# AWT – ANSWERS

# UNIT - 1

**Q.1] Explain responsive web Page Design. & What are the benefits of Responsive Web Page design.?**

- responsive web design generally means website that respond to or resize or adjust itself depending upon screen size .
- It automatically adjusts to fit user's screen whether it's desktop, laptop, mobile, tablet, etc. It only uses one layout for web page and it can be done either using CSS and HTML.
- Responsive web design is not a program or a javascript.
- Flexible Layouts: Use grids and layouts that can resize and rearrange based on the screen size.
- Fluid Images: Make images scale appropriately to fit different screen widths without losing quality
- Media Queries: Use CSS rules to apply different styles for different devices, ensuring the design adapts to various screen sizes.
- Consistent User Experience: Ensure the website is easy to use and navigate on any device, providing a seamless experience for all users.

**Benefits of responsive web design:-**

- Responsive web design is best for small to large scale business or company that want to grow their business.
- For growth and success, they need to update their existing website and get it converted into responsive website that not only help one to rank higher in search engine, but also offer better user experience to visitors.
- Responsive web design is best for service-based industries because it includes creative UX and UI design and is made up of images and text.
- Images and graphics are also clear and crisp when viewed on mobile device.
- Responsive web design is also best for companies that have less or tight budget and limited resources because responsive design is budget friendly.
- One has to invest their money and effort only in one rather because of its flexibility.

**Q.2] How do you reverse order of flex items in rows & Column?**

**row-reverse**: This will arrange the flex items in the reverse order of a row, from right to left.

**column-reverse**: This will arrange the flex items in the reverse order of a column, from bottom to top.

By setting the flex-direction property to row-reverse or column-reverse, you can easily reverse the order of flex items in your layout.

```
<div class="container">
  <div class="item">Item 1</div>
  <div class="item">Item 2</div></div>
```

```
.container {
  display: flex;
  flex-direction: row-reverse;
}
//
.container {
  display: flex;
  flex-direction: column-reverse;
}
```

**Q.3] What is flex box & how does it work ?**

- Flexbox is a one-dimensional layout method for arranging items in rows or columns.
- Items flex (expand) to fill additional space or shrink to fit into smaller spaces.

- Flexible Layout: Flexbox can dynamically adjust the size and position of items to best fit the available space, making it ideal for responsive design.
- Alignment Control: Flexbox provides powerful alignment capabilities both along the main axis (horizontal or vertical) and the cross axis, allowing for precise control over the positioning of items.
- Distribution of Space: Flexbox can distribute space between items, including any extra space, ensuring consistent spacing and alignment.
- Direction Control: Items can be laid out in horizontal rows or vertical columns, and their order can be easily reversed.
- The flex container properties are:
  flex-direction
  flex-wrap
  flex-flow
  justify-content
  align-items
  align-content

**How does it work**

```
<!DOCTYPE html>
<html>
<head>
<style>
.flex-container {
  display: flex;
  background-color: DodgerBlue;
}

.flex-container > div {
  background-color: #f1f1f1;
  margin: 10px;
  padding: 20px;
  font-size: 30px;
}
</style>
</head>
<body>

<h1>Create a Flex Container</h1>

<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
</div>

<p>Direct child elements(s) of the flexible container automatically becomes flexible items.</p>

</body>
</html>
```

**Q.4 ] Describe CSS grid Layout & Explain how its work ?**

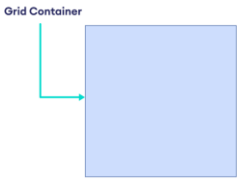
- The CSS Grid is a two-dimensional layout system that allows designers and developers to create complex and responsive layouts.
- Grid layout creates a grid structure of rows and columns and positions elements within the grid's individual cells.
- CSS Grid Layout is a powerful layout system in CSS that allows for the creation of complex and responsive web designs. It uses a two-dimensional grid-based layout system, which can handle both rows and columns, providing more control over the alignment and placement of items within a container.

