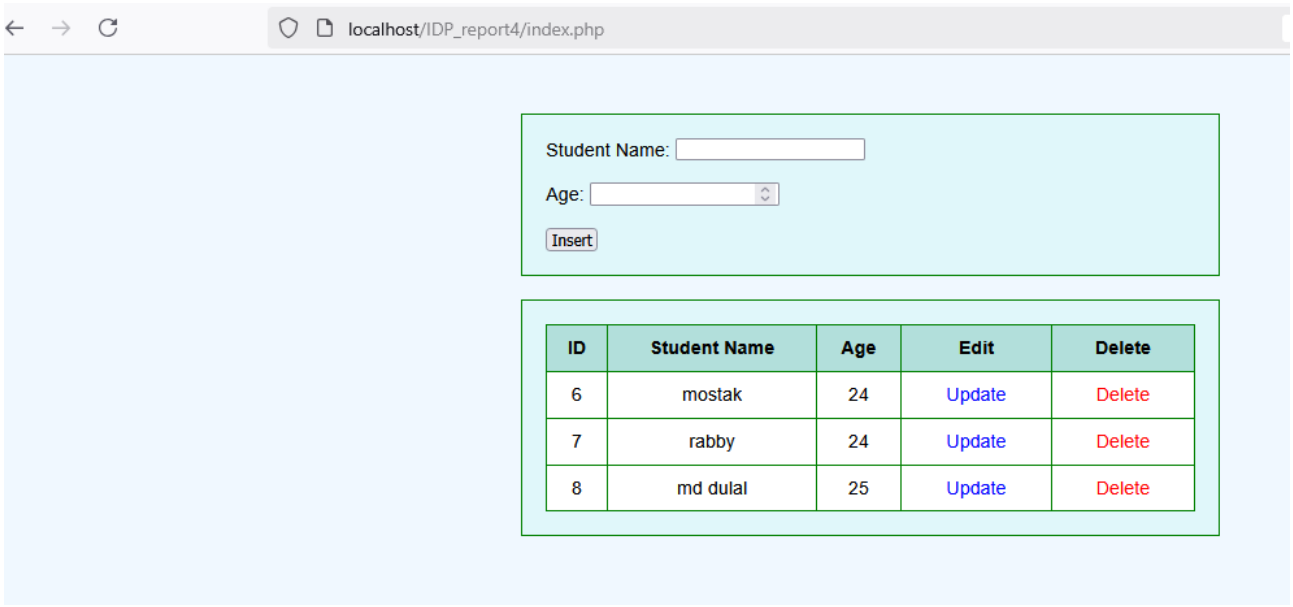

Figure 1.13: After Delete show student information.

1.6 Analysis and Discussion

The lab demonstrated the practical application of CRUD operations in PHP using MySQL as the database. Each function—Create, Read, Update, and Delete—plays a vital role in managing data dynamically on the server.

- **Create:** Users could enter new data through an HTML form, which PHP then processes and stores in the database.
- **Read:** Data retrieval was implemented using SELECT queries, and the results were presented in a clean tabular format. This is essential for any data-driven application to provide visibility of stored information.
- **Update:** The update functionality allowed modification of existing records. This part reinforced the importance of pre-filling form fields with current data for better user experience and using SQL UPDATE queries.
- **Delete:** The deletion was handled using a simple hyperlink with the user ID passed as a GET parameter. A DELETE SQL command executed the operation.