Page:
Date: / /

module :- 1

7 mark

1) Explain ES6 with history and features in details

Ans ES6, also known as ECMAScript 2015, is a significant update to the javascript language standard.

- it was finalized in June 2015 by the ECMA international standards organization.

- ES6 brought numerous enhancements and new features to javascript, aimed at making the language more powerful, expressive, and easier to work with.

History :-

- The ECMAScript specification is the standardized specification of scripting language, which is developed by Brendan Eich of Netscape.

- Prior to ES6, the last major update to javaScript was ES5, which was released in 2009.

- ES6 was in development for several years before its finalization and brought many new feature and improvements to the language.

- The development of ES6 was driven by the need to address shortcomings in javascript, make the language more expressive, and improve developer productivity.

Page:
Date: / /

Features:-

1) Default Parameters:- Default Parameters allow function Parameters to have default values if no argument or 'undefined' is provided

Ex:- function fun(a, b=1){
    return a+b;
}
console.log(fun(5,2));
console.log(fun(3)) // default value =1

2) Template literals:- Template literals allow for easy string interpolation and multiline strings using backticks (`...`).

Ex:- let name = 'Rahul';
let message = `Hello, ${name}!`;

3) Arrow function:- Arrow function (=>) provide a concise syntax for writing anonymous function, enhancing code readability and reducing boilerplate.

Ex:-
const add = (a,b) => a+b;

4) let and const:- The const keyword is used to declare constant variables whose values can't be changed.

- The let variables are mutable i.e. their values can be changed.

- ES6 introduced block-scoped variable using 'let' and 'const' keywords, which provided a more predictable variable scope behavior compared to 'var'.

5) Classes :- ES6 introduced classes in JavaScript. classes in JavaScript can be used to create new objects with the help of a constructor, each class can only have one constructor inside it.

Eg :-
```
class Person {
    constructor(name) {
        this.name = name;
    }
}
const P = new Person('Rahul');
console.log(P.name);
```

6) Rest and Spread operators :-

- The spread operator ('...') spreads the elements of an iterable (e.g., an array) into individual elements

- The rest operator ('...') collects multiple function arguments into a single array parameter.

7) Promises :- Promises provide a cleaner and more robust way to work with asynchronous code compared to callback functions.

2) define given concepts of advanced Javascript which are let, const, arrow function, ternary operator, destructuring, spread and modules.

Ans 1) let :- 'let' is keyword introduced in ES6 for declaring variables.

- Variables declared with 'let' are block-scoped, meaning they are only accessible within the block in which the are defined.
- The let variables are mutable i.e. their values can be changed. it works similar to the var keyword with some key differences like scoping which makes it a better option when compared to var.

Ex:-

```
let x = 10;
if (true) {
    let y = 20;
    console.log(x);
}
console.log(y); // Reference Error: y is not defined
```

2) const :- constants are block-scoped and cannot be reassigned after declaration.

- However, if a constant holds a reference to an object or array, the properties or elements of that object or array can still be mutated.

Ex:-

```
const PI = 3.14
PI = 3.14159; // TypeError
const name = "Rahul"
console.log(name); // output : Rahul.
```

Page:
Date: / /

3) arrow function :- Arrow functions provide a concise syntax for writing anonymous functions.
- they are denoted by the '=>' syntax.
- Arrow function are useful for writing shorter and cleaner code, especially for callback functions.
- you don't need the function keyword, the return keyword, and the curly brackets.

Ex:-
```
let age = 5;
let welcome = (age < 18) ?
    () => console.log ('Baby') :
    () => console.log ('Adult');

welcome();
```

4) ternary operator :- The ternary operator, also known as the conditional operator, is a JavaScript operator that provides a compact way to write simple conditional statements.
- it consists of a condition followed by a question mark ('?'), then an expression to execute if the condition is true, followed by a colon (':'), and finally an expression to execute if the condition is falsy.

Ex:-
```
const x = 10;

const message = x > 5 ? 'greater than 5' :
    'less than or equal to 5';
```

Page:
Date: / /

5) Destructuring :- Destructuring allows you to extract values from arrays or objects and assign them to variables in a more concise way.