## How its Works :

- The Grid Layout consists of the following steps:
- 1. Set up the Grid Container
- The first step for a grid-based layout is to define a grid container. This container will hold the grid items.

- The display: grid declaration converts an element as a grid container.

  The following diagram shows a simple grid container




Define a Grid Container: Set the display property of a container element to grid or inline-grid.

Css syntax

```
.container {
  display: grid;
}
```

# UNIT - 2

**Q.1 ] How does the query selector method differ from query selector All ().**

**Q.2 ] Explain Hosting in JavaScript with example ?**

- Hoisting in JavaScript means that variable and function declarations are moved to the top of their scope before the code is executed.

- This allows you to use variables and functions before they are declared in the code.

- Hoisting in JavaScript is a behavior in which a function or a variable can be used before declaration.

For example,

```
// using test before declaring

console.log(test);   // undefined

var test;
```

The above program works and the output will be undefined. The above program behaves as

Since the variable test is only declared and has no value, undefined value is assigned to it.

**Features of Hoisting:**

- In JavaScript, Hoisting is the default behavior of moving all the declarations at the top of the scope before code execution.
- Basically, it gives us an advantage that no matter where functions and variables are declared, they are moved to the top of their scope regardless of whether their scope is global or local.
- It allows us to call functions before even writing them in our code.
- Note: JavaScript only hoists declarations, not initializations.
- JavaScript allocates memory for all variables and functions defined in the program before execution

**Q.3 ] Discuss query selector and query selector all() method in detail.**

**Single Element Selection –**

-querySelector returns the first element that matches the specified CSS selector.

- If no matching element is found, it returns null.

**Usage**

-It is useful when you need to select only one element, typically the first match.

**Syntax**:-

let element = document.querySelector('selector');

**querySelectorAll**

**Multiple Elements Selection –**

-querySelectorAll returns a static NodeList containing all elements that match the specified CSS selector.

- If no matching elements are found, it returns an empty NodeList.

**Usage**:

-It is useful when you need to select multiple elements.

- The returned NodeList can be iterated over using methods like forEach or converted to an array if needed.

**Syntax**:

let elements = document.querySelectorAll('selector');

**Key Differences**

**Return Type:**

**querySelector**: Returns a single DOM element or null.

**querySelectorAll**: Returns a NodeList of DOM elements (which can be empty).

Number of Matches:

**querySelector**: Only the first matching element is returned.

**querySelectorAll**: All matching elements are returned.

Iteration:

**querySelector**: No need for iteration as it returns a single element.

**querySelectorAll**: May require iteration to work with each element in the NodeList.

**Q.4 ] Write JavaScript program for Email validation using regular expression?**

```
function validateEmail(email) {
  // Regular expression for basic email validation
  const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
  // Test the email against the regular expression
  return emailRegex.test(email);
}
// Example usage:
const email1 = "test@example.com";
const email2 = "invalid-email@com";
console.log(validateEmail(email1)); // Output: true
console.log(validateEmail(email2)); // Output: false
```

# UNIT - 3

**Q.1 ] Explain Regular Expression with syntax. & what are the regular expression flags ?**

**Regular expressions -**

- A regular expression is a sequence of characters that forms a search pattern.
- When you search for data in a text, you can use this search pattern to describe what you are searching for.

- A regular expression can be a single character, or a more complicated pattern.
- Regular expressions can be used to perform all types of text search and text replace operations.
- **Syntax**:-

  /pattern/modifiers;
- Regular Expression Modifiers can be used to perform multiline searches which can also be set to case-insensitive matching:

| Expressions | Descriptions |
|---|---|
| g | Find the character globally |
| i | Find a character with case-insensitive matching |
| m | Find multiline matching |

```
<!DOCTYPE html>
<html>
<body>
<h1>JavaScript Regular Expression</h1>
<h2>The g Modifier</h2>

<p>A global search for "is" in a string:</p>

<p id="demo"></p>

<script>
let text = "Is this all there is?";
let pattern = /is/g;
let result = text.match(pattern);

document.getElementById("demo").innerHTML = result;
</script>

</body>
</html>
```

**JavaScript Regular Expression**

**The g Modifier**

A global search for "is" in a string:

is,is

**Q.2 ] Write JavaScript program for Email validation using regular expression**

XML

```
<input type='text' id='txtEmail'/>
<input type='submit' name='submit' onclick='Javascript:checkEmail();'/>
```

Now when the button is clicked then the JavaScript function `SubmitFunction()` will be called. Now write the bellow code in this function.

