Name=Divya Sondagar

MODULE – 4(Advance PHP)

OOPS

1.What Is Object Oriented Programming?

OOP stands for Object-Oriented Programming.
Procedural programming is about writing procedures or functions that perform operations on the data, while object-oriented programming is about creating objects that contain both data and functions.

2.What Is Object Oriented Programming?
Object-oriented programming has several advantages over procedural programming:

OOP is faster and easier to execute
OOP provides a clear structure for the programs
OOP helps to keep the PHP code DRY "Don't Repeat Yourself", and makes the code easier to maintain, modify and debug
OOP makes it possible to create full reusable applications with less code and shorter development time

3. What Are Properties Of Object Oriented Systems?
Properties of Object-Oriented Systems

Encapsulation: Bundling data (attributes) and methods (functions) that operate on that data within a single unit called an object.

Abstraction: Simplifying complex reality by modeling classes on essential properties and behaviors relevant to the problem.

Inheritance: Creating new classes (subclasses) that inherit properties and methods from existing classes (superclasses).

Polymorphism: The ability of objects of different types to be treated as if they were of the same type.

4. What Is Difference Between Class And Interface?
Class: A blueprint for creating objects. It defines properties and methods.
Interface: A contract that defines methods that a class must implement. It doesn't contain implementation details.

Name=Divya Sondagar

5. What Is Overloading?
Overloading refers to the ability to have multiple methods with the same name but different parameters.

6.What Is T_PAAMAYIM_NEKUDOTAYIM (Scope Resolution Operator (::) with Example Sure, I can remove the dots from this page for you. Is there anything else I can assist you with today?
The scope resolution operator also known as Paamayim Nekudotayim or more commonly known as the double colon is a token that allows access to static, constant, and overridden properties or methods of a class.
It is used to refer to blocks or codes in context to classes, objects, etc. An identifier is used with the scope resolution operator. The most common example of the application of the scope resolution operator in PHP is to access the properties and methods of the class.

7. What are the differences between abstract classes and interfaces?
Abstract Class:
o Abstract class comes under partial abstraction.
o Abstract classes can maintain abstract methods and non abstract methods.
o In abstract classes, we can create the variables.
o In abstract classes, we can use any access specifier.
o By using 'extends' keyword we can access the abstract class features from derived class.
o Multiple inheritance is not possible.
Interface:
o Interface comes under fully abstraction.
o Interfaces can maintain only abstract methods.
o In interfaces, we can't create the variables.
o In interface, we can use only public access specifier.
o By using 'implement' keyword we can get interface from derived class.
o By using interfaces multiple inheritance is possible.

8. Define Constructor and Destructor?
Constructor : Constructors are the blueprints for object creation providing values for member functions and member variables.
Syntax:

```
Class  __construct():

function __construct()
      {
      // initialize the object and its properties by assigning
      //values
      }
```

Destructor: Once the object is initialized, the constructor is automatically called. Destructors are for destroying objects and automatically called at the end of execution.
Syntax:

Name=Divya Sondagar

class__destruct():

```
function __destruct()
      {
      // destroying the object or clean up resources here
      }
```

9. How to Load Classes in PHP?
Manual Loading:
Traditionally, you include class files using require_once or include_once statements.
For example, if you have a class named Contact defined in a file named Contact.php, you can load it like this:

```
require_once 'models/Contact.php';
$contact = new Contact('john.doe@example.com');
```

Autoloading with spl_autoload_register():
PHP introduced the spl_autoload_register() function to automatically load classes when needed.
Instead of manually including files, you register an autoloading function that PHP calls when a class is not yet loaded.

```
function load_model($class_name) {
   $path_to_file = 'models/' . $class_name . '.php';
   if (file_exists($path_to_file)) {
      require $path_to_file;
   }
}

spl_autoload_register('load_model');
```

10. How to Call Parent Constructor?
Automatic Parent Constructor Call:

When you create an instance of a child class, PHP automatically searches for the constructor in the child class.
If the child class doesn't have its own constructor, PHP looks for the constructor in the parent class.
It then invokes the parent class constructor.
Explicitly Calling Parent Constructor:

If the child class has its own constructor, you can explicitly call the parent constructor using parent::__construct(arguments).

11. Are Parent Constructor Called Implicitly When Create An ObjectOf Class?

```
<?php

class ParentClass {
   public function __construct() {
        echo "Parent constructor called.\n";
    }
}

class ChildClass extends ParentClass {
    public function __construct() {
        parent::__construct(); // Explicitly call parent constructor
        echo "Child constructor called.\n";
    }
}

$childObj = new ChildClass();
?>
```

12.What Happen, If Constructor Is Defined As Private Or Protected?

```
<?php
class A{
    public function __construct(){
        echo 'Instance of A created';
    }
}

class B{
    protected function __construct(){
        echo 'Instance of B created';
    }
}

class C{
    private function __construct(){
        echo 'Instance of C created';
    }
}
?>
```

Output:

```
new A; // OK
 new B; // Fatal error: Call to protected B::__construct() from invalid context
new C; // Fatal error: Call to private C::__construct() from invalid context
```

13.What are PHP Magic Methods/Functions? List them Write program for Static Keyword in PHP?

MAGIC METHODS:

__construct()

This method gets called automatically every time the object of a particular class is created. The function of this magic method is the same as the constructor in any OOP language.

Name=Divya Sondagar

__destruct()
As the name suggests this method is called when the object is destroyed and no longer in use.
Generally at the end of the program and end of the function.

__call($name,$parameter)
__toString()
__get($name)
__set($name , $value)
__debugInfo()
Static keyword program:-

```
<?php
class MyClass {
  public static $str = "Hello World!";

  public static function hello() {
    echo MyClass::$str;
  }
}

echo MyClass::$str;
echo "<br>";
echo MyClass::hello();
?>
```

Output:

```
Hello World!
Hello World!
```

14.Create multiple Traits and use it in to a single class?

```
<?php
trait message1 {
  public function msg1() {
    echo "OOP is fun! ";
  }
}

trait message2 {
  public function msg2() {
    echo "OOP reduces code duplication!";
  }
}

class Welcome {
  use message1;
}

class Welcome2 {
  use message1, message2;
}

$obj = new Welcome();
$obj->msg1();
echo "<br>";

$obj2 = new Welcome2();
$obj2->msg1();
$obj2->msg2();
?>
```

Output:

```
OOP is fun!
OOP is fun! OOP reduces code duplication!
```

15.Write PHP Script of Object Iteration?

```
<?php
class MyClass {
    public $var1 = 'value 1';
    public $var2 = 'value 2';
    public $var3 = 'value 3';
    protected $protected = 'protected var';
    private $private = 'private var';

    function iterateVisible() {
        echo "MyClass::iterateVisible:\n";
        foreach ($this as $key => $value) {
            print "$key => $value\n";
        }
    }
}

$class = new MyClass();

// Iterate through visible properties
foreach ($class as $key => $value) {
    print "$key => $value\n";
}

// Call custom method for visible properties
$class->iterateVisible();
?>
```

Output:

```
var1 => value 1
var2 => value 2
var3 => value 3
```

Name=Divya Sondagar

```
MyClass::iterateVisible:
var1 => value 1
var2 => value 2
var3 => value 3
protected => protected var
private => private var
```

16.Use of The $this keyword

In PHP, $this is a special keyword that represents the instance of the class where it is used. It allows access to the properties and methods of the current object within the class context. $this is only available within methods of a class, and attempting to use it outside of a class context will result in an error.

Example:-

```
<?php
        Class MyClass
{
        Public $mySample="Hello !";
        Public function printSample()
{
        Echo $this->mySample;
}
}
$obj=new MyClass();
$obj->printSample();
?>
```

• Create Hotel Room Booking System User can book room by 3 ways

1.Website Booking:
User Interface:
Homepage: Displays available rooms, promotions, and general information.
Search Page: Allows users to filter rooms based on criteria (e.g., date, number of guests).
Room Details: Provides information about each room, including amenities and prices.
Booking Form: Users can input details such as check-in/out dates, the number of guests, and payment information.
Backend Logic:
Validate user inputs and check room availability.
Calculate total cost based on room rates and selected dates.
Secure payment processing.
Send confirmation email with booking details.
Database:
Store room information (availability, rates, amenities).
Maintain user data (bookings, personal information).
2.Mobile App Booking:
User Interface:
Similar to the website but optimized for mobile devices.
Responsive design for various screen sizes.

