UNIT-1

INTRODUCTION TO FULL STACK WEB DEVELOPMENT

1.1 INTRODUCTION TO FULL STACK DEVELOPMENT

Full stack development: It refers to the development of both front end (client side) and back end(server side) portions of web application.

Full stack web Developers: Full stack web developers have the ability to design complete web applications and websites. They work on the frontend, backend, database and debugging of web applications or websites.

1.2 OVERVIEW OF FULL STACK AND PRE-REQUISITES

Full stack development: It refers to the development of both front-end(client side) and back end(server side) portions of web application.

Full stack web Developers: Full stack web developers have the ability to design complete web applications and websites. They work on the frontend, backend, database and debugging of web applications or websites.

1.2.1 What is Frontend Web Development?

Front end web development is the process of developing the client side of the web application wherein the focus is on the users. The full stack developer with users in mind undertakes this process to develop a website that is easy to use and interact with. The developer in this process focuses on the visual elements of the application. It is also the responsibility of the developer in front end development to ensure that the website or application runs in a smooth fashion. Front end development focuses on elements like:

1. Images
2. Graphics
3. Animations
4. Layouts
5. Content organization
6. Navigation

1.2.2 What is Backend Web Development?

Back-end web development refers to the process of developing the server side of the website, i.e., the side that users cannot see or interact with. The main aim of back-end development is to make the website interactive and easy to use. During the back-end development process, developers lay emphasis on:

1. Building the code

2. Database management
3. Debugging or troubleshooting the website or application
4. Application programming interfaces
5. Architecture

Therefore, front end and back-end development are distinct from each other in terms of the sides of the website application they focus on. However, the two work together to create a dynamic and attractive website that provides users with an immersive experience when browsing

1.2.3 Front-end Languages and Frameworks

HTML and CSS

HTML short of Hypertext markup language governs the appearance of text/images on the screen. It formats the text as headings, paragraphs, pop-ups and so on. It typically consists of tags that give a different meaning to the text in a document structure.

CSS (Cascading Style Sheets) is a style sheet that allows to alter and style different web components for font, size, spacing, etc. It makes the content on the webpage more decorative.

JavaScript

It is a powerful client-side programming language mainly used for enhancing user interaction with the application. It offers several libraries and frameworks like React, Angular, Vue, and JQuery. A full stack developer needs to be proficient in one or more of these tools.

1.2.4 Backend Technologies and Frameworks

JavaScript and its environments like Node JS and Express JS Node is an open-source, cross-platform runtime environment that allows developers to create server-side tools and JavaScript applications. Knowledge of the basic command line like npm (Node Package Manager) is essential.

Java, in particular, was built from the ground up to be run on the server-side. Popular Java frameworks include Spring and Java Server faces.

Python is the most popularly used language. It is flexible and easy to use. We use Python frameworks like Django and Flask for backend development. Django is a high-level Python web framework that enables rapid development of websites. It is free and open-source, has a huge and active community of users.

C# language is for many the most preferred architecture when it comes to backend programming in Windows environments.

Other languages include PHP, Perl, and Ruby.

Database Management Systems

A full-stack developer writes code that is used to perform Relational mapping to fetch data from the database. Some popularly used DBMS are MySQL, SQL SERVER and, and Oracle Database. PostgreSQL,MongoDB

Version Control

Version control systems help manage the project files and keep track of the entire history of the user's things. Popular ones are Git,GitHub and Subversion.

Git helps manage the project files. It keeps track of the entire history of things that the user is working on. GitHub is a web-based service for version control using Git. The user can look at other people's code, identify issues or errors, and even propose changes.

Web Hosting Platforms

Web hosting platforms allow you to deploy your product on a cloud service provider so we can access it from the WWW. Popular ones are Amazon web services, Google cloud platform, Heroku, and Microsoft Azure.

Soft Skills

A full-stack developer needs to possess some basic soft skills as well.

