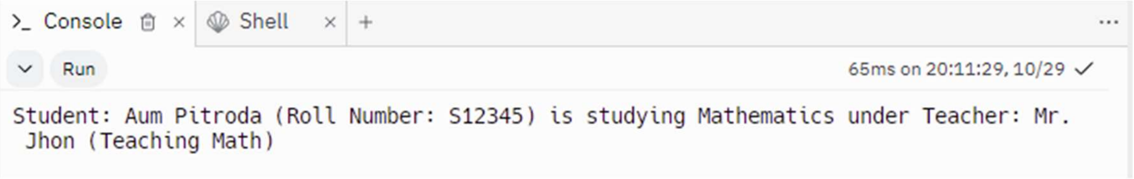
Lab #4

#

Lab#4(a) Implement OOP concepts with PHP for following program: Use Student, Teacher and Subject Class to define relation between each of them. Display the details of the student studying a particular subject under a particular teacher. Use Constructors and Polymorphism.

TAGS USED	ATTRIBUTES USED	DESCRIPTION
<?php ?>	--	The opening and closing tags to denote PHP code blocks. All PHP code goes between these tags.
spl_autoload_register	--	Registers a function as an autoloader, which is called when a class is accessed and not yet defined. It is used to automatically load classes as needed.
_DIR	--	A magic constant that returns the directory of the current file.
file_exists	--	Checks if a file or directory exists.
require_once	--	Includes and evaluates a specified file only once during a script's execution. It is used to load class files.
echo	--	Outputs one or more strings. It is used to print content to the output
foreach	--	Iterates over arrays and objects and executes a block of code for each element.
if	--	Executes a block of code if a specified condition is true.
public, private, protected	--	Access modifiers used to define the visibility of class properties and methods.
return	--	Exits a function and returns a value to the caller.

Final Outcome:



```
Student: Aum Pitroda (Roll Number: S12345) is studying Mathematics under Teacher: Mr.
 Jhon (Teaching Math)
```

Code Snippet:

```
<?php     class
Student {
   private $name; private
   $subjects = [];

   public function_construct($name) { $this->name
      = $name;
   }

   public function getName() {
      return $this->name;
   }

   public function addSubject(Subject $subject) { $this->subjects[]
      = $subject;
   }

   public function getSubjects() {
      return $this->subjects;
   }
}
?>
```

Teacher.php:

```
<?php     class
Teacher {
   private $name; private
   $subjects = [];

   public function_construct($name) { $this->name
      = $name;
   }

   public function getName() {
        return $this->name;
    }

    public function addSubject(Subject $subject) { $this->subjects[]
        = $subject;
    }
    public function getSubjects() {
        return $this->subjects;
    }
}
?>
```

Subject.php:

```
<?php
class Subject {
    private $name; private
    $teacher;

    public function_construct($name, Teacher $teacher) {
        $this->name = $name;
        $this->teacher = $teacher;
    }

    public function getName() {
        return $this->name;
    }

    public function getTeacherName() {
        return $this->teacher->getName();
    }
}
?>
```

Index.php:

```
<?php
spl_autoload_register(function      ($class)      {
    $classPath =_DIR_. '/classes/' . $class . '.php'; if
    (file_exists($classPath)) {
        require_once $classPath;
    }
});
```

```
$teacher1 = new Teacher("Jiya Manek");
$teacher2 = new Teacher("Kaushal Nathnani");
$teacher3 = new Teacher("Alis Khachar");

$mathSubject = new Subject("Math", $teacher1);
$scienceSubject = new Subject("Science", $teacher2);
$historySubject = new Subject("History", $teacher3); $student1 =
new Student("Ameesha Parmar ");
$student2 = new Student("Jency Patel ");
$student3 = new Student("Ridhi Keraliya ");

// Add subjects to students
$student1->addSubject($mathSubject);
$student1->addSubject($scienceSubject);
$student2->addSubject($scienceSubject);
$student3->addSubject($historySubject);

function displayStudentSubjectDetails(Student $student, $subjectName) {
   $subjects = $student->getSubjects(); foreach
   ($subjects as $subject) {
      if ($subject->getName() === $subjectName) {
         $studentName = $student->getName(); $teacherName =
         $subject->getTeacherName(); echo "Student Name:
         {$studentName}<br>"; echo "Subject Name:
         {$subjectName}<br>"; echo "Teacher Name:
         {$teacherName}<br><br>"; return;
      }
   }
}


displayStudentSubjectDetails($student1, "Math");
displayStudentSubjectDetails($student2, "Science");
displayStudentSubjectDetails($student3, "History");

?>
```