Name=Divya Sondagar

Backend Logic:
Sure, I can definitely help with that. What kind of objective facts would you like to discuss?Utilize APIs for seamless communication between the app and server.Sure, I can definitely help with that. What kind of objective facts would you like to discuss?
Implement push notifications for booking updates.
Offline functionality for viewing bookings without an internet connection.
Database:
Synchronize data between the app and the central database.
3.Phone Call Booking:
Interactive Voice Response (IVR) System:
Welcome and guide the user through the booking process.
Voice recognition for user input.
Provide options for room selection, dates, and payment.
Backend Logic:
Convert voice inputs to text and process them.
Check room availability and calculate the total cost.
Generate a booking confirmation.
Database:
Update room availability and store booking details.
• Full day
1. User Registration:
oUsers need to register for an account to book a room.
oCollect basic information like name, email, and password for registration.
oImplement authentication to secure user accounts.
2. Room Selection:
oDisplay a list of available rooms with details (e.g., room type, price, amenities).
oAllow users to select a room for booking.
3. Booking Methods:
oa. Online Booking:
Users can log in, choose the desired room, and make an online payment.
Integrate a payment gateway for secure transactions.
Upon successful payment, confirm the booking and provide a booking ID.
ob. Walk-in Booking:
Users can physically visit the hotel and book a room at the reception.
Receptionist enters user details and assigns a room.
Generate a booking ID and provide it to the user.
oc. Phone Booking:
Users can call the hotel to inquire about room availability.
Hotel staff can check availability, take user details over the phone, and book the room.
Provide a booking confirmation with a unique ID.
4.Reservation Management:
oImplement a system to manage reservations.
oAvoid double bookings by updating room availability in real-time.
oProvide an admin interface to view and manage bookings.
5. User Dashboard:
oUsers can log in to their accounts to view their booking history.
oDisplay upcoming and past bookings with details.
6. Notifications:

Name=Divya Sondagar

oSend email/SMS notifications for successful bookings and reminders.
oNotify users of any changes to their reservations.
7. Cancellation:
oAllow users to cancel bookings with a refund policy.
oImplement cancellation confirmation and refund processing.
8. Reporting:
oGenerate reports for the hotel staff to analyze booking trends, occupancy rates, etc.
9. Security Measures:
oImplement secure coding practices.
oUse HTTPS for secure data transmission.
oRegularly update and patch the system to prevent security vulnerabilities.

• Half day
1. User Registration:
oUsers should be able to create an account or log in if they already have one.
oCollect necessary information such as name, email, contact details, and password.
2. Room Selection:
oDisplay a list of available rooms with details like room type, price, and availability.
oAllow users to choose a room based on their preferences.
oImplement a calendar to show room availability for half-day bookings.
3. Booking Process:
oUsers can initiate a booking by selecting the check-in and check-out dates and times.
oFor half-day bookings, provide options for morning or afternoon check-in/check-out.
oCalculate the total cost based on the selected room and duration.
4. Payment Integration:
oIntegrate a secure payment gateway to handle transactions.
oAccept various payment methods (credit/debit cards, online wallets, etc.).
oConfirm the booking only after successful payment.
5. Booking Confirmation:
oSend a confirmation email or message to the user with booking details.
oDisplay a confirmation page with a summary of the booking.
6. User Dashboard:
oProvide a user dashboard where users can view their booking history.
oAllow users to cancel or modify bookings within a specified timeframe.
7. Admin Panel:
oImplement an admin panel for hotel staff to manage room availability, bookings, and user accounts.
oAllow admins to add new rooms, update prices, and view booking reports.
8. Notifications:
oSend reminders to users before their check-in/check-out time.
oNotify users of successful bookings, cancellations, or any changes to their reservation.
9. Reviews and Feedback:
oAllow users to leave reviews and ratings for their stay.
oDisplay reviews to help future users make informed decisions.
10. Security:
oImplement secure authentication and authorization mechanisms.
oEnsure that sensitive information such as payment details is encrypted.
11. Mobile Responsiveness:

Name=Divya Sondagar

oDesign the system to be mobile-friendly for users who prefer booking on smartphones or tablets.
12. Analytics:
oIntegrate analytics to track user behavior, popular rooms, and booking patterns.

• Custom
Hotel Room Booking System Components:
1.Database:
Create a database to store information about rooms, reservations, and users. Tables could include Rooms, Reservations, and Users.
2.Backend:
Use a backend framework (e.g., Flask, Django, Express.js) to handle server-side logic and interact with the database.
3.User Authentication:
Implement a user authentication system to secure user accounts and booking history.
4.Web Interface:
Create a web interface for users to book rooms. Use HTML, CSS, and JavaScript for the frontend. Ensure a responsive design for different devices.
5.Mobile App:
Develop a mobile app (iOS/Android) using a framework like React Native or Flutter, providing similar functionalities to the web interface.
6.Phone Booking:
Allow users to book rooms over the phone by calling a dedicated reservation hotline. Staff members can use a secure admin panel to manually enter reservations made via phone.
7.Room Availability:
Implement a mechanism to check room availability for specific dates. Update the availability status in real-time.
8.Payment Integration:
Integrate a payment gateway for online bookings. Ensure secure handling of payment information.

• If user select for the full day than user only have selection for the checking checkout date
1.Radio Buttons for Stay Option:
Full Day
Custom Stay
2.Date Picker for Check-In:
Check-In Date: [Date Picker]
3.Date Picker for Check-Out (Conditional):
Check-Out Date: [Date Picker]

• If user select Half day than user have option of date and slot option(like user want to book room for first half – Morning (8AM to 6PM) if user select for second halfit‟s for evening (7PM to Morning 7AM)). Do proper validation like if user can book only available slot. (have touse jQuery -> Ajax, validation, Json passing).
To implement the described functionality with jQuery, Ajax, and JSON passing for booking a room with date and slot options, you can follow these steps:
1. HTML Structure :
<!DOCTYPE html>

Name=Divya Sondagar

```
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Room Booking</title>
  <!-- Include jQuery -->
  <script src="https://code.jquery.com/jquery-3.6.4.min.js"></script>
</head>
<body>
  <h2>Room Booking</h2>

  <label for="bookingType">Select Booking Type:</label>
  <select id="bookingType">
    <option value="fullDay">Full Day</option>
    <option value="halfDay">Half Day</option>
  </select>
  <div id="dateSlotOptions" style="display:none;">
    <label for="bookingDate">Select Date:</label>
    <input type="date" id="bookingDate">
    <label for="bookingSlot">Select Slot:</label>
    <select id="bookingSlot">
      <option value="morning">Morning (8AM to 6PM)</option>
      <option value="evening">Evening (7PM to 7AM)</option>
    </select>
  </div>
  <button id="submitBooking">Submit Booking</button>
  <script src="script.js"></script>
</body>
</html>
```

1.JavaScript (script.js):

```
$(document).ready(function()
{
  // Event listener for booking type change
  $("#bookingType").change(function()
{
    if ($(this).val() === "halfDay")
{
      $("#dateSlotOptions").show();
    }
else
{
      $("#dateSlotOptions").hide();
    }
  });
  // Event listener for submit button
  $("#submitBooking").click(function()
{
    // Validate and gather user inputs
```

Name=Divya Sondagar

```
    var bookingType = $("#bookingType").val();
    var bookingDate = $("#bookingDate").val();
    var bookingSlot = $("#bookingSlot").val();
    // Validate user inputs
    if (bookingType === "halfDay" && (!bookingDate || !bookingSlot))
{
      alert("Please select date and slot for half-day booking.");
      return;
    }

    // Construct data to send via Ajax
    var bookingData =
{
      type: bookingType,
      date: bookingDate,
      slot: bookingSlot
    };
    // Perform Ajax request
    $.ajax({
      url: "your_booking_endpoint.php", // Replace with your server-side endpoint
      type: "POST",
      data: JSON.stringify(bookingData),
      contentType: "application/json; charset=utf-8",
      dataType: "json",
      success: function(response)
{
        // Handle success response from server
        alert("Booking successful!");
      },
      error: function(error)
{
        // Handle error response from server
        alert("Booking failed. Please try again.");
      }
    });
  });
});
```

2.Server-side (your_booking_endpoint.php):

```
<?php
header('Content-Type: application/json');

// Retrieve JSON data from the request
$data = json_decode(file_get_contents('php://input'), true);

// Validate and process booking
if ($data['type'] === 'halfDay')
{
```

Name=Divya Sondagar

```
    // Perform validation for half-day booking and process accordingly
    // Replace this with your actual validation and booking logic
    $isValidBooking = true; // Replace with your validation logic

    if ($isValidBooking)
{
        // Booking successful
        $response = array('status' => 'success', 'message' => 'Booking successful!');
    }
else
{
        // Booking failed
        $response = array('status' => 'error', 'message' => 'Invalid date or slot for half-day
booking.');
    }
}
else
{
    // Process full-day booking
    // Replace this with your logic for full-day booking
    $response = array('status' => 'success', 'message' => 'Booking successful!');
}

// Send JSON response back to the client
echo json_encode($response);
?>
```

Jquery

What is jQuery?

jQuery is a fast, small, and feature-rich JavaScript library. It simplifies the process of traversing HTML documents, handling events, creating animations, and managing Ajax interactions. jQuery was created by John Resig and was first released in 2006. It quickly gained popularity due to its ease of use and the ability to perform common tasks with less code compared to raw JavaScript.

Some key features of jQuery include:

1.DOM Manipulation: jQuery simplifies the process of selecting and manipulating HTML elements on the page. It provides a concise syntax for common tasks like changing content, attributes, and styles.

2.Event Handling: jQuery makes it easy to handle various events, such as clicks, keypresses, and mouse movements. Event handling in jQuery is straightforward and cross-browser compatible.

3.Ajax Support: jQuery simplifies asynchronous JavaScript and XML (Ajax) requests, allowing developers to fetch data from a server and update parts of a web page without requiring a full page refresh.

4.Animation Effects: jQuery includes built-in functions for creating animations and transitions, making it easier to add visual effects to web pages.