Creativity: The developer must understand the project's objectives and create a product that engages the target audience.

Strategic planning: Planning is a critical phase in any development life-cycle. The developer should possess the aptitude to plan the design, development, and implementation phases strategically. This helps streamline the process.

Analytical skills: As the name suggests, these skills help analyze information accurately and make logical, data-driven decisions that help in maximizing efficiency. Now, these are skills that one develops when they're exposed to it. But a good sense of understanding is required.

Problem-solving skills are again crucial to predict any errors and develop a bug-free application. From a business perspective, problem-solving skills play a significant role in ensuring the smooth conduction of tasks.

Time management skills: The developer must meet the deadlines while ensuring proper execution of his tasks.

1.3 Introduction of Front-End Development

The front-end development, also known as client-side development is the execution of HTML, CSS, and JavaScript for a website or web application to make the data user friendly.

There are a few challenges associated with front end development and one of them is the constant changes in the tool and technique that is used to create the front end of the website. The biggest challenge for the front end developer is to make sure the site works smoothly across different browsers, different operating systems and different devices.

Roles and responsibilities of a Front End Development

Though different organizations assign different tasks to the developers, there are few common tasks which everyone needs to perform.

- Using HTML, CSS and JavaScript to give life to the data.
- Making the website attractive and user friendly.
- Enhancing the user experience.
- Designing, developing and maintaining the user interface.
- Innovating tools that improve user interaction and audience retention.
- Fixing the bugs in application and testing for usability.
- Making sure that application works smoothly on different devices and operating systems.

Web technologies that are involved in Front End Development

Front end developers use numerous technologies to convert backend code into a user-friendly interface. Few of the web technologies that are used by front end developers are; Hypertext Markup Language (HTML), Cascading Style Sheets (CSS) and JavaScript.

HTML: It is considered as the building block of a website. For a browser to display the content correctly, HTML is used as a programming language to describe and mark the desired content.

CSS: It is way more than a programming language, it's like a set of instructions that controls the style and structure of a webpage. CSS helps the front-end developers to manage a website or web application's formatting, presentation and layout.

The major role of CSS is to dictate how users will see the content.

JavaScript: JavaScript came into action after 1995, before that developers used to design the website using HTML and CSS. JavaScript has helped the developers to make the website interactive and attractive. JavaScript helps to make changes in the content of the website based on the actions of the user.

DOM: DOM stands for Document Object Model. It is a cross-platform and language-independent convention for representing and interfacing with objects in HTML, XHTML documents.

CSS preprocessors: To add functionality in the CSS code and to make it more interactive and scalable, front-end developers use CSS preprocessors. Its task is to transit in the well-formatted CSS that work on different browsers. The most in-demand CSS preprocessors are LESS and SASS.

1.4 Basics of javascript

1.4.1 What is Javascript?

JavaScript is a scripting or programming language that allows you to implement complex features on web pages — every time a web page does more than just sit there and display static information for you to look at — displaying timely content updates, interactive maps, animated 2D/3D graphics, scrolling video jukeboxes, etc. — you can bet that JavaScript is probably involved. It is the third layer of the layer cake of standard web technologies, two of which (HTML and CSS) we have covered in much more detail in other parts of the Learning Area.

HTML is the markup language that we use to structure and give meaning to our web content, for example defining paragraphs, headings, and data tables, or embedding images and videos in the page.

CSS is a language of style rules that we use to apply styling to our HTML content, for example setting background colors and fonts, and laying out our content in multiple columns.

JavaScript is a scripting language that enables you to create dynamically updating content, control multimedia, animate images, and pretty much everything else. (Okay, not everything, but it is amazing what you can achieve with a few lines of JavaScript code.)

1.4.2 Javascript Datatype

JavaScript provides different **data types** to hold different types of values. There are two types of data types in JavaScript.