- Destructing in Javascript basically means the breaking down of a complex structure into simpler parts.

Ex:- const person = { name : 'Rahul', age = 21 };
cons { name, age } = person;
console.log (name, age);

6) Spread :- The spread operator ('...') allows an iterable (like an array or string) to be expanded into individual elements.

- it is commonly used for array manipulation, function arguments, and object literals.

Ex:- const arr1 = [1, 2, 3];
const arr2 = [4, 5, 6];
const mergeArray = [...arr1, ...arr2];

7) modules :- modules are a way to organize and encapsulate javascript code into reusable units of functionality.

- Javascript modules allow you to break up your code into separate files.
- modules can export functions, objects, or other values for use in other modules and can also import functionality from other modules.

Eg:- // module.js

export const add = (a, b) => a+b;

// main.js

import {add} from './module.js';

console.log(add(2,3));

5 marks:-

1) differentiate types of variable in js with a truth table.

Ans

1) var :- var is function-scoped. It means that variables declared with 'var' are accessible within the function they are declared in, regardless of block scope.

- variables declared with 'var' can be redeclared and reassigned within the same scope.

Syntax:- var variableName = value;

Eg:- var x = 10;

function example() {

var x = 20;

console.log(x); // output 20

}

example();

console.log(x); // output: 10

Page:
Date: / /

2) let :- 'let' is block-scoped. It means that variables declared with 'let' are accessible only within the block they are declared in (e.g., within curly braces '{}').

- variables declared with 'let' can be reassigned but cannot be redeclared within the same scope.

Syntax:- let VariableName = value;

Ex:- let x = 10;
if (true) {
let x = 20;
console.log(x); // output 20
}
console.log(x); // output : 10

3) const :- const is also block-scoped like 'let'.

- Variables declared with 'const' cannot be reassigned once they are initialized. However, for objects and arrays, the properties or elements of the object or array can be mutated.

- variables declared with 'const' cannot be redeclared within the same scope.

Syntax:- const variableName = value;

Ex:- const PI = 3.14;
PI = 3.14159 // cannot be reassigned
const person = { name: 'Rahul' };
person.age = 20; // properties of object can be modified is valid

Page:
Date: / /

Truth Table :-

Behavior	Var	let	const
Hoisting	Hoisted	Not Hoisted	Not Hoisted
Block scope	No	yes	yes
can be reassigned	yes	yes	no
can be re-declared	yes	no	no
function scope	yes	yes	yes

2) Explain template literal with ternary operator. give an example.

Ans Template literals are a feature in Javascript that allow for easier string interpolation and multiline strings by using backticks (`) instead of single or double quotes.

- They also support expression interpolation, including the use of ternary operators.
- The ternary operator (`? :`) is a concise way to write conditional statements in Javascript
- it evaluates a condition and returns one value if the condition is true, and another value if the condition is false.
- Combining template literals with the ternary operator allows ye to dynamically construct strings based on conditions.

Page:
Date: / /

- A ternary operator is a concise way to write conditional statements. It has the following syntax:-

'condition ? expressionIfTrue : expressionIfFalse'.

Ex:-

```
const temperature = 25;
const weather = temperature > 20 ?
                'warm' : 'cold';

const message = 'The weather today is
${weather} with a temperature of ${temperature}
degrees celsius.';

console.log(message)
```

Page:
Date: / /

3) Explain async await as asynchronous javascript with example.

Ans Async/await is modern way of handling asynchronous in Javascript. It provides a more readable and structured approach to writing asynchronous code compared to using callbacks or promises.

Async function:
Async simply allows us to write promises based code as if it was synchronous and it checks that we are not breacking the execution thread.
- Async functions will always return a value.

Await function:- Await function is used to wait for the promise. It could be used within the async block only.
- It makes the code wait until the promise returns a result.

- The async keyword transforms a regular javascript function into an asynchronous function.

- The await keyword is used inside an async function to pause its execution and wait for a promise to resolve before continuing.