JavaScript

```
script language="javascript">

function checkEmail() {

    var email = document.getElementById('txtEmail');
    var filter = /^([a-zA-Z0-9_\.\-])+\@(([a-zA-Z0-9\-])+\.)+([a-zA-Z0-9]{2,4})+$/;

    if (!filter.test(email.value)) {
    alert('Please provide a valid email address');
    email.focus;
    return false;
 }
}</script>
```

**Q.3 ] What is frontend & backend? Which language are used to perform frontend & backend ?**

- The difference between Front-End and Back-End is that Front-End refers to how a web page looks, while back-end refers to how it works.

- You can think of Front-End as **client-side** and Back-End as **server-side**.

- The basic languages for Front-End Development are HTML, CSS, and JavaScript.

- ▶ The part of a website that the user interacts with directly is termed the front end.
- ▶ It is also referred to as the 'client side of the application.
- ▶ It includes everything that users experience directly: text colors and styles, images, graphs and tables, buttons, colors, and a navigation menu.

- ▶ HTML, CSS, and JavaScript are the languages used for Front End development.
- ▶ Responsiveness and performance are the two main objectives of the Front End.

-The developer must ensure that the site is responsive i.e.

- it appears correctly on devices of all sizes no part of the website should behave abnormally irrespective of the size of the screen.

**Frontend Developer's Tasks**

-The tasks a frontend developer completes and works on day-to-day will vary.

- They will heavily depend on the company they work for and the role.

- As a frontend dev, you may do a lot of design work.

- That could be creating a style guide to create consistent styles and the brand's identity and overall look and image.

- It includes the typography (fonts) used for all text, the color scheme, the company's logo, and layouts, to name a few.

**Front End Languages**

**HTML**:-

-HTML stands for Hypertext Markup Language.

- It is used to design the front-end portion of web pages using a markup language.

**CSS**:

-Cascading Style Sheets fondly referred to as CSS is a simply designed language intended to simplify the process of making web pages presentable.

**JavaScript**:

JavaScript is a famous scripting language used to create magic on sites to make the site interactive for the user.

**AngularJS**:

-AngularJs is a JavaScript open-source front-end framework that is mainly used to develop single-page web applications(SPAs).

- It is a continuously growing and expanding framework which provides better ways for developing web applications.

**React.js:**

-React is a declarative, efficient, and flexible JavaScript library for building user interfaces.

**Bootstrap:-**

-Bootstrap is a free and open-source tool collection for creating responsive websites and web applications

**jQuery**:

-jQuery is an open-source JavaScript library that simplifies the interactions between an HTML/CSS document, or more precisely the Document Object Model (DOM), and JavaScript.

**Back End Development:-**

-The backend is the server side of the website.

- It stores and arranges data, and also makes sure everything on the client side of the website works fine.

- It is part of the website that you cannot see and interact with.

- It is the portion of software that does not come in direct contact with the users.

- The parts and characteristics developed by backend designers are indirectly accessed by users through a front-end application.

- Activities, like writing APIs, creating libraries, and working with system components without user interfaces or even systems of scientific programming, are also included in the backend.

**Back End Languages**

-There are different tools and technologies backend developers use on a day-to-day basis to implement logic into web applications.

- A key component of backend programming is using a server-side scripting programming language.

Some of the most frequently used ones are -

**PHP**: (Hypertext preprocesser)

-PHP is a server-side scripting language designed specifically for web development.

**C++:**

-It is a general-purpose programming language and is widely used nowadays for competitive programming.