1. Primitive data type
2. Non-primitive (reference) data type

JavaScript is a **dynamic type language**; means you don't need to specify type of the variable because it is dynamically used by JavaScript engine.

- **Var**

var here to specify the data type. It can hold any type of values such as numbers, strings etc.

The var keyword was used in all JavaScript code from **1995** to **2015**

Example:

```
var a=40;
var b="abc";
```

- **Let**

The let keyword was introduced in ES6 (2015)

Variables defined with let cannot be Redeclared

Variables defined with let must be Declared before use

Variables defined with let have Block Scope

Example:

let x = 1;

let x = "John Doe";

- **Const**

The const keyword was introduced in ES6 (2015)

Variables defined with const cannot be **Redeclared**

Variables defined with const cannot be **Reassigned**

Variables defined with const have **Block Scope**

Example:

const PI = 3.141592653589793;

JavaScript primitive data types

Data Type	Description
String	represents sequence of characters e.g. "hello"
Number	represents numeric values e.g. 100
Boolean	represents boolean value either false or true
Undefined	represents undefined value
Null	represents null i.e. no value at all

JavaScript non-primitive data types

Data Type	Description
Object	represents instance through which we can access members
Array	represents group of similar values
RegExp	represents regular expression

1.4.3 Conditional Loop

1.For loop

```
for (expression 1; expression 2; expression 3) {
  // code block to be executed
}
```

Expression 1 is executed (one time) before the execution of the code block.

Expression 2 defines the condition for executing the code block.

Expression 3 is executed (every time) after the code block has been executed.

Example:

```
<script>
let text = "";
for (let i = 0; i < 5; i++) {
  text += "The number is " + i + "<br>";
}
document.getElementById("demo").innerHTML = text;
</script>
```

output:

The number is 0
The number is 1
The number is 2
The number is 3
The number is 4

2. While loop

```
while (condition){
//code block to be executed
}
```

Example

```
<script>
let text = "";
let i = 0;
while (i < 5) {
```

```
  text += "<br>The number is " + i;
  i++;
}
document.getElementById("demo").innerHTML = text;
</script>
```

output:

```
The number is 0
The number is 1
The number is 2
The number is 3
The number is 4
```

3.Do While loop

The do while loop is a variant of the while loop. This loop will execute the code block once, before checking if the condition is true, then it will repeat the loop as long as the condition is true.

Syntax

```
do {
  // code block to be executed
}
while (condition);
```

Example

```
<script>
let text = ""
let i = 0;
do {
  text += "<br>The number is " + i;
  i++;
}
while (i < 10);
document.getElementById("demo").innerHTML = text;
</script>
```

output:

```
The number is 0
The number is 1
The number is 2
```

The number is 3
The number is 4
The number is 5
The number is 6
The number is 7
The number is 8
The number is 9

4.If Else Statement

- Use if to specify a block of code to be executed, if a specified condition is true
- Use else to specify a block of code to be executed, if the same condition is false
- Use else if to specify a new condition to test, if the first condition is false
- Use switch to specify many alternative blocks of code to be executed

```
<script>
const hour = new Date().getHours();
let greeting;
if (hour < 18) {
  greeting = "Good day";
} else {
  greeting = "Good evening";
}
document.getElementById("demo").innerHTML = greeting;
</script>
```

Output

Good day

5) Else If Statement

Use the else if statement to specify a new condition if the first condition is false.

Syntax

```
if (condition1) {
  //  block of code to be executed if condition1 is true
} else if (condition2) {
  //  block of code to be executed if the condition1 is false and condition2 is true
} else {
  //  block of code to be executed if the condition1 is false and condition2 is false
}
```

Example

```
<script>
```

```
const time = new Date().getHours();
let greeting;
if (time < 10) {
  greeting = "Good morning";
} else if (time < 20) {
  greeting = "Good day";
} else {
  greeting = "Good evening";
}
document.getElementById("demo").innerHTML = greeting;
</script>
```

Output

Good day

6.Switch

Use the switch statement to select one of many code blocks to be executed.