Page:
Date: / /

Ex:-
```
function fetchData () {
    return new Promise ((resolve, reject) => {
        setTimeout (() => {
            resolve ('Data fetched successfully!');
        }, 2000);
    });
}

async function ~~fetch~~ DataAsync () {
    try {
        console.log ('fetching data...');
        const data = await fetchData ();
        console.log ('Data received:', data);
    } catch (error) {
        console.log ('Error', error);
    }
}
DataAsync ();
console.log ('Async operation started...');
```

Page:
Date: / /

4] give an example of real time entity problem using oop is class and object.

Ans example of a Bank account using oop.

Class :- BankAccount

- attributes :-
  - accountNumber : ac_Number
  - accountHolder : ac_Holder
  - balance.

- method :-
  - deposit()
  - withdraw()
  - checkBalance()

Example :-

```
Class BankAccount {
    constructor (ac_no, ac_Holder, balance=0){
        this.ac_no = ac_no;
        this.ac_Holder = ac_Holder;
        this.balance = balance;
    }

    deposit (amount) {
        this.balance += amount;
        console.log(`Deposit of ${amount}
        successful. current balance : ${this.balance}`
        );
    }
```

```
withdraw(amount) {
  if(amount > this.balance){
    console.log("Insufficient funds in account");
  }else
  {
    this.balance -= amount;
    console.log(`withdrawal of ${amount}
    successful. current balance : ${this.balance}`
    );
  }
}
checkBalance(){
    console.log(`current balance : ${this.
        balance}`);
}
}
const account1 = new BankAccount(12345, 'Rahul');
account1.deposit(1000000);
account1.withdraw(500000);
account1.getBalance();
```

Page:
Date: / /

4 mark module:-1

3) give appropriate example of map, filter, for of and for in with concept.

Ans 1) map() :- The 'map()' function in JavaScript is used to apply a function to each element of an array and creates a new array with the results of calling that function on each element.

- it doesn't mutate the original array.

Syntax:-

```
map((element, index, array) => { /*...*/ })
```

Ex:- const number = [1,2,3,4,5];

```
const squrno = number.map(num => num*num)
console.log(squrno);
```

2) filter :- The 'filter()' function in JavaScript is used to filter out elements from an array based on a specified condition.

- It creates a new array containing only the elements that pass the condition provided by a callback function.

Syntax:-

```
array.filter(callback(element, index, arr), this value)
```

Page:
Date: / /

Ex:- const ages = [32, 30, 16, 40];

function check(age) {
return age >= 18;
}
const result = ages.filter(check);
console.log(result);

3) for... of :- the for... of loop in JavaScript is used to iterate over iterable objects like arrays, strings, maps, sets, etc.
It provides a concise and readable way to loop through the elements of an iterable without explicitly tracking the index.

Syntax:- for (variable of iterable) {
// code
}

Ex:- const fruits = ['apple', 'banana', 'kiwi'];

for (const fruit of fruits) {
console.log(fruit);
}

4) for... in :- for... in loop in JavaScript is used to iterate over the properties of an object.

Page:
Date: / /

- It iterates over all enumerable properties of an object, including inherited enumerable properties from its prototype chain.

Syntax:

```
for (variable in object){
    // code
}
```

Ex:-

```
const Person = {
    name : 'Rahul',
    age : '21',
    city : 'Jamnagar'
};
for (let key in Person)
{
  console.log(`${key} : ${Person[key]}`);
}
```

Page:
Date: / /

module:-2

1) what is react js. explain its features in details.

Ans

Recat js is an open-source javascript library developed by facebook for building user interfaces, particularly for single-page applications and complex web interfaces.

- It's one of the most popular libraries for frontend development due to its simplicity, flexibility, and high performance.
- React follows a component-based architecture, where UIs are composed of reusable components that manage their own state, making it easier to build complex UIs.

Features:-

1) JSX:- JSX stands for javascript xml. It is a javascript syntax extension. Its an xml or HTML like syntax used by Reacts.
- This syntax is processed into javascript calls of react framework.
- JSX makes it easier to describe UI components and their structure in a more declarative and intuitive way.