**Java**:

-Java is one of the most popular and widely used programming languages and platforms.

**Python**:

-Python is a programming language that lets you work quickly and integrate systems more efficiently.

**Node.js**:

- Node.js is an open-source and cross-platform runtime environment for executing JavaScript code outside a browser.

- You need to remember that NodeJS is not a framework, and it’s not a programming language.

**Q.4 ] Explain in detail any 4 Meta characters with example.?**

**1] . (Dot)**

Description: The dot meta character matches any single character except newline characters.

```
let regex = /a.b/;

let str1 = "acb";   // Match

let str2 = "a b";   // Match

let str3 = "a\nb";  // No match (because dot doesn't match newlines)

console.log(regex.test(str1)); // Output: true

console.log(regex.test(str2)); // Output: true

console.log(regex.test(str3)); // Output: false
```

**2] ^ (Caret)**

Description: The caret meta character matches the start of a string.

**Example**:

```
let regex = /^Hello/;

let str1 = "Hello world"; // Match (because "Hello" is at the start)

let str2 = "Say Hello";   // No match (because "Hello" is not at the start)

console.log(regex.test(str1)); // Output: true

console.log(regex.test(str2)); // Output: false
```

**3] $ (Dollar Sign)**

Description: The dollar sign meta character matches the end of a string.

**Example**:

```
let regex = /world$/;

let str1 = "Hello world"; // Match (because "world" is at the end)

let str2 = "world Hello"; // No match (because "world" is not at the end)
```

console.log(regex.test(str1)); // Output: true

console.log(regex.test(str2)); // Output: false

**4] \d (Digit)**

Description: The \d meta character matches any digit (0-9).

**Example**:

let regex = /\d/;

let str1 = "There are 3 apples"; // Match (because "3" is a digit)

let str2 = "There are apples";  // No match (because there are no digits)

console.log(regex.test(str1)); // Output: true

console.log(regex.test(str2)); // Output: false

# UNIT - 4

**Q.1 ] Describe JQuery ready function along with syntax.?**

- The $(document).ready(function() { ... }) function ensures your code executes only after the document is fully loaded, preventing errors.

- The `ready` function in jQuery is used to ensure that the DOM (Document Object Model) is fully loaded and ready to be manipulated.

- This is important because trying to manipulate DOM elements before they are loaded can result in errors.

- The `ready` function allows you to execute code once the DOM is ready, but before all external resources like images and stylesheets have fully loaded.

```
<!DOCTYPE html>
<html>
    <head>
        <script src=
"https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js">
        </script>

        <script>
            $(document).ready(function () {
                $("h1").hover(
                    function () {
                        $(this).css(
                            "color",
                            "green"
                        );
                    },
                    function () {
                        $(this).css(
                            "color",
                            "aliceblue"
                        );
                    }
                );
            });
        </script>
    </head>

    <body>
        <h1>DYP-ATU</h1>
    </body>
</html>
```

DYP-ATU

**Q.2 ] Define JQuery. Explain JQuery Features.?**

- Jquery is a lightweight, "write less, do more", javascript library.
- The purpose of jquery is to make it much easier to use javascript on your website.
- Jquery takes a lot of common tasks that require many lines of javascript code to accomplish, and wraps them into methods that you can call with a single line of code.
- Jquery also simplifies a lot of the complicated things from javascript, like ajax calls and dom manipulation.

**The jquery library contains the following features:**

**1] Html/dom manipulation –**

- jQuery allows easy traversal and manipulation of the HTML DOM (Document Object Model).

- You can select DOM elements using CSS-style selectors and then perform actions on them such as changing their content, style, or position.

**2] CSS manipulation –**

- CSS Animations: You can create smooth animations and effects using jQuery.

**3] HTML event methods –**

- jQuery simplifies event handling in JavaScript.

- You can easily attach event handlers to DOM elements and respond to user interactions like clicks, mouse movements, keypresses, etc.

- This simplifies the process of creating interactive web pages.