Syntax

```
switch(expression) {
  case x:
    // code block
    break;
  case y:
    // code block
    break;
  default:
    // code block
}
```

This is how it works:

The switch expression is evaluated once.

The value of the expression is compared with the values of each case.

If there is a match, the associated block of code is executed.

If there is no match, the default code block is executed.

Example

```
<script>
let day;
switch (new Date().getDay()) {
```

```
  case 0:
    day = "Sunday";
    break;
  case 1:
    day = "Monday";
    break;
  case 2:
    day = "Tuesday";
    break;
  case 3:
    day = "Wednesday";
    break;
  case 4:
    day = "Thursday";
    break;
  case 5:
    day = "Friday";
    break;
  case  6:
    day = "Saturday";
}
document.getElementById("demo").innerHTML = "Today is " + day;
</script>
```

Output

Today is Saturday

1.4.4 Operators

JavaScript Arithmetic Operators

Arithmetic operators perform arithmetic on numbers (literals or variables).

Operator	Description
+	Addition

-	Subtraction
*	Multiplication
**	Exponentiation (ES2016)
/	Division
%	Modulus (Remainder)
++	Increment
--	Decrement

Arithmetic Operations

A typical arithmetic operation operates on two numbers.

The two numbers can be literals:

Example:

let x = 100 + 50;

Output

150

Operators and Operands

The numbers (in an arithmetic operation) are called operands.

The operation (to be performed between the two operands) is defined by an operator.

Operand	Operator	Operand
100	+	50

1.4.5 Array

Creating an Array

Syntax:

const *array_name* = [*item1*, *item2*, ...];

Example:

1) const cars = ["Volvo", "BMW"];

Output:

Volvo,BMW

2) provide the elements:

```
const cars = [];
cars[0]= "Volvo";
cars[1]= "BMW";
```

Output:

Volvo,BMW

3) Accessing Array Elements

You access an array element by referring to the **index number**:

```
const cars = ["Volvo", "BMW"];
let car = cars[0];
```

Output:

Volvo

1.4.6 Function

Functions are one of the fundamental building blocks in JavaScript. A function in JavaScript is similar to a procedure—a set of statements that performs a task or calculates a value, but for a procedure to qualify as a function, it should take some input and return an output where there is some obvious relationship between the input and the output. To use a function, you must define it somewhere in the scope from which you wish to call it

Example

```
<script>

function myFunction(p1, p2) {

 return p1 * p2;

}

let result = myFunction(4, 3);

document.getElementById("demo").innerHTML = result;

</script>
```

Output:

12

1.4.7 Arrow Function

Arrow functions were introduced in ES6.

Arrow functions allow us to write shorter function

syntax:

```
let myFunction = (a, b) => a * b;
```

Example

<script>

let hello = "";

hello = () => {

  return "Hello World!";

}

Output

Hello World!

1.5 Installation for React

Facebook developed React JS .React js is an open-source component-based front-end JavaScript library. It is used to create fast and interactive user interfaces for web and mobile applications. It is easy to create a dynamic application in React because it requires less coding and offer more functionality. It is used by big MNC and fresh new startups

Features of React:

Reusable Components: A single React app consists of many components each component have their own logic and code but we can easily reuse components any number of time hence reducing the developers time and increasing the efficiency of work

Debugging: React app can be easily debug using "react developer tools".It's a browser extension that can be used for both chrome as well as Firefox.

Installation Reactjs on Windows:

Step 1: Install Node.js installer for windows. Once downloaded open NodeJS without disturbing other settings, click on the Next button until it's completely installed.