Name=Divya Sondagar

5. Utilities: jQuery includes a variety of utility functions that streamline common tasks, such as working with arrays and objects, making AJAX requests, and handling browser compatibility issues.

How are JavaScript and jQuery different?

1.Core Purpose:
oJavaScript: It is a general-purpose programming language that can be used for a wide range of tasks, not limited to web development. It is supported by all major web browsers and can be used for both client-side and server-side development.
ojQuery: It is a lightweight, cross-platform JavaScript library designed to simplify the client-side scripting of HTML. jQuery makes it easier to perform common tasks like DOM manipulation, event handling, animation, and AJAX with less code.

2.Syntax:
oJavaScript: It is a programming language with its syntax. It provides a way to interact with the DOM (Document Object Model) and handle events, among other things.
ojQuery: It is essentially a set of functions written in JavaScript. jQuery syntax is often considered simpler and more concise than raw JavaScript, especially for tasks like DOM manipulation.

3.DOM Manipulation:
oJavaScript: It uses native methods to manipulate the DOM. For example, document.getElementById(), document.createElement(), etc.
ojQuery: It provides a simplified and often more cross-browser compatible syntax for DOM manipulation. For instance, $("#elementId") or $(".className") to select elements.

4.Event Handling:
oJavaScript: Event handling in JavaScript can be done using the addEventListener method or by assigning event attributes directly in the HTML.
ojQuery: It simplifies event handling with methods like click(), change(), etc. For example, $("#myButton").click(function() { /* do something */ });.

5.Ajax:
oJavaScript: Native JavaScript provides the XMLHttpRequest object for making AJAX requests.
ojQuery: jQuery simplifies AJAX requests with methods like $.ajax() or shorthand methods like $.get(), $.post().

6.Cross-browser Compatibility:
oJavaScript: Developers need to be aware of and handle cross-browser compatibility issues themselves.
ojQuery: One of the main reasons jQuery gained popularity was its ability to abstract away many cross-browser inconsistencies, providing a more consistent interface.

Name=Divya Sondagar

7.Size and Performance:
oJavaScript: Native JavaScript code can sometimes be more lightweight because it doesn't include the additional abstraction layer that jQuery provides.
ojQuery: While it simplifies coding, it adds an extra layer, and the library itself adds some file size overhead. In modern web development, where bandwidth and performance are crucial, some developers opt to use JavaScript directly for smaller projects.

Which is the starting point of code execution in jQuery?
The starting point of code execution is often inside a document-ready event handler. The document-ready event occurs when the DOM (Document Object Model) has been fully loaded and is ready to be manipulated. This ensures that the jQuery code runs after the HTML document has been parsed.
Syntax:-

```
$(document).ready(function()
{
   // jQuery code goes here
});
```

Document Load Vs Window. Load() jQuery
$(document).ready():
This event occurs when the DOM (Document Object Model) is ready, which means the HTML structure of the page has been fully loaded and parsed.
It is triggered as soon as the DOM is ready, even if other external resources like images, stylesheets, and scripts are still loading.
It is suitable for tasks that don't require waiting for all external resources to be fully loaded.

Example:

```
$(document).ready(function()
{
   // Your code here
});
```

$(window).load():
This event occurs when all assets on the page, including images and other resources, have finished loading.
It waits for the entire page, including external resources, to be fully loaded before triggering.
It is suitable for tasks that depend on the dimensions or content of images, as it ensures that all images have been loaded.
Example:

```
$(window).load(function()
{
   // Your code here
});
```

What is the difference between prop and attr?

Name=Divya Sondagar

1.Attribute (attr):
In HTML, attributes provide additional information about HTML elements and are always included in the opening tag of an element.
Attributes define the properties of an HTML element and are used to configure or modify the behavior of the element.
Examples of attributes include class, id, src, href, alt, etc.
Attributes are part of the HTML document and can be accessed and modified using JavaScript.

Example:

```
<img src="image.jpg" alt="An example image" class="my-image">
```

2.Property (prop):
In the context of JavaScript and the Document Object Model (DOM), properties are values associated with JavaScript objects.
Elements in the DOM are represented as objects, and these objects have properties that can be accessed and modified using JavaScript.
Properties often represent the current state of an element, and they can be dynamic, changing based on user interactions or other events.
Examples of properties include innerHTML, value, style, etc.

Example:

```
var myElement = document.getElementById("exampleElement");
console.log(myElement.innerHTML);     // Accessing the innerHTML property of an element
```

Explain Difference Between JQuery And JavaScript?
1.Core Purpose:
oJavaScript: It is a general-purpose programming language that can be used for a wide range of tasks, not limited to web development. It is supported by all major web browsers and can be used for both client-side and server-side development.
ojQuery: It is a lightweight, cross-platform JavaScript library designed to simplify the client-side scripting of HTML. jQuery makes it easier to perform common tasks like DOM manipulation, event handling, animation, and AJAX with less code.

2.Syntax:
oJavaScript: It is a programming language with its syntax. It provides a way to interact with the DOM (Document Object Model) and handle events, among other things.
ojQuery: It is essentially a set of functions written in JavaScript. jQuery syntax is often considered simpler and more concise than raw JavaScript, especially for tasks like DOM manipulation.

3.DOM Manipulation:
oJavaScript: It uses native methods to manipulate the DOM. For example, document.getElementById(), document.createElement(), etc.
ojQuery: It provides a simplified and often more cross-browser compatible syntax for DOM manipulation. For instance, $("#elementId") or $(".className") to select elements.

Name=Divya Sondagar

4.Event Handling:
oJavaScript: Event handling in JavaScript can be done using the addEventListener method or by assigning event attributes directly in the HTML.
ojQuery: It simplifies event handling with methods like click(), change(), etc. For example, $("#myButton").click(function() { /* do something */ });.

5.Ajax:
oJavaScript: Native JavaScript provides the XMLHttpRequest object for making AJAX requests.
ojQuery: jQuery simplifies AJAX requests with methods like $.ajax() or shorthand methods like $.get(), $.post().

6.Cross-browser Compatibility:
oJavaScript: Developers need to be aware of and handle cross-browser compatibility issues themselves.
ojQuery: One of the main reasons jQuery gained popularity was its ability to abstract away many cross-browser inconsistencies, providing a more consistent interface.
Sure, I can help with that. To remove the paragraph layout from a full file, you can open the document in a word processing program such as Microsoft Word and select all the text. Then, go to the "format" or "layout" menu and choose the option to remove paragraph or line spacing. This will adjust the formatting of the text to remove any paragraph layout. Let me know if you need further assistance.
7.Size and Performance:
oJavaScript: Native JavaScript code can sometimes be more lightweight because it doesn't include the additional abstraction layer that jQuery provides.
ojQuery: While it simplifies coding, it adds an extra layer, and the library itself adds some file size overhead. In modern web development, where bandwidth and performance are crucial, some developers opt to use JavaScript directly for smaller projects.

How We Can Select The Specified <li> Element From The ListOf <li> Elements In <ul>?
To select a specific <li> element from a list of <li> elements within a <ul> (unordered list) using JavaScript, you can use various methods.
Example:
1. Using JavaScript and the DOM:
Assuming you have an unordered list with an id, let's say "myList":

```
<ul id="myList">
  <li>Item 1</li>
  <li>Item 2</li>
  <li>Item 3</li>
</ul>
```

JavaScript code to select the second <li> element:

```
// Get the <ul> element by its id
var myList = document.getElementById("myList");
```

Name=Divya Sondagar

```
// Get the second <li> element (index 1, as indices start from 0)
var secondLi = myList.getElementsByTagName("li")[1];

// Now 'secondLi' contains the reference to the second <li> element
console.log(secondLi.textContent);
```

2.Using querySelector:

```
// Use querySelector to directly select the second <li> element
var secondLi = document.querySelector("#myList li:nth-child(2)");

// Now 'secondLi' contains the reference to the second <li> element
console.log(secondLi.textContent);
```

3. Using querySelectorAll:

```
var allLiElements = document.querySelectorAll("#myList li"); // Use querySelectorAll to select
all <li> elements and then choose the second one

var secondLi = allLiElements[1]; // Get the second <li> element (index 1, as indices start
from 0)

console.log(secondLi.textContent);  // Now 'secondLi' contains the reference to the second
<li> element
```

In <table> Design Change The Color Of Even <tr> Elements To “green” And   Change The Color Off Odd <tr> Elements To “blue” Color? Give An Example Code?
the even and odd <tr> elements differently. Here's an example code:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <style>
        table
        {
            width: 100%;
            border-collapse: collapse;
        }

        tr:nth-child(even)
{
            background-color: green;
        }

        tr:nth-child(odd)
{
            background-color: blue;
        }

        th, td
{
            border: 1px solid black;
            padding: 8px;
            text-align: left;
        }
    </style>
</head>
<body>

    <table>
        <thead>
            <tr>
                <th>Header 1</th>
                <th>Header 2</th>
                <th>Header 3</th>
            </tr>
        </thead>
        <tbody>
            <tr>
                <td>Row 1, Cell 1</td>
                <td>Row 1, Cell 2</td>
                <td>Row 1, Cell 3</td>
            </tr>
            <tr>
                <td>Row 2, Cell 1</td>
                <td>Row 2, Cell 2</td>
                <td>Row 2, Cell 3</td>
            </tr>
            <tr>
                <td>Row 3, Cell 1</td>
                <td>Row 3, Cell 2</td>
                <td>Row 3, Cell 3</td>
            </tr>
        </tbody>
    </table>
</body>
</html>
```

How We Can Implement Animation Effects In Jquery?
Implementing animation effects in jQuery involves using the built-in animation functions provided by jQuery. These functions allow you to animate the properties of HTML elements, such as their position, size, opacity, and more.

Include jQuery:

Name=Divya Sondagar

Ensure that you include the jQuery library in your HTML file. You can either download it and host it locally or use a CDN (Content Delivery Network). For example:

```
<script src="https://code.jquery.com/jquery-3.6.4.min.js"></script>
```

Select Elements: Use jQuery to select the HTML elements you want to animate. You can do this using the $ function.