2) Components:- ReactJs is all about components. ReactJs application is made up of multiple components, and each component has its own logic and controls.

- These components can be reusable which help you to maintain the code when working on larger scale projects.

3) one-way Data Binding :- ReactJS is designed in such a manner that follows unidirectional data flow or one-way data binding. The benefits of one-way data binding give you better control throughout the application.

- If the data flow is in another direction, then it requires additional features.

4) virtual Dom :- A virtual Dom object is a representation of the original Dom object. It works like a one-way data binding.

- when the state of a component changes, React first updates the virtual Dom, then compares it with the previous version to identify the minimal set of Dom operations needed to update the actual Dom.

- The this minimize Dom manipulation and improves performance.

5) simplicity :- ReactJS uses JSX file which makes the application simple and to code as well as understand.

- we know that ReactJS is a component-based approach which makes the code reusable as your need.

- this makes it simple to use and learn.

Page:
Date: / /

6) Performance :- As we discussed earlier, React uses virtual Dom and updates only the modified parts. So, this makes the Dom to run faster.

Dom executes in memory so we can create separate components which makes the Dom run faster.

7) Conditional Statements :-

JSX allows us to write conditional statements. The data in the browser is displayed according to the conditions provided inside the JSX.

Page:
Date: / /

2) How to create react application and explain project structure in details.

Ans How to create react application:-

1) install Node.js and NPM:- Ensure you have node.js installed on your system as it comes with npm

2) install create-react-app:- open your terminal or command prompt and run the following command to install.
- npm install -g create-react-app

3) Create a new React app:- you can create a new React app by running the following command:-
- npx create-react-app my-app

4) Navigate to the Project directory:- once the app is created, navigate to the Project directory.
- cd my-app

* Project Structure :-

```
my-app/
├── node_modules/
├── public/
│   ├── favicon.ico
│   ├── index.html
│   └── ...
```

Page:
Date: / /

```
├── src/
│   ├── components/
│   │   ├── App.js
│   │   └── ...
│   ├── App.css
│   ├── App.test.js
│   ├── index.js
│   ├── index.css
│   └── ...
├── .gitignore
├── package.json
├── package-lock.json
└── ...
```

- node_modules/ :- This directory contains all the dependencies installed by npm or yarn. You don't need to modify anything inside this directory.

- Public/ :- This directory contains static assets such as HTML files, images, and the favicon. The 'index.HTML' file is the main HTML file that serves as the entry point for your React application.
  - it contains a where React will render your components.

- src/ :- This directory contains the source code of your react application.

- components/ :- This directory contains react components. App.js is the main component

Page:
Date: / /

that serves as the root of your application.

- App.css : css styles specific to the 'App' component.
- App.test.js : Test file for the 'App' component.
- index.css :- contains global css styles that are applied to the entire application.
- index.js :- The entry point of the application, where React is initialized and the main 'App' component is rendered.
- .gitignore : This file specifies which files and directories should be ignored by git.
- Package.json :- contains project configuration, dependencies, and scripts defined for running various tasks.

Page:
Date: / /

5 marks:

1) describe about creating react application with dependencies as environment with set of commans:

Ans

Step 1:- install Node.js and NPM:-

if you haven't already, you need to install Node.js which includes NPM.

Step 2:- create a new react application:-

open your terminal or command prompt and run the following command to create a new react application using create react app:-

- npx create-react-app my-app

Step 3:- After the project is created, navigate to the project directory using the 'cd' command:

- cd my-app

Step 4:- install additional dependencies:-

If you need to install additional dependencies, such as react routes for routing or Axios for making HTTP requests, you can do so using NPM.

Ex

NPM install react-routes-dom axios

Page:
Date: 1 · 1

Step 5:- Start the development server!

Once you have installed the neccessary dependencies, you can start the development server to see your react application in action.