Install the 14.18.1 LTS

Step 2: Open command prompt to check whether it is completely installed or not type the command –>

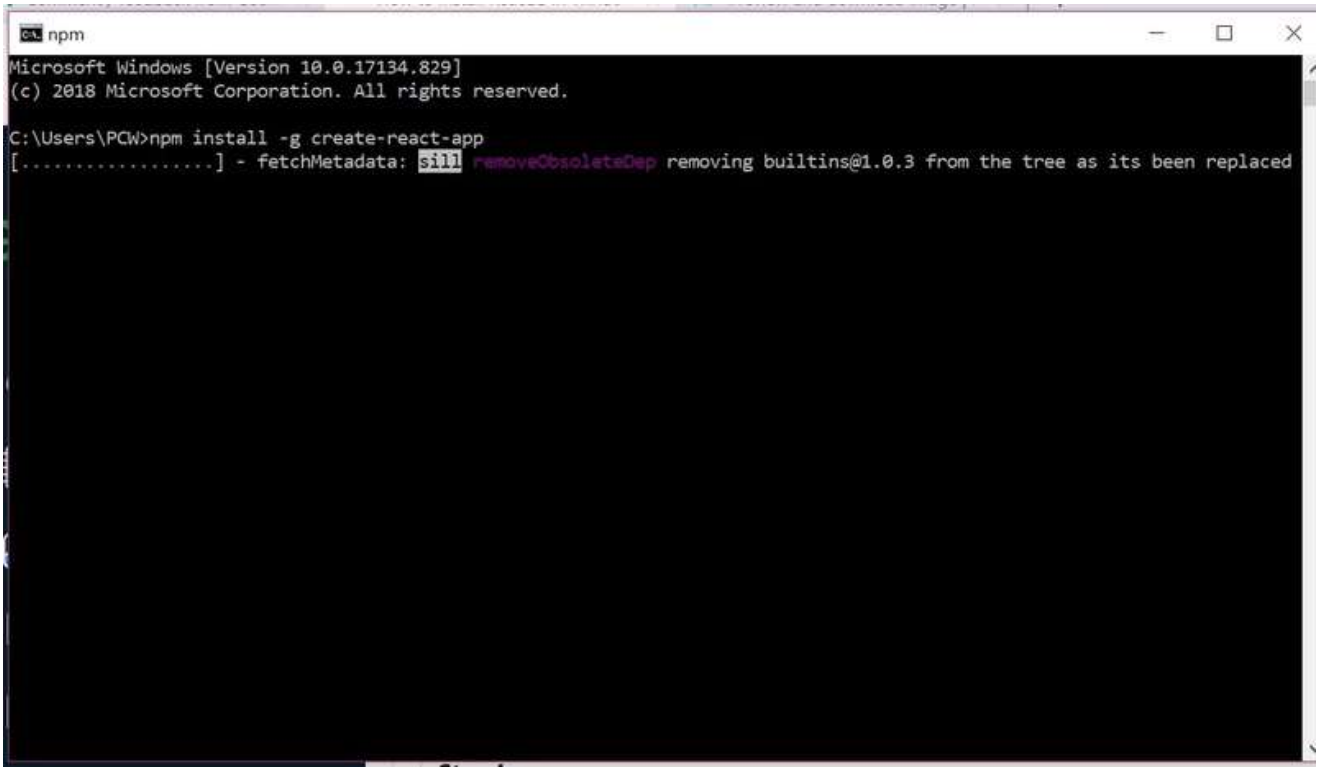
node -v

.