```
<div id="myElement">Animate me!</div>
var myElement = $("#myElement");
```

Animate Properties: Use jQuery's animation functions to change the properties of the selected elements. Common properties include height, width, opacity, left, top, etc.

```
myElement.animate(
{
  opacity: 0.5,
  left: '+=50',
  height: 'toggle'
}, 1000); // 1000 is the duration in milliseconds
```

Chaining Animations: You can chain multiple animations together using the queue option or by simply calling additional animation functions.

```
myElement
  .animate({ left: '+=50' }, 500)
  .animate({ opacity: 0.5 }, 500);
```

Callback Functions: You can use callback functions to execute code after an animation completes.

```
myElement.animate({ opacity: 0.5 }, 1000, function() {
  // Code to execute after the animation is complete
});
```

Easing: jQuery provides easing options to control the speed of animations. You can use built-in options like 'swing' or 'linear', or include a custom easing function.

```
myElement.animate({ left: '+=50' }, { duration: 500, easing: 'linear' });
```

Stop Animation: You can stop an animation in progress using the stop function.

```
myElement.stop();
```

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
```

Name=Divya Sondagar

```
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <script src="https://code.jquery.com/jquery-3.6.4.min.js"></script>
   <title>jQuery Animation Example</title>
</head>
<body>

   <button id="animateButton">Animate</button>
   <div id="animatedDiv">Hello, I'm animated!</div>

   <script>
      $(document).ready(function()
{
         $("#animateButton").click(function()
{
            $("#animatedDiv").animate(
{
               left: '+=100',
               opacity: 0.5
            }, 1000);
         });
      });
   </script>

</body>
</html>
```

Apply jQuery validation using library.
jQuery Validation is a popular library for handling form validation in web applications. To use jQuery Validation, you need to include the jQuery library and the jQuery Validation plugin in your HTML file. Here's a step-by-step guide:

Include jQuery: Make sure to include the jQuery library in your HTML file. You can do this by adding the following line in the <head> section of your HTML:

```
<script src="https://code.jquery.com/jquery-3.6.4.min.js"></script>
```

Include jQuery Validation: Include the jQuery Validation plugin by adding the following line below the jQuery script tag:

```
<script src="https://cdn.jsdelivr.net/jquery.validation/1.19.3/jquery.validate.min.js"></script>
```

Create a Form: Create an HTML form that you want to validate. For example:

```
<form id="myForm">
   <label for="name">Name:</label>
   <input type="text" id="name" name="name" required>

   <label for="email">Email:</label>
```

Name=Divya Sondagar

```
  <input type="email" id="email" name="email" required>

  <input type="submit" value="Submit">
</form>
```

Initialize jQuery Validation: Add a script to initialize the jQuery Validation on your form. Place this script after including the jQuery and jQuery Validation scripts:

```
<script>
  $(document).ready(function ()
  {
    $("#myForm").validate();
  });
</script>
```

Create custom dynamic function for require field validator.
To create a custom dynamic function for a required field validator in JavaScript, you can define a function that takes an object as input and checks if the required fields are present. Here's an example of a simple dynamic required field validator function:

```
function createRequiredFieldValidator(requiredFields)
{
  return function (data)
{
    const missingFields = [];

    // Check each required field
    requiredFields.forEach(field =>
{
      if (!data.hasOwnProperty(field) || data[field] === null || data[field] === undefined ||
data[field] === '')
{
        missingFields.push(field);
      }
    });

    // Return the result
    if (missingFields.length === 0)
{
      return { isValid: true };
    }
else
{
      return { isValid: false, missingFields };
    }
  };
}
```

Name=Divya Sondagar

Get state data by country selection (Ajax).
To retrieve state data based on country selection using Ajax, you'll need to create a web page with HTML, JavaScript, and use a server-side script (like PHP, Node.js, or any server-side technology of your choice) to handle the data fetching. Below is a simple example using HTML, JavaScript, and PHP:

(HTML)Index.php

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Country and State Selection</title>
</head>
<body>
  <label for="country">Select Country:</label>
  <select id="country" onchange="getStates()">
    <!-- Populate this with your list of countries -->
    <option value="usa">USA</option>
    <option value="canada">Canada</option>
    <!-- Add more options as needed -->
  </select>

  <label for="state">Select State:</label>
  <select id="state"></select>

  <script>
    function getStates()
   {
      var countrySelect = document.getElementById('country');
      var stateSelect = document.getElementById('state');
      var selectedCountry = countrySelect.value;

      // Make an Ajax request to get states based on the selected country
      var xhr = new XMLHttpRequest();
      xhr.onreadystatechange = function()
     {
        if (xhr.readyState == 4 && xhr.status == 200)
       {
          // Parse the JSON response
          var states = JSON.parse(xhr.responseText);

          // Clear existing options
          stateSelect.innerHTML = '';

          // Populate the state dropdown with new options
          for (var i = 0; i < states.length; i++)
        {
          var option = document.createElement('option');
          option.value = states[i];
          option.text = states[i];
          stateSelect.add(option);
         }
       }
     };

     // Replace 'get_states.php' with the path to your server-side script
     xhr.open('GET', 'get_states.php?country=' + selectedCountry, true);
     xhr.send();
   }
  </script>
</body>
</html>
```

PHP (get_states.php):

```
<?php
// Replace this with your own logic to fetch states based on the selected country
if(isset($_GET['country'])) {
  $selectedCountry = $_GET['country'];

  // Example: Provide states based on the selected country
  $states = [];
  if($selectedCountry == 'usa') {
    $states = ['New York', 'California', 'Texas'];
  } elseif($selectedCountry == 'canada') {
    $states = ['Ontario', 'Quebec', 'British Columbia'];
  }

  // Send the states as a JSON response
  echo json_encode($states);
}
?>
```

Image uploading with preview.
=> To implement image uploading with a preview in a web application, you can use HTML, JavaScript, and CSS.

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Image Upload with Preview</title>
```

Name=Divya Sondagar

```
  <style>
    #preview-container
   {
      display: flex;
      justify-content: center;
      align-items: center;
      height: 200px;
      border: 2px dashed #ccc;
      margin-bottom: 20px;
    }

    #preview-img
   {
      max-width: 100%;
      max-height: 100%;
    }
  </style>
</head>
<body>

  <input type="file" id="imageInput" accept="image/*" onchange="previewImage(event)">
  <div id="preview-container">
    <img id="preview-img" src="#" alt="Image Preview">
  </div>

  <script>
    function previewImage(event)
{
      const input = event.target;
      const previewImg = document.getElementById('preview-img');
      const previewContainer = document.getElementById('preview-container');

      const file = input.files[0];

      if (file)
    {
        const reader = new FileReader();

        reader.onload = function (e)
      {
          previewImg.src = e.target.result;
        };

        reader.readAsDataURL(file);
        previewContainer.style.display = 'block';
      }
   else
    {
        previewImg.src = '#';
        previewContainer.style.display = 'none';
      }
    }
  </script>

</body>
</html>
```

MODULE – 4(Advance PHP)

OOPS

1.What Is Object Oriented Programming?

OOP stands for Object-Oriented Programming.
Procedural programming is about writing procedures or functions that perform operations on the data, while object-oriented programming is about creating objects that contain both data and functions.
2.What Is Object Oriented Programming?
Object-oriented programming has several advantages over procedural programming:

OOP is faster and easier to execute
OOP provides a clear structure for the programs
OOP helps to keep the PHP code DRY "Don't Repeat Yourself", and makes the code easier to maintain, modify and debug
OOP makes it possible to create full reusable applications with less code and shorter development time

3. What Are Properties Of Object Oriented Systems?
Properties of Object-Oriented Systems

Encapsulation: Bundling data (attributes) and methods (functions) that operate on that data within a single unit called an object.

Abstraction: Simplifying complex reality by modeling classes on essential properties and behaviors relevant to the problem.

Inheritance: Creating new classes (subclasses) that inherit properties and methods from existing classes (superclasses).

Polymorphism: The ability of objects of different types to be treated as if they were of the same type.

Name=Divya Sondagar

4. What Is Difference Between Class And Interface?
Class: A blueprint for creating objects. It defines properties and methods.
Interface: A contract that defines methods that a class must implement. It doesn't contain implementation details.

5. What Is Overloading?
Overloading refers to the ability to have multiple methods with the same name but different parameters. T
6.What Is T_PAAMAYIM_NEKUDOTAYIM (Scope Resolution Operator (::) with Example
Sure, I can remove the dots from this page for you. Is there anything else I can assist you with today?
The scope resolution operator also known as Paamayim Nekudotayim or more commonly known as the double colon is a token that allows access to static, constant, and overridden properties or methods of a class.
It is used to refer to blocks or codes in context to classes, objects, etc. An identifier is used with the scope resolution operator. The most common example of the application of the scope resolution operator in PHP is to access the properties and methods of the class.

