Course Scheduling System

Cohesion:

In the Course Scheduler, we designed our modules to exhibit high cohesion by carefully defining each module's responsibility. This clear separation allows each module to focus on a specific part of the scheduling process, making the codebase more modular, easier to debug, and straightforward to extend.

1.Function Cohesion

Example Files: addclassrooms.php, deleteclassroom.php

```php
<?php
// addclassrooms.php
require 'connection.php';

if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    $classroom_name = $_POST['classroom_name'];
    $capacity = $_POST['capacity'];

    // Insert classroom into the database
    $sql = "INSERT INTO classrooms (name, capacity) VALUES (?, ?)";
    $stmt = $conn->prepare($sql);
    $stmt->bind_param("si", $classroom_name, $capacity);
    $stmt->execute();
    echo "Classroom added successfully.";
}
?>
```

2.Sequential Cohesion

Example Files: deleteclassroom.php

```php
<?php
// deleteclassroom.php
require 'connection.php';

$classroom_id = $_POST['classroom_id'];

// Step 1: Delete any associated allotments with the classroom
$sql_delete_allotments = "DELETE FROM allotments WHERE classroom_id = ?";
$stmt = $conn->prepare($sql_delete_allotments);
$stmt->bind_param("i", $classroom_id);
$stmt->execute();

// Step 2: Delete the classroom itself
$sql_delete_classroom = "DELETE FROM classrooms WHERE id = ?";
$stmt = $conn->prepare($sql_delete_classroom);
$stmt->bind_param("i", $classroom_id);
$stmt->execute();

echo "Classroom and associated allotments deleted successfully.";
?>
```

3.Temporal Cohesion

```php
<?php
// connection.php
$servername = "localhost";
$username = "root";
$password = "";
$database = "course_scheduler";

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

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

Coupling:

In the Course Scheduler, we designed modules to have loose coupling by minimizing dependencies and interactions between them. This ensures that changes in one module don't require extensive changes in others, which enhances the maintainability and scalability of the application.

1.Data Coupling

```php
// connection.php
$servername = "localhost";
$username = "root";
$password = "";
$database = "course_scheduler";
$conn = new mysqli(hostname: $servername, username: $username,
 password: $password, database: $database);

// addclassrooms.php
require 'connection.php'; // Only requires the $conn object
$classroom_name = $_POST['classroom_name'];
$capacity = $_POST['capacity'];

$sql = "INSERT INTO classrooms (name, capacity) VALUES (?, ?)";
$stmt = $conn->prepare(query: $sql);
$stmt->bind_param(types: "si", var: &$classroom_name, vars: &$capacity);
$stmt->execute();
echo "Classroom added successfully.";
```

2.Stamp Coupling

```php
include 'connection.php';
if (isset($_POST['UN']) && isset($_POST['PASS'])) {
    $id = $_POST['UN'];
    $password = $_POST['PASS'];
} else {
    die();
}
$q = mysqli_query(mysql: mysqli_connect
(hostname: "localhost",
username: "root", password: "", database: "ttms"),
query: "SELECT name                    e = '$id'
 and password = '$pa
if (mysqli_num_rows(result: $q) == 1) {
    echo 'welcome admin';
} else {
    $message = "Username and/or Password incorrect.\\nTry again.";
    echo "<script type='text/javascript'>alert('$message');</script>";
}
```

Double-click to insert

2

3.Control Coupling

```php
require 'connection.php';

Tabnine | Edit | Test | Explain | Document | Ask | 1 reference
function generateTimetable($type): void {
    global $conn;

    if ($type == "daily") {
        $sql = "SELECT * FROM timetable WHERE type = 'daily'";
    } else if ($type == "weekly") {
        $sql = "SELECT * FROM timetable WHERE type = 'weekly'";
    }

    $result = $conn->query($sql);
    while ($row = $result->fetch_assoc()) {
        echo "Timetable: " . $row['details'] . "<br>";
    }
}

// Calling the function with control flag
generateTimetable(type: "daily");
```

Summary of Cohesion

File	Cohesion Type	Description
addclassrooms.php	Functional Cohesion	Handles the addition of classrooms.
deleteclassroom.php	Sequential Cohesion	Deletes related records, then the classroom.
connection.php	Temporal Cohesion	Initializes the database connection for other files.

COUPLING SUMMARY

Coupling Type	Example File	Description
Data Coupling	addclassrooms.php + connection.php	Simple data (connection object) is shared without implementation details.
Stamp Coupling	assignSubstituteForm.php	Composite data structure (substituteDetails) passed between modules.
Control Coupling	generateTimetable.php	Control flag ($type) influences the behavior of another module.

4

5