If the installation went well it will give you the version you have installed

1.6 Create new React App

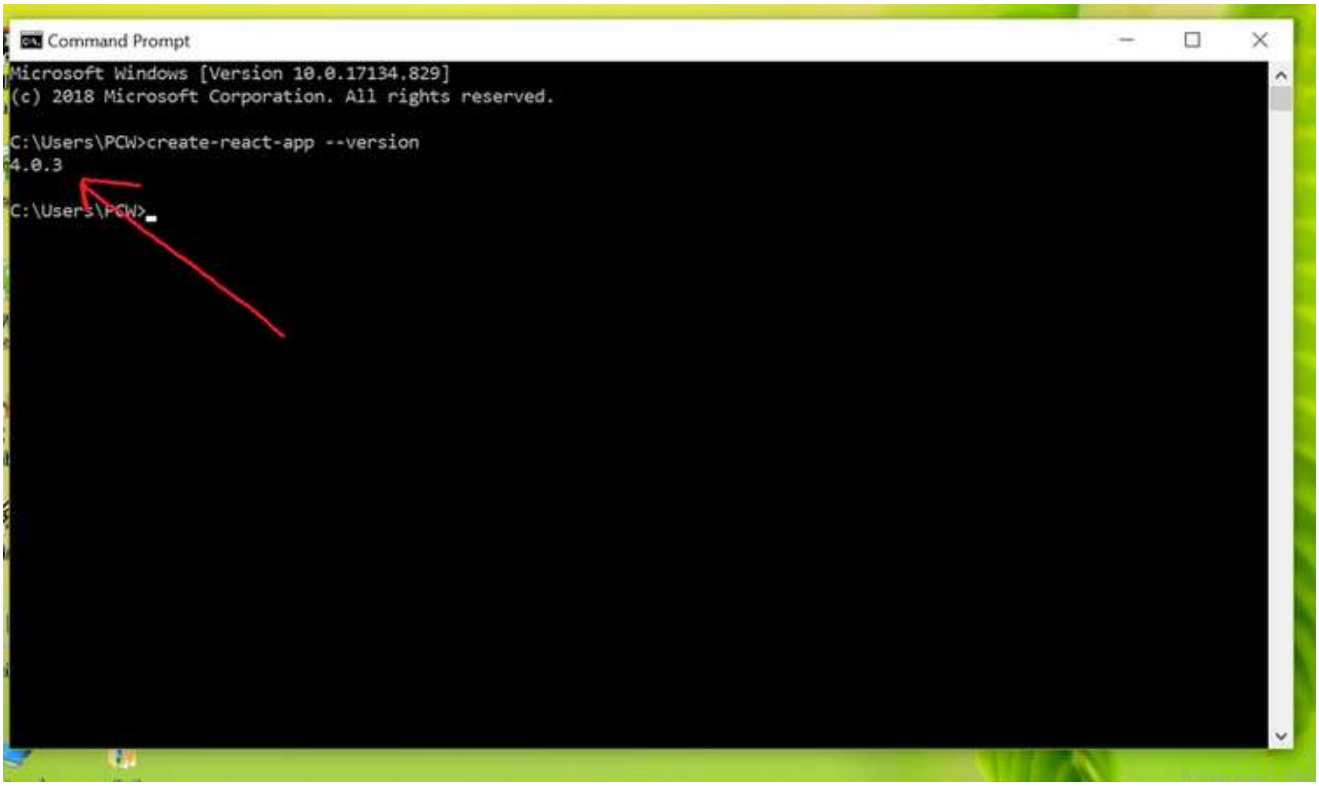
npm install -g create-react-app



Installation will take few seconds

It will globally install react app for you. To check everything went well run the command