7. What are the differences between abstract classes and interfaces?
Abstract Class:
o Abstract class comes under partial abstraction.
o Abstract classes can maintain abstract methods and non abstract methods.
o In abstract classes, we can create the variables.
o In abstract classes, we can use any access specifier.
o By using 'extends' keyword we can access the abstract class features from derived class.
o Multiple inheritance is not possible.
Interface:
o Interface comes under fully abstraction.
o Interfaces can maintain only abstract methods.
o In interfaces, we can't create the variables.
o In interface, we can use only public access specifier.
o By using 'implement' keyword we can get interface from derived class.
o By using interfaces multiple inheritance is possible.

8. Define Constructor and Destructor?
Constructor : Constructors are the blueprints for object creation providing values for member functions and member variables.
Syntax:

```
Class  __construct():

function __construct()
    {
    // initialize the object and its properties by assigning
    //values
    }
```

Name=Divya Sondagar

Destructor: Once the object is initialized, the constructor is automatically called. Destructors are for destroying objects and automatically called at the end of execution.
Syntax:
class__destruct():

```
function __destruct()
      {
      // destroying the object or clean up resources here
      }
```

9. How to Load Classes in PHP?
Manual Loading:
Traditionally, you include class files using require_once or include_once statements.
For example, if you have a class named Contact defined in a file named Contact.php, you can load it like this:

```
require_once 'models/Contact.php';
$contact = new Contact('john.doe@example.com');
```

Autoloading with spl_autoload_register():
PHP introduced the spl_autoload_register() function to automatically load classes when needed.
Instead of manually including files, you register an autoloading function that PHP calls when a class is not yet loaded.

```
function load_model($class_name) {
   $path_to_file = 'models/' . $class_name . '.php';
   if (file_exists($path_to_file)) {
      require $path_to_file;
   }
}

spl_autoload_register('load_model');
```

10. How to Call Parent Constructor?
Automatic Parent Constructor Call:

When you create an instance of a child class, PHP automatically searches for the constructor in the child class.
If the child class doesn’t have its own constructor, PHP looks for the constructor in the parent class.
It then invokes the parent class constructor.
Explicitly Calling Parent Constructor:

If the child class has its own constructor, you can explicitly call the parent constructor using parent::__construct(arguments).

11. Are Parent Constructor Called Implicitly When Create An ObjectOf Class?

Name=Divya Sondagar

```
<?php

class ParentClass {
    public function __construct() {
        echo "Parent constructor called.\n";
    }
}

class ChildClass extends ParentClass {
    public function __construct() {
        parent::__construct(); // Explicitly call parent constructor
        echo "Child constructor called.\n";
    }
}

$childObj = new ChildClass();
?>
```

12.What Happen, If Constructor Is Defined As Private Or Protected?

```
<?php
class A{
    public function __construct(){
        echo 'Instance of A created';
    }
}

class B{
    protected function __construct(){
        echo 'Instance of B created';
    }
}

class C{
    private function __construct(){
        echo 'Instance of C created';
    }
}
?>
```

Output:

```
new A; // OK
 new B; // Fatal error: Call to protected B::__construct() from invalid context
new C; // Fatal error: Call to private C::__construct() from invalid context
```

13.What are PHP Magic Methods/Functions? List them Write program for Static Keyword in PHP?

MAGIC METHODS:

Name=Divya Sondagar

__construct()
This method gets called automatically every time the object of a particular class is created.
The function of this magic method is the same as the constructor in any OOP language.

__destruct()
As the name suggests this method is called when the object is destroyed and no longer in use.
Generally at the end of the program and end of the function.

__call($name,$parameter)
__toString()
__get($name)
__set($name , $value)
__debugInfo()
Static keyword program:-

```
<?php
class MyClass {
  public static $str = "Hello World!";

  public static function hello() {
    echo MyClass::$str;
  }
}

echo MyClass::$str;
echo "<br>";
echo MyClass::hello();
?>
```

Output:

```
Hello World!
Hello World!
```

14.Create multiple Traits and use it in to a single class?

```
<?php
trait message1 {
  public function msg1() {
    echo "OOP is fun! ";
  }
}

trait message2 {
  public function msg2() {
    echo "OOP reduces code duplication!";
  }
}

class Welcome {
  use message1;
```

Name=Divya Sondagar

```
}

class Welcome2 {
  use message1, message2;
}

$obj = new Welcome();
$obj->msg1();
echo "<br>";

$obj2 = new Welcome2();
$obj2->msg1();
$obj2->msg2();
?>
```

Output:

```
OOP is fun!
OOP is fun! OOP reduces code duplication!
```

15.Write PHP Script of Object Iteration?

```
<?php
class MyClass {
    public $var1 = 'value 1';
    public $var2 = 'value 2';
    public $var3 = 'value 3';
    protected $protected = 'protected var';
    private $private = 'private var';

    function iterateVisible() {
        echo "MyClass::iterateVisible:\n";
        foreach ($this as $key => $value) {
            print "$key => $value\n";
        }
    }
}

$class = new MyClass();

// Iterate through visible properties
foreach ($class as $key => $value) {
    print "$key => $value\n";
}

// Call custom method for visible properties
$class->iterateVisible();
?>
```

Name=Divya Sondagar

```
Output:
var1 => value 1
var2 => value 2
var3 => value 3
MyClass::iterateVisible:
var1 => value 1
var2 => value 2
var3 => value 3
protected => protected var
private => private var
```

16.Use of The $this keyword
In PHP, $this is a special keyword that represents the instance of the class where it is used. It allows access to the properties and methods of the current object within the class context. $this is only available within methods of a class, and attempting to use it outside of a class context will result in an error.
Example:-

```
<?php
        Class MyClass
{
        Public $mySample=”Hello !”;
        Public function printSample()
{
        Echo $this->mySample;
}
}
$obj=new MyClass();
$obj->printSample();
?>
```

• Consider the exercise11and add a edit link near delete link e.g. Clicking up on edit button a particular row should be open in
adding an "edit" link next to a "delete" link in a table or list.
1.HTML Structure: Ensure each row has a unique identifier. You might use the id attribute for this purpose.

```
<table>
  <tr id="row1">
    <td>Data 1</td>
    <td>Data 2</td>
    <td><a href="#" class="edit-link" onclick="editRow('row1')">Edit</a></td>
    <td><a href="#" class="delete-link" onclick="deleteRow('row1')">Delete</a></td>
  </tr>
  <!-- More rows... -->
</table>
```

2.JavaScript Functions: Add JavaScript functions to handle the edit and delete actions. These functions will be responsible for performing the necessary actions on the data or UI.

```
<script>
  function editRow(rowId)
```

Name=Divya Sondagar

```
{
    // Add logic to open the edit form or perform edit action
    console.log("Editing row with ID: " + rowId);
  }
  function deleteRow(rowId)
{
    // Add logic to delete the row
    console.log("Deleting row with ID: " + rowId);
  }
</script>
```

3.Styling: You might want to style the edit and delete links to make them visually distinct.

```
.edit-link, .delete-link
{
  margin-right: 10px;
  color: blue; /* Edit link color */
}
.delete-link
{
  //code
}
```

• editing mode
Visual Code Studio
Notepad++
Sublime Text
Net Beans
Atom
PHP Strom
• e.g. on the Particular row there should be filled text box with data and on the option column there should be a confirm button clicking upon it arrow should be updated.
1.Particular Row :
oThis likely refers to a specific entry or record in your data.
2.Text Box With Data :-
oThis is where information related to the particular row is displayed or entered.
3.Option Column :-
oThis could be a column proving different actions or options for each row.
4.Confirm Button :-
oClicking on this button likely triggers a specific action related to the row.
5.Arrow Update :-
oThe arrow updates suggests a visual change, possibly indicating the completion or confirmation of an action.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Interactive Table</title>
    <style>
        table
```

Name=Divya Sondagar

```
{
            border-collapse: collapse;
            width: 100%;
        }

      th, td
{
            border: 1px solid #ddd;
            padding: 8px;
            text-align: left;
        }
        button
{
            padding: 5px 10px;
            cursor: pointer;
        }
    </style>
</head>
<body>
<table>
    <thead>
        <tr>
            <th>Particular</th>
            <th>Text Box with Data</th>
            <th>Options</th>
        </tr>
    </thead>
    <tbody>
        <tr>
            <td>Row 1</td>
            <td><input type="text" id="dataInput1"></td>
            <td>
                <button onclick="confirmAction(1)">Confirm</button>
            </td>
        </tr>
        <tr>
            <td>Row 2</td>
            <td><input type="text" id="dataInput2"></td>
            <td>
                <button onclick="confirmAction(2)">Confirm</button>
            </td>
        </tr>
        <!-- Add more rows as needed -->
    </tbody>
</table>
<script>
    function confirmAction(rowNumber)
{
```

Name=Divya Sondagar

```
        // Perform actions related to the confirmation, e.g., update arrow
        alert('Confirmed for Row ' + rowNumber);
        // You can update the arrow or perform other actions here
    }
</script>