**4] Effects and animations –**

- jQuery includes methods to create animations and effects on web pages.

- You can animate DOM elements, change their dimensions, opacity, or position over time.

- These animations can enhance the visual appeal of a website and improve user engagement.

**5] Ajax –**

- jQuery provides a set of methods for making asynchronous HTTP requests, commonly known as AJAX (Asynchronous JavaScript and XML) requests.

- This enables web pages to retrieve data from a server without needing to reload the entire page, resulting in faster and more dynamic user experiences.

**6] Utilities –**

- jQuery offers various utility functions to simplify common tasks in JavaScript programming.

- These include functions for working with arrays, objects, strings, and more.

- jQuery's utility functions help in writing concise and readable code.

**7] Cross-browser Compatibility** –

- One of the primary goals of jQuery is to provide a consistent programming interface across different web browsers.

- It handles browser quirks and inconsistencies behind the scenes, allowing developers to write code that works reliably across various browsers without needing to worry about compatibility issues.

**Q.3 ] Explain sliding effect with two methods.?**

**jQuery Effects – Sliding:-**

-The jQuery slide methods slide elements up and down.

- jQuery Sliding Methods With jQuery you can create a sliding effect on elements.

**jQuery has the following slide methods:**

1] slideDown()

2] slideUp()

3] slideToggle()

**1] jQuery slideDown() Method -**

- The jQuery slideDown() method is used to slide down an element.

**Syntax**:

$(selector).slideDown(speed,callback);

-The optional speed parameter specifies the duration of the effect. It can take the following values: "slow", "fast", or milliseconds.

- The optional callback parameter is a function to be executed after the sliding completes.

**Example of slideDown() method -**

```
<!DOCTYPE html>

<html>

<head>

<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.js"></scrip
t>

<script>
```

```
$(document).ready(function(){
        $("#flip").click(function(){
          $("#panel").slideDown("slow");
        });
});
</script>
<style>
#panel, #flip {
        padding: 5px;
        text-align: center;
        background-color: #e5eecc;
        border: solid 1px #c3c3c3;
}
#panel {
        padding: 50px;
        display: none;
}
</style>
</head>
<body>
<div id="flip">Click to slide down panel</div>
<div id="panel">Hello world!</div>
</body>
</html>
```

**2] jQuery slideUp() Method -**

- The jQuery slideUp() method is used to slide up an element.

**Syntax**:

$(selector).slideUp(speed,callback);

-The optional speed parameter specifies the duration of the effect. It can take the following values: "slow", "fast", or milliseconds.

- The optional callback parameter is a function to be executed after the sliding completes.

**Example of slideUp() method -**

```
<!DOCTYPE html>
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.js"></script>
<script>
        $(document).ready(function(){
          $("#flip").click(function(){
            $("#panel").slideUp("slow");
          });
});
</script>
<style>
#panel, #flip {
          padding: 5px;
          text-align: center;
          background-color: #e5eecc;
          border: solid 1px #c3c3c3;
}
```

```
#panel {
        padding: 50px;
}
</style>
</head>
<body>
<div id="flip">Click to slide up panel</div>
<div id="panel">Hello world!</div>
</body>
</html>
```

# UNIT - 5

**Q.1 ] Explain advantages of Node Js ?**

**Advantages –**

-Node.js is an open-source, cross-platform JavaScript runtime environment that executes JavaScript code outside of a web browser. It's a powerful tool used for various types of projects.

**some key aspects:**

JavaScript Runtime: Node.js runs on the V8 JavaScript engine, which is also the core engine behind Google Chrome.

However, unlike the browser context, Node.js executes JavaScript code outside of the browser.

**Single Process Model -**

- A Node.js application operates within a single process, avoiding the need to create a new thread for every request.

-This design choice contributes to Node.js' performance.

**Asynchronous Programming Model –**

-Node.js uses an event-driven, non-blocking (asynchronous) I/O model.

- This allows handling multiple requests simultaneously without blocking the execution of other tasks.