Ex:- npm start

6) Build for production:-

When you're ready to deploy your React application to production, you can build it using the following command:-

- npm run build

- That's it you've now created a React application with dependencies and set up your development environment.

Page:
Date: / /

2) Explain Package.json file with its attributes and Purpose.

Ans Package.json is a metadata file in a Node.js Project that describes the project's dependencies, scripts, configuration, and other details.

- It typically contains information about the Project such as its name, version, author, and license.
- The package.json file is typically located in the root directory of a Node.js Project and is automatically generated when you run npm init command to initialize a new Project.

* attributes and Purpose:-

1) name:- specifies the name of the Package. It should be unique within the npm registry. For a Project, it typically matches the Project's directory name.

2) version:- specifies the version of the Package. e.x('1.0.0').

3) description:- Provides a brief description of the Package. It helps users understand the Purpose of functionality of the Project.

~~These are three dependencies, react, react-dom and react-scripts in our JS~~

4) main :- specifies the entry point of the package. For most Node.js projects, this is typically 'index.js' or 'app.js'.

5) script :- Defines various scripts to be run with npm. Common scripts include 'start' for starting the application, 'test' for running tests, and 'build' for building the project for production.
- Users can also define custom scripts for specific tasks.

6) dependencies :- lists the dependencies required for the project to run in production.
- These are packages that are necessary for the application to function properly.

7) keywords :- Provides an array of keywords that describe the package. These keywords help users discover the package when searching on npm.

8) author :- This attribute specifies the author of the package. It can be a string or an object.

Page:
Date: / /

3) what is npm. and how to use in React Js list out different commands with purpose.

Ans npm, which stands for Node package manager, is a package manager for Node.js packages or modules.

- It is primarily used for installing, sharing, and managing dependencies in JavaScript projects.

- in the context of React.js, npm is often used to manage project dependencies and scripts.

* npm commands :-

1) npm init :- initializes a new 'package.json' file for your project.

2) npm install :- installs all dependencies listed in the 'package.json' file.

- it reads the 'dependencies' and devDependencies sections and installs all the required packages locally in the 'node_modules' directory.

3) npm install <package-name> :- installs a specific package and adds it to the 'dependencies' section of the package.json file.

Ex :- npm install react.

Page:
Date: / /

4) npm start :- Runs the script specified in the start field of the scripts section in package.json.
- in many React.js projects, this command is used to start the development server.

5) npm run build :- This command is used to build the production-ready version of react-js application.

6) npm update :- updates all the packages listed to the latest version according to the version range specified in the 'package.json' file.
- It updates the 'package.json' and package-lock.json files accordingly.

7) npm test :- This command runs tests for your React.js application.
- This script typically runs unit tests, integration tests, or end-to-end tests depending on your testing setup.

8) npm audit :- checks for known vulnerabilities in the dependencies of your project and provides recommendations for resolving them.

9) npm Publish :- This command is used to publish a package to the npm registry.

Page:
Date: / /

4) describe history of react js and application scope of it

Ans React is the most powerful javascript library, which is used for building user interfaces.

- it was developed by facebook and is often used for building single-page application and mobile application.

- React was ~~reused~~ released first in 2013, and since then, it has become one of the most popular javascript libraries for building user interfaces.

- it was developed by jordan walke, a software engineer at facebook, and was initially released as an open-source project on github.

- React.js quickly gained popularity due to its component-based architecture, virtual Dom, and efficient rendering.

* application scope:-

1) user interfaces:- React.js is primarily used for building dynamic and interactive user interfaces for web applications.

2) single page applications (SPAs):- React.js is well-suited for developing SPAs where content is dynamically routed and updated without requiring pages reloads.

Page:
Date: / /

3) Component-Based Architecture!- React.js promotes a modular approach to ui development, making it ideal for large-scale applications where components can be reused and maintained easily.

4) Cross-Platform Development!- With frameworks like React js, react native extends its scope to building mobile applications for ios and Android platforms.

5) Real-Time Data Applications!- React.js is often used in conjunction with other technologies like Redux or graphql to build real-time data applications such as dashboards or collaborative tools.