</body>
</html>
```

• Create Hotel Room Booking System User can book room by 3 ways
1.Website Booking:
User Interface:
Homepage: Displays available rooms, promotions, and general information.
Search Page: Allows users to filter rooms based on criteria (e.g., date, number of guests).
Room Details: Provides information about each room, including amenities and prices.
Booking Form: Users can input details such as check-in/out dates, the number of guests, and payment information.
Backend Logic:
Validate user inputs and check room availability.
Calculate total cost based on room rates and selected dates.
Secure payment processing.
Send confirmation email with booking details.
Database:
Store room information (availability, rates, amenities).
Maintain user data (bookings, personal information).
2.Mobile App Booking:
User Interface:
Similar to the website but optimized for mobile devices.
Responsive design for various screen sizes.
Backend Logic:
Sure, I can definitely help with that. What kind of objective facts would you like to discuss?Utilize APIs for seamless communication between the app and server.Sure, I can definitely help with that. What kind of objective facts would you like to discuss?
Implement push notifications for booking updates.
Offline functionality for viewing bookings without an internet connection.
Database:
Synchronize data between the app and the central database.
3.Phone Call Booking:
Interactive Voice Response (IVR) System:
Welcome and guide the user through the booking process.
Voice recognition for user input.
Provide options for room selection, dates, and payment.
Backend Logic:
Convert voice inputs to text and process them.
Check room availability and calculate the total cost.
Generate a booking confirmation.
Database:
Update room availability and store booking details.
• Full day

Name=Divya Sondagar

1. User Registration:
oUsers need to register for an account to book a room.
oCollect basic information like name, email, and password for registration.
oImplement authentication to secure user accounts.
2. Room Selection:
oDisplay a list of available rooms with details (e.g., room type, price, amenities).
oAllow users to select a room for booking.
3. Booking Methods:
oa. Online Booking:
Users can log in, choose the desired room, and make an online payment.
Integrate a payment gateway for secure transactions.
Upon successful payment, confirm the booking and provide a booking ID.
ob. Walk-in Booking:
Users can physically visit the hotel and book a room at the reception.
Receptionist enters user details and assigns a room.
Generate a booking ID and provide it to the user.
oc. Phone Booking:
Users can call the hotel to inquire about room availability.
Hotel staff can check availability, take user details over the phone, and book the room.
Provide a booking confirmation with a unique ID.
4.Reservation Management:
oImplement a system to manage reservations.
oAvoid double bookings by updating room availability in real-time.
oProvide an admin interface to view and manage bookings.
5. User Dashboard:
oUsers can log in to their accounts to view their booking history.
oDisplay upcoming and past bookings with details.
6. Notifications:
oSend email/SMS notifications for successful bookings and reminders.
oNotify users of any changes to their reservations.
7. Cancellation:
oAllow users to cancel bookings with a refund policy.
oImplement cancellation confirmation and refund processing.
8. Reporting:
oGenerate reports for the hotel staff to analyze booking trends, occupancy rates, etc.
9. Security Measures:
oImplement secure coding practices.
oUse HTTPS for secure data transmission.
oRegularly update and patch the system to prevent security vulnerabilities.

• Half day
1. User Registration:
oUsers should be able to create an account or log in if they already have one.
oCollect necessary information such as name, email, contact details, and password.
2. Room Selection:
oDisplay a list of available rooms with details like room type, price, and availability.
oAllow users to choose a room based on their preferences.
oImplement a calendar to show room availability for half-day bookings.

Name=Divya Sondagar

3. Booking Process:
oUsers can initiate a booking by selecting the check-in and check-out dates and times.
oFor half-day bookings, provide options for morning or afternoon check-in/check-out.
oCalculate the total cost based on the selected room and duration.
4. Payment Integration:
oIntegrate a secure payment gateway to handle transactions.
oAccept various payment methods (credit/debit cards, online wallets, etc.).
oConfirm the booking only after successful payment.
5. Booking Confirmation:
oSend a confirmation email or message to the user with booking details.
oDisplay a confirmation page with a summary of the booking.
6. User Dashboard:
oProvide a user dashboard where users can view their booking history.
oAllow users to cancel or modify bookings within a specified timeframe.
7. Admin Panel:
oImplement an admin panel for hotel staff to manage room availability, bookings, and user accounts.
oAllow admins to add new rooms, update prices, and view booking reports.
8. Notifications:
oSend reminders to users before their check-in/check-out time.
oNotify users of successful bookings, cancellations, or any changes to their reservation.
9. Reviews and Feedback:
oAllow users to leave reviews and ratings for their stay.
oDisplay reviews to help future users make informed decisions.
10. Security:
oImplement secure authentication and authorization mechanisms.
oEnsure that sensitive information such as payment details is encrypted.
11. Mobile Responsiveness:
oDesign the system to be mobile-friendly for users who prefer booking on smartphones or tablets.
12. Analytics:
oIntegrate analytics to track user behavior, popular rooms, and booking patterns.

• Custom
Hotel Room Booking System Components:
1.Database:
Create a database to store information about rooms, reservations, and users. Tables could include Rooms, Reservations, and Users.
2.Backend:
Use a backend framework (e.g., Flask, Django, Express.js) to handle server-side logic and interact with the database.
3.User Authentication:
Implement a user authentication system to secure user accounts and booking history.
4.Web Interface:
Create a web interface for users to book rooms. Use HTML, CSS, and JavaScript for the frontend. Ensure a responsive design for different devices.
5.Mobile App:

Name=Divya Sondagar

Develop a mobile app (iOS/Android) using a framework like React Native or Flutter, providing similar functionalities to the web interface.
6.Phone Booking:
Allow users to book rooms over the phone by calling a dedicated reservation hotline. Staff members can use a secure admin panel to manually enter reservations made via phone.
7.Room Availability:
Implement a mechanism to check room availability for specific dates. Update the availability status in real-time.
8.Payment Integration:
Integrate a payment gateway for online bookings. Ensure secure handling of payment information.

• If user select for the full day than user only have selection for the checking checkout date
1.Radio Buttons for Stay Option:
Full Day
Custom Stay
2.Date Picker for Check-In:
Check-In Date: [Date Picker]
3.Date Picker for Check-Out (Conditional):
Check-Out Date: [Date Picker]

• If user select Half day than user have option of date and slot option(like user want to book room for first half – Morning (8AM to 6PM) if user select for second halfit‟s for evening (7PM to Morning 7AM)). Do proper validation like if user can book only available slot. (have touse jQuery -> Ajax, validation, Json passing).
To implement the described functionality with jQuery, Ajax, and JSON passing for booking a room with date and slot options, you can follow these steps:
1. HTML Structure :

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Room Booking</title>
  <!-- Include jQuery -->
  <script src="https://code.jquery.com/jquery-3.6.4.min.js"></script>
</head>
<body>
  <h2>Room Booking</h2>

  <label for="bookingType">Select Booking Type:</label>
  <select id="bookingType">
    <option value="fullDay">Full Day</option>
    <option value="halfDay">Half Day</option>
  </select>
  <div id="dateSlotOptions" style="display:none;">
    <label for="bookingDate">Select Date:</label>
    <input type="date" id="bookingDate">
    <label for="bookingSlot">Select Slot:</label>
```

Name=Divya Sondagar

```
    <select id="bookingSlot">
      <option value="morning">Morning (8AM to 6PM)</option>
      <option value="evening">Evening (7PM to 7AM)</option>
    </select>
  </div>
  <button id="submitBooking">Submit Booking</button>
  <script src="script.js"></script>
</body>
</html>
```

1.JavaScript (script.js):

```
$(document).ready(function()
{
  // Event listener for booking type change
  $("#bookingType").change(function()
{
    if ($(this).val() === "halfDay")
{
      $("#dateSlotOptions").show();
    }
else
{
      $("#dateSlotOptions").hide();
    }
  });
  // Event listener for submit button
  $("#submitBooking").click(function()
{
    // Validate and gather user inputs
    var bookingType = $("#bookingType").val();
    var bookingDate = $("#bookingDate").val();
    var bookingSlot = $("#bookingSlot").val();
    // Validate user inputs
    if (bookingType === "halfDay" && (!bookingDate || !bookingSlot))
{
      alert("Please select date and slot for half-day booking.");
      return;
    }

    // Construct data to send via Ajax
    var bookingData =
{
      type: bookingType,
      date: bookingDate,
      slot: bookingSlot
    };
    // Perform Ajax request
    $.ajax({
```

Name=Divya Sondagar

```
      url: "your_booking_endpoint.php", // Replace with your server-side endpoint
      type: "POST",
      data: JSON.stringify(bookingData),
      contentType: "application/json; charset=utf-8",
      dataType: "json",
      success: function(response)
{
        // Handle success response from server
        alert("Booking successful!");
      },
      error: function(error)
{
        // Handle error response from server
        alert("Booking failed. Please try again.");
      }
    });
  });
});
```

2.Server-side (your_booking_endpoint.php):

```
<?php
header('Content-Type: application/json');

// Retrieve JSON data from the request
$data = json_decode(file_get_contents('php://input'), true);

// Validate and process booking
if ($data['type'] === 'halfDay')
{
    // Perform validation for half-day booking and process accordingly
    // Replace this with your actual validation and booking logic
    $isValidBooking = true; // Replace with your validation logic

    if ($isValidBooking)
{
        // Booking successful
        $response = array('status' => 'success', 'message' => 'Booking successful!');
    }
else
{
        // Booking failed
        $response = array('status' => 'error', 'message' => 'Invalid date or slot for half-day
booking.');
    }
}
else
{
    // Process full-day booking
```

Name=Divya Sondagar

```
    // Replace this with your logic for full-day booking
    $response = array('status' => 'success', 'message' => 'Booking successful!');
}

// Send JSON response back to the client
echo json_encode($response);
?>
```

Jquery
What is jQuery?
jQuery is a fast, small, and feature-rich JavaScript library. It simplifies the process of traversing HTML documents, handling events, creating animations, and managing Ajax interactions. jQuery was created by John Resig and was first released in 2006. It quickly gained popularity due to its ease of use and the ability to perform common tasks with less code compared to raw JavaScript.
Some key features of jQuery include:
1.DOM Manipulation: jQuery simplifies the process of selecting and manipulating HTML elements on the page. It provides a concise syntax for common tasks like changing content, attributes, and styles.