- As a result, Node.js applications are highly responsive and efficient.

**Fast Execution –**

-Node.js leverages the V8 engine, developed by Google, which compiles and executes JavaScript at lightning speeds. This performance advantage makes it suitable for real-time applications and microservices.

**Large and Active Community -**

-Node.js has a vibrant community of developers, libraries, and tools. You'll find extensive resources, tutorials, and support to enhance your learning experience.

**JavaScript Everywhere –**

-Frontend developers familiar with JavaScript can seamlessly transition to writing server-side code using Node.js. You don't need to learn a different language.

**Scalability –**

-Node.js is lightweight and scalable, making it an excellent choice for building real-time applications, RESTful APIs, and microservices.

**Cross-Platform Compatibility –**

-Node.js runs on Windows, Linux, Unix, macOS, and more. This flexibility allows developers to write code once and deploy it anywhere.

**Q.2 ] Explain REPL in Node js.?**

- Node.js is an open source server-side Javascript run-time environment built on Chrome's JavaScript Engine(V8).

- Node.js is used for building fast and scalable applications and is an event driven, non-blocking I/O model.

- REPL (READ, EVAL, PRINT, LOOP) is a computer environment similar to Shell (Unix/Linux) and command prompt.

- Node comes with the REPL environment when it is installed.

- System interacts with the user through outputs of commands/expressions used.

- It is useful in writing and debugging the codes.

**- The work of REPL can be understood from its full form:**

**Read** : It reads the inputs from users and parses it into JavaScript data structure. It is then stored to memory.

**Eval** : The parsed JavaScript data structure is evaluated for the results.

**Print** : The result is printed after the evaluation.

**Loop** : Loops the input command. To come out of NODE REPL, press ctrl+c twice

# UNIT - 6

**Q.1 ] Explain any 5 Features of React Js ?**

**Features of React**

- React is one of the most demanding JavaScript librarys because it is equipped with a ton of features which makes it faster and production-ready.

**1] Component-Based Architecture**

- React provides the feature to break down the UI into smaller, self-contained components. Each component can have its own state and props.

**2] JSX (JavaScript Syntax Extension)**

- JSX is a syntax extension for JavaScript that allows developers to write HTML-like code within their JavaScript files.

- It makes React components more readable and expressive.

**3] Virtual DOM**

- Explanation: React uses a virtual DOM (Document Object Model) to improve performance.

- The virtual DOM is a lightweight copy of the actual DOM.

- When changes occur, React updates the virtual DOM first and then calculates the most efficient way to update the real DOM.

**Benefits:**

**Performance**: Minimizes direct manipulation of the real DOM, which is costly in terms of performance.

**Efficient Updates**: Updates only the parts of the DOM that have changed, rather than re-rendering the entire UI.

**4] Unidirectional Data Flow -**

**Explanation**: React enforces a unidirectional data flow, meaning data flows from parent components to child components. This makes the application easier to debug and understand.

**Benefits**:

Predictability: The flow of data is more predictable, which makes it easier to track the state of the application.

**Debugging**: Simplifies debugging because data flows in a single direction.

**5] Hooks -**

**Explanation** –

-Hooks are functions that let you use state and other React features in functional components.

- Introduced in React 16.8, hooks provide a way to reuse stateful logic without changing the component hierarchy.