2.Event Handling: jQuery makes it easy to handle various events, such as clicks, keypresses, and mouse movements. Event handling in jQuery is straightforward and cross-browser compatible.

3.Ajax Support: jQuery simplifies asynchronous JavaScript and XML (Ajax) requests, allowing developers to fetch data from a server and update parts of a web page without requiring a full page refresh.

4.Animation Effects: jQuery includes built-in functions for creating animations and transitions, making it easier to add visual effects to web pages.

5. Utilities: jQuery includes a variety of utility functions that streamline common tasks, such as working with arrays and objects, making AJAX requests, and handling browser compatibility issues.

How are JavaScript and jQuery different?

1.Core Purpose:
oJavaScript: It is a general-purpose programming language that can be used for a wide range of tasks, not limited to web development. It is supported by all major web browsers and can be used for both client-side and server-side development.
ojQuery: It is a lightweight, cross-platform JavaScript library designed to simplify the client-side scripting of HTML. jQuery makes it easier to perform common tasks like DOM manipulation, event handling, animation, and AJAX with less code.

2.Syntax:

Name=Divya Sondagar

oJavaScript: It is a programming language with its syntax. It provides a way to interact with the DOM (Document Object Model) and handle events, among other things.
ojQuery: It is essentially a set of functions written in JavaScript. jQuery syntax is often considered simpler and more concise than raw JavaScript, especially for tasks like DOM manipulation.

3.DOM Manipulation:
oJavaScript: It uses native methods to manipulate the DOM. For example, document.getElementById(), document.createElement(), etc.
ojQuery: It provides a simplified and often more cross-browser compatible syntax for DOM manipulation. For instance, $("#elementId") or $(".className") to select elements.

4.Event Handling:
oJavaScript: Event handling in JavaScript can be done using the addEventListener method or by assigning event attributes directly in the HTML.
ojQuery: It simplifies event handling with methods like click(), change(), etc. For example, $("#myButton").click(function() { /* do something */ });.

5.Ajax:
oJavaScript: Native JavaScript provides the XMLHttpRequest object for making AJAX requests.
ojQuery: jQuery simplifies AJAX requests with methods like $.ajax() or shorthand methods like $.get(), $.post().

6.Cross-browser Compatibility:
oJavaScript: Developers need to be aware of and handle cross-browser compatibility issues themselves.
ojQuery: One of the main reasons jQuery gained popularity was its ability to abstract away many cross-browser inconsistencies, providing a more consistent interface.

7.Size and Performance:
oJavaScript: Native JavaScript code can sometimes be more lightweight because it doesn't include the additional abstraction layer that jQuery provides.
ojQuery: While it simplifies coding, it adds an extra layer, and the library itself adds some file size overhead. In modern web development, where bandwidth and performance are crucial, some developers opt to use JavaScript directly for smaller projects.

Which is the starting point of code execution in jQuery?
The starting point of code execution is often inside a document-ready event handler. The document-ready event occurs when the DOM (Document Object Model) has been fully loaded and is ready to be manipulated. This ensures that the jQuery code runs after the HTML document has been parsed.
Syntax:-

```
$(document).ready(function()
{
   // jQuery code goes here
});
```

Name=Divya Sondagar

Document Load Vs Window. Load() jQuery
$(document).ready():
This event occurs when the DOM (Document Object Model) is ready, which means the HTML structure of the page has been fully loaded and parsed.
It is triggered as soon as the DOM is ready, even if other external resources like images, stylesheets, and scripts are still loading.
It is suitable for tasks that don't require waiting for all external resources to be fully loaded.

Example:

```
$(document).ready(function()
{
   // Your code here
});
```

$(window).load():
This event occurs when all assets on the page, including images and other resources, have finished loading.
It waits for the entire page, including external resources, to be fully loaded before triggering.
It is suitable for tasks that depend on the dimensions or content of images, as it ensures that all images have been loaded.
Example:

```
$(window).load(function()
{
   // Your code here
});
```

What is the difference between prop and attr?
1.Attribute (attr):
In HTML, attributes provide additional information about HTML elements and are always included in the opening tag of an element.
Attributes define the properties of an HTML element and are used to configure or modify the behavior of the element.
Examples of attributes include class, id, src, href, alt, etc.
Attributes are part of the HTML document and can be accessed and modified using JavaScript.

Example:

```
<img src="image.jpg" alt="An example image" class="my-image">
```

2.Property (prop):
In the context of JavaScript and the Document Object Model (DOM), properties are values associated with JavaScript objects.
Elements in the DOM are represented as objects, and these objects have properties that can be accessed and modified using JavaScript.

Name=Divya Sondagar

Properties often represent the current state of an element, and they can be dynamic, changing based on user interactions or other events.
Examples of properties include innerHTML, value, style, etc.

Example:

```
var myElement = document.getElementById("exampleElement");
console.log(myElement.innerHTML);     // Accessing the innerHTML property of an element
```

Explain Difference Between JQuery And JavaScript?
1.Core Purpose:
oJavaScript: It is a general-purpose programming language that can be used for a wide range of tasks, not limited to web development. It is supported by all major web browsers and can be used for both client-side and server-side development.
ojQuery: It is a lightweight, cross-platform JavaScript library designed to simplify the client-side scripting of HTML. jQuery makes it easier to perform common tasks like DOM manipulation, event handling, animation, and AJAX with less code.

2.Syntax:
oJavaScript: It is a programming language with its syntax. It provides a way to interact with the DOM (Document Object Model) and handle events, among other things.
ojQuery: It is essentially a set of functions written in JavaScript. jQuery syntax is often considered simpler and more concise than raw JavaScript, especially for tasks like DOM manipulation.

3.DOM Manipulation:
oJavaScript: It uses native methods to manipulate the DOM. For example, document.getElementById(), document.createElement(), etc.
ojQuery: It provides a simplified and often more cross-browser compatible syntax for DOM manipulation. For instance, $("#elementId") or $(".className") to select elements.

4.Event Handling:
oJavaScript: Event handling in JavaScript can be done using the addEventListener method or by assigning event attributes directly in the HTML.
ojQuery: It simplifies event handling with methods like click(), change(), etc. For example, $("#myButton").click(function() { /* do something */ });.

5.Ajax:
oJavaScript: Native JavaScript provides the XMLHttpRequest object for making AJAX requests.
ojQuery: jQuery simplifies AJAX requests with methods like $.ajax() or shorthand methods like $.get(), $.post().

6.Cross-browser Compatibility:
oJavaScript: Developers need to be aware of and handle cross-browser compatibility issues themselves.
ojQuery: One of the main reasons jQuery gained popularity was its ability to abstract away many cross-browser inconsistencies, providing a more consistent interface.

Name=Divya Sondagar

Sure, I can help with that. To remove the paragraph layout from a full file, you can open the document in a word processing program such as Microsoft Word and select all the text. Then, go to the "format" or "layout" menu and choose the option to remove paragraph or line spacing. This will adjust the formatting of the text to remove any paragraph layout. Let me know if you need further assistance.
7.Size and Performance:
oJavaScript: Native JavaScript code can sometimes be more lightweight because it doesn't include the additional abstraction layer that jQuery provides.
ojQuery: While it simplifies coding, it adds an extra layer, and the library itself adds some file size overhead. In modern web development, where bandwidth and performance are crucial, some developers opt to use JavaScript directly for smaller projects.

How We Can Select The Specified <li> Element From The ListOf <li> Elements In <ul>?
To select a specific <li> element from a list of <li> elements within a <ul> (unordered list) using JavaScript, you can use various methods.
Example:
1. Using JavaScript and the DOM:
Assuming you have an unordered list with an id, let's say "myList":

```
<ul id="myList">
  <li>Item 1</li>
  <li>Item 2</li>
  <li>Item 3</li>
</ul>
```

JavaScript code to select the second <li> element:

```
// Get the <ul> element by its id
var myList = document.getElementById("myList");

// Get the second <li> element (index 1, as indices start from 0)
var secondLi = myList.getElementsByTagName("li")[1];

// Now 'secondLi' contains the reference to the second <li> element
console.log(secondLi.textContent);
```

2.Using querySelector:

```
// Use querySelector to directly select the second <li> element
var secondLi = document.querySelector("#myList li:nth-child(2)");

// Now 'secondLi' contains the reference to the second <li> element
console.log(secondLi.textContent);
```

3. Using querySelectorAll:

```
var allLiElements = document.querySelectorAll("#myList li"); // Use querySelectorAll to select all <li> elements and then choose the second one
```

Name=Divya Sondagar

var secondLi = allLiElements[1]; // Get the second <li> element (index 1, as indices start from 0)

console.log(secondLi.textContent); // Now 'secondLi' contains the reference to the second <li> element

In <table> Design Change The Color Of Even <tr> Elements To “green” And   Change The Color Off Odd <tr> Elements To “blue” Color? Give An Example Code?
the even and odd <tr> elements differently. Here's an example code:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <style>
      table
      {
         width: 100%;
         border-collapse: collapse;
      }

      tr:nth-child(even)
{
         background-color: green;
      }

      tr:nth-child(odd)
{
         background-color: blue;
      }

      th, td
{
         border: 1px solid black;
         padding: 8px;
         text-align: left;
      }
   </style>
</head>
<body>

   <table>
      <thead>
         <tr>
            <th>Header 1</th>
            <th>Header 2</th>
            <th>Header 3</th>
        </tr>
    </thead>
    <tbody>
        <tr>
            <td>Row 1, Cell 1</td>
            <td>Row 1, Cell 2</td>
            <td>Row 1, Cell 3</td>
        </tr>
        <tr>
            <td>Row 2, Cell 1</td>
            <td>Row 2, Cell 2</td>
            <td>Row 2, Cell 3</td>
        </tr>
        <tr>
            <td>Row 3, Cell 1</td>
            <td>Row 3, Cell 2</td>
            <td>Row 3, Cell 3</td>
        </tr>
    </tbody>
  </table>
</body>
</html>
```

How We Can Implement Animation Effects In Jquery?
Implementing animation effects in jQuery involves using the built-in animation functions provided by jQuery. These functions allow you to animate the properties of HTML elements, such as their position, size, opacity, and more.

Include jQuery:
Ensure that you include the jQuery library in your HTML file. You can either download it and host it locally or use a CDN (Content Delivery Network). For example:

```
<script src="https://code.jquery.com/jquery-3.6.4.min.js"></script>
```

Select Elements: Use jQuery to select the HTML elements you want to animate. You can do this using the $ function.

```
<div id="myElement">Animate me!</div>
var myElement = $("#myElement");
```

Animate Properties: Use jQuery's animation functions to change the properties of the selected elements. Common properties include height, width, opacity, left, top, etc.

```
myElement.animate(
{
  opacity: 0.5,
  left: '+=50',
```

Name=Divya Sondagar

```
  height: 'toggle'
}, 1000); // 1000 is the duration in milliseconds
```

Chaining Animations: You can chain multiple animations together using the queue option or by simply calling additional animation functions.

```
myElement
  .animate({ left: '+=50' }, 500)
  .animate({ opacity: 0.5 }, 500);
```

Callback Functions: You can use callback functions to execute code after an animation completes.

```
myElement.animate({ opacity: 0.5 }, 1000, function() {
  // Code to execute after the animation is complete
});
```

Easing: jQuery provides easing options to control the speed of animations. You can use built-in options like 'swing' or 'linear', or include a custom easing function.

```
myElement.animate({ left: '+=50' }, { duration: 500, easing: 'linear' });
```

Stop Animation: You can stop an animation in progress using the stop function.

```
myElement.stop();
```

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <script src="https://code.jquery.com/jquery-3.6.4.min.js"></script>
  <title>jQuery Animation Example</title>
</head>
<body>

  <button id="animateButton">Animate</button>
  <div id="animatedDiv">Hello, I'm animated!</div>

  <script>
    $(document).ready(function()
{
      $("#animateButton").click(function()
{
        $("#animatedDiv").animate(
{
          left: '+=100',
          opacity: 0.5
        }, 1000);
      });
    });
  </script>

</body>
</html>
```

Apply jQuery validation using library.
jQuery Validation is a popular library for handling form validation in web applications. To use jQuery Validation, you need to include the jQuery library and the jQuery Validation plugin in your HTML file. Here's a step-by-step guide:

Include jQuery: Make sure to include the jQuery library in your HTML file. You can do this by adding the following line in the <head> section of your HTML:

```
<script src="https://code.jquery.com/jquery-3.6.4.min.js"></script>
```

Include jQuery Validation: Include the jQuery Validation plugin by adding the following line below the jQuery script tag:

```
<script src="https://cdn.jsdelivr.net/jquery.validation/1.19.3/jquery.validate.min.js"></script>
```

Create a Form: Create an HTML form that you want to validate. For example:

```
<form id="myForm">
  <label for="name">Name:</label>
  <input type="text" id="name" name="name" required>

  <label for="email">Email:</label>
  <input type="email" id="email" name="email" required>

  <input type="submit" value="Submit">
</form>
```

Initialize jQuery Validation: Add a script to initialize the jQuery Validation on your form. Place this script after including the jQuery and jQuery Validation scripts:

```
<script>
  $(document).ready(function ()
  {
    $("#myForm").validate();
  });
</script>
```

Create custom dynamic function for require field validator.

Name=Divya Sondagar

To create a custom dynamic function for a required field validator in JavaScript, you can define a function that takes an object as input and checks if the required fields are present. Here's an example of a simple dynamic required field validator function:

```
function createRequiredFieldValidator(requiredFields)
{
  return function (data)
{
    const missingFields = [];

    // Check each required field
    requiredFields.forEach(field =>
{
      if (!data.hasOwnProperty(field) || data[field] === null || data[field] === undefined ||
data[field] === '')
{
        missingFields.push(field);
      }
    });

    // Return the result
    if (missingFields.length === 0)
{
      return { isValid: true };
    }
else
{
      return { isValid: false, missingFields };
    }
  };
}
```

Get state data by country selection (Ajax).
 To retrieve state data based on country selection using Ajax, you'll need to create a web page with HTML, JavaScript, and use a server-side script (like PHP, Node.js, or any server-side technology of your choice) to handle the data fetching. Below is a simple example using HTML, JavaScript, and PHP:

(HTML)Index.php

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Country and State Selection</title>
</head>
<body>
```

Name=Divya Sondagar

```
  <label for="country">Select Country:</label>
  <select id="country" onchange="getStates()">
    <!-- Populate this with your list of countries -->
    <option value="usa">USA</option>
    <option value="canada">Canada</option>
    <!-- Add more options as needed -->
  </select>

  <label for="state">Select State:</label>
  <select id="state"></select>

  <script>
    function getStates()
   {
      var countrySelect = document.getElementById('country');
      var stateSelect = document.getElementById('state');
      var selectedCountry = countrySelect.value;

      // Make an Ajax request to get states based on the selected country
      var xhr = new XMLHttpRequest();
      xhr.onreadystatechange = function()
     {
        if (xhr.readyState == 4 && xhr.status == 200)
       {
          // Parse the JSON response
          var states = JSON.parse(xhr.responseText);

          // Clear existing options
          stateSelect.innerHTML = '';

          // Populate the state dropdown with new options
          for (var i = 0; i < states.length; i++)
         {
            var option = document.createElement('option');
            option.value = states[i];
            option.text = states[i];
            stateSelect.add(option);
          }
        }
      };

      // Replace 'get_states.php' with the path to your server-side script
      xhr.open('GET', 'get_states.php?country=' + selectedCountry, true);
      xhr.send();
    }
  </script>
</body>
</html>
```

Name=Divya Sondagar

PHP (get_states.php):

```
<?php
// Replace this with your own logic to fetch states based on the selected country
if(isset($_GET['country'])) {
  $selectedCountry = $_GET['country'];

  // Example: Provide states based on the selected country
  $states = [];
  if($selectedCountry == 'usa') {
    $states = ['New York', 'California', 'Texas'];
  } elseif($selectedCountry == 'canada') {
    $states = ['Ontario', 'Quebec', 'British Columbia'];
  }

  // Send the states as a JSON response
  echo json_encode($states);
}
?>
```

Image uploading with preview.
=> To implement image uploading with a preview in a web application, you can use HTML, JavaScript, and CSS.

Example:
```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Image Upload with Preview</title>
  <style>
    #preview-container
   {
      display: flex;
      justify-content: center;
      align-items: center;
      height: 200px;
      border: 2px dashed #ccc;
      margin-bottom: 20px;
    }

    #preview-img
   {
      max-width: 100%;
      max-height: 100%;
    }
```

Name=Divya Sondagar

```
  </style>
</head>
<body>

  <input type="file" id="imageInput" accept="image/*" onchange="previewImage(event)">
  <div id="preview-container">
    <img id="preview-img" src="#" alt="Image Preview">
  </div>

  <script>
    function previewImage(event)
{
      const input = event.target;
      const previewImg = document.getElementById('preview-img');
      const previewContainer = document.getElementById('preview-container');

      const file = input.files[0];

      if (file)
    {
        const reader = new FileReader();

        reader.onload = function (e)
      {
          previewImg.src = e.target.result;
        };

        reader.readAsDataURL(file);
        previewContainer.style.display = 'block';
      }
   else
    {
        previewImg.src = '#';
        previewContainer.style.display = 'none';
      }
    }
  </script>

</body>
</html>
```