**Benefits**:

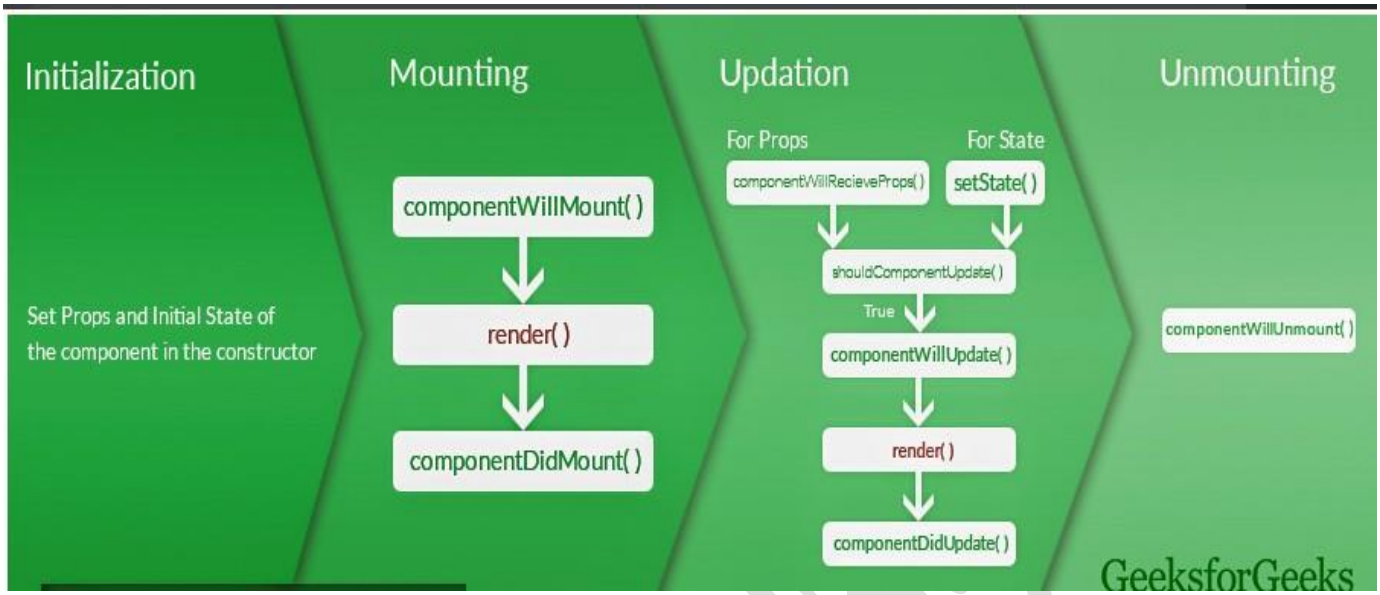
**State Management**: Allows the use of state and lifecycle methods in functional components.

**Reusable Logic**: Enables the reuse of logic between components without relying on higher-order components or render props.

**Q.2 ] Explain REPL in Node js.?**

**Ans is above**

**Q.3 ] Explain Life Cycle of React ?**




**ReactJS Lifecycle**

-Every React Component has a lifecycle of its own, lifecycle of a component can be defined as the series of methods that are invoked in different stages of the component's existence.

- React automatically calls these methods at different points in a component's life cycle.

- Understanding these phases helps manage state, perform side effects, and optimize components effectively.

**1] Initialization -**

- This is the stage where the component is constructed with the given Props and default state.

- This is done in the constructor of a Component Class.

**2] Mounting Phase -**

**- componentWillMount():**

- As the name clearly suggests, this function is invoked right before the component is mounted on the DOM i.e. this function gets invoked once before the render() function is executed for the first time.

**- render():** This method returns the JSX representation of the component. It's called during initial rendering and subsequent updates.

**- componentDidMount():** After the component is inserted into the DOM, this method is invoked. Use it for side effects like data fetching or setting timers.

**3] Updating Phase –**

**setState() Function:**
This is not particularly a Lifecycle function and can be invoked explicitly at any instant. This function is used to update the State of a component.

**- shouldComponentUpdate(nextProps, nextState):** Determines if the component should re-render. Optimize performance by customizing this method.

**- componentWillUpdate():**

This function is invoked before the component is re-rendered i.e. this function gets invoked once before the render() function is executed after the updation of State or Props.

**- render():** Again, the render() method reflects changes in state or props during updates.

- **componentDidUpdate(prevProps, prevState):** Called after the component updates due to new props or state changes. Handle side effects here.

**4] Unmounting Phase –**

-**componentWillUnmount():** Invoked just before the component is removed from the DOM. Clean up resources (e.g., event listeners, timers).