create-react-app --version

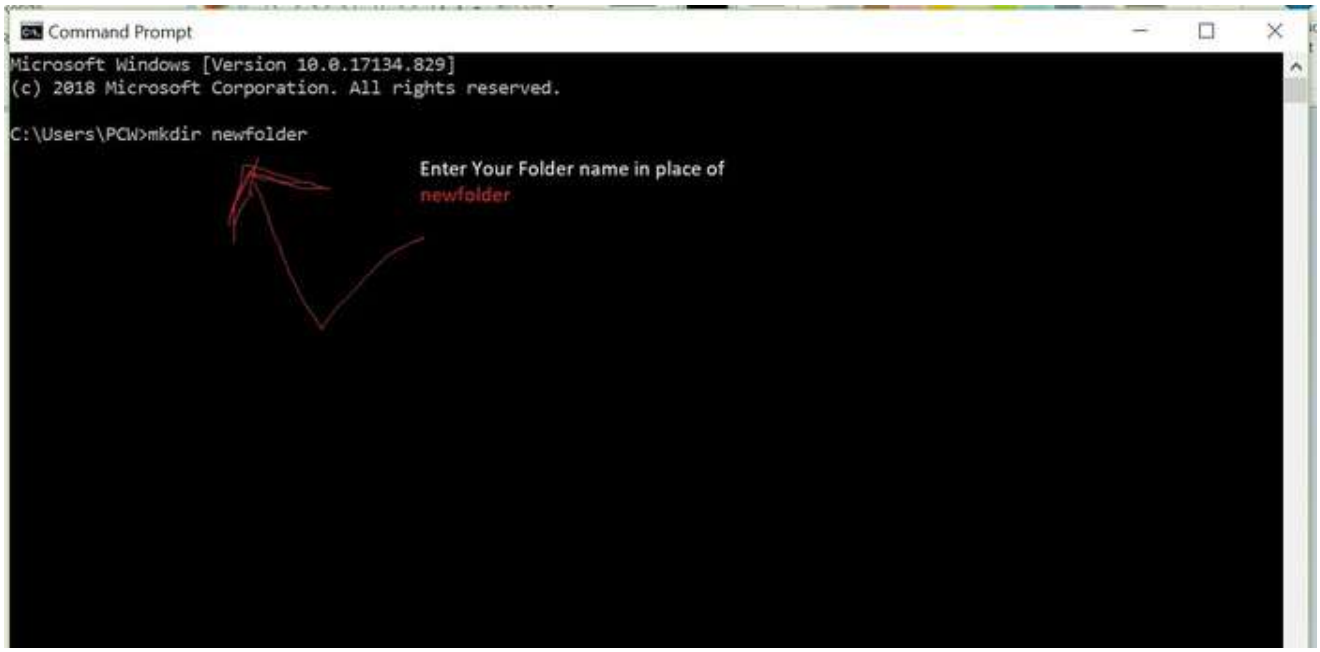


version 4.0.3

If everything went well it will give you the installed version of react app

Step 4:Now Create a new folder where you want to make your react app using the below command:

mkdir

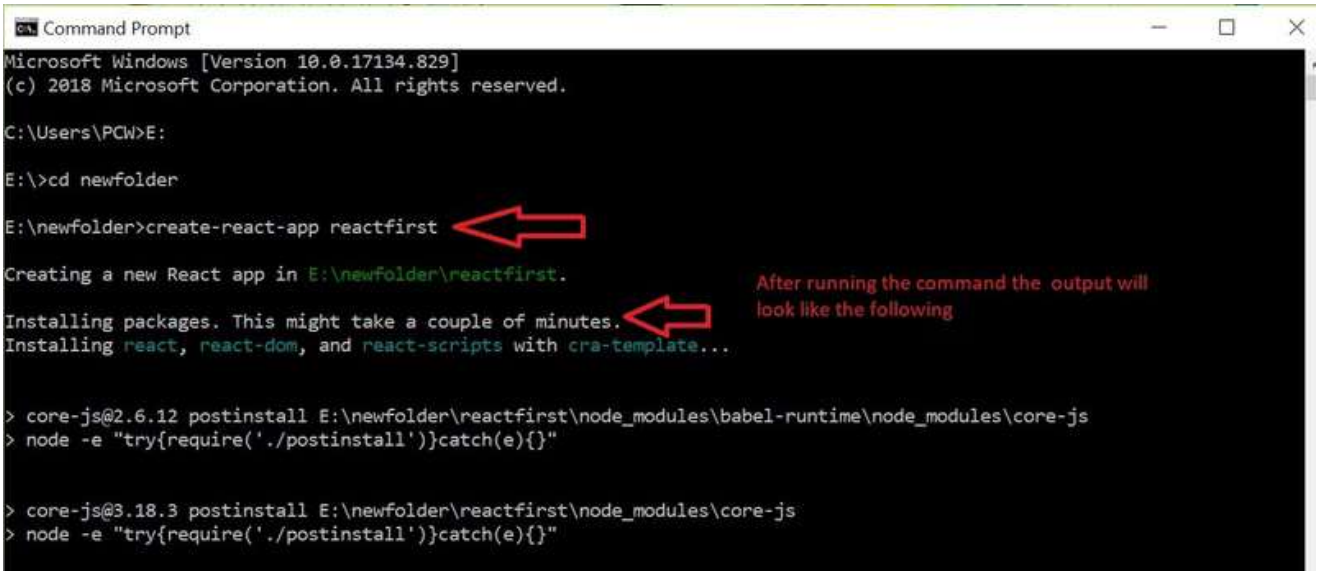


Move inside the same folder using the below command:

cd new folder (your folder name)

Step 5: Now inside this folder run the command –>

create-react-app reactfirst YOUR_APP_NAME

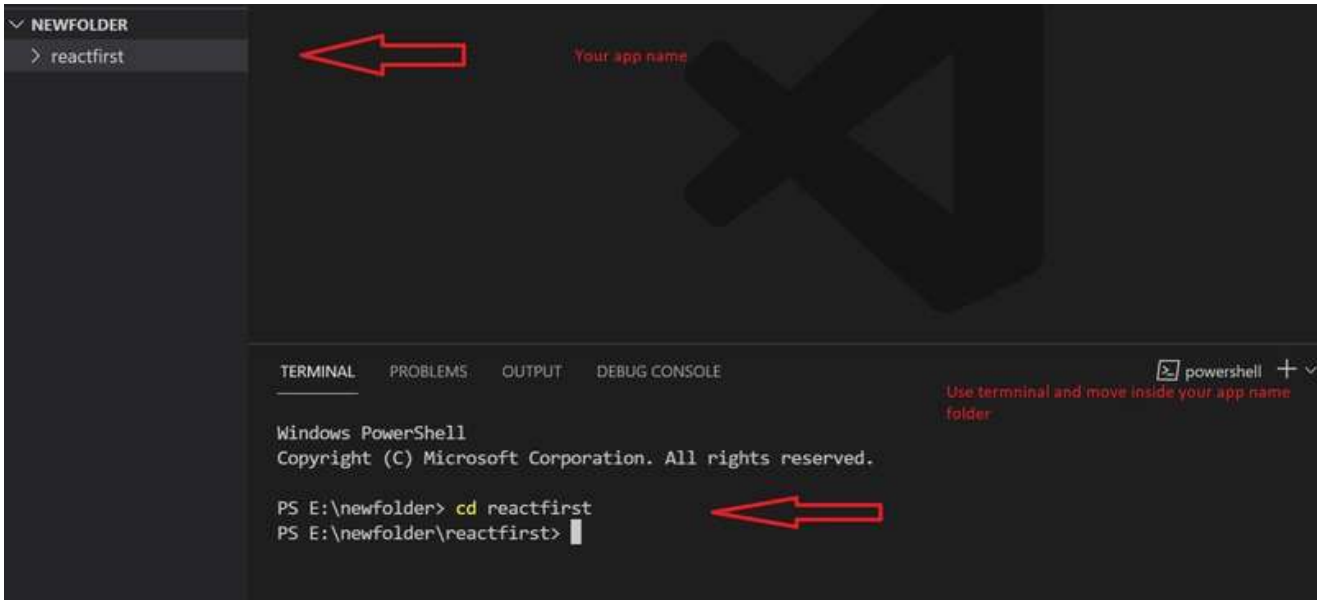


It will take some time to install the required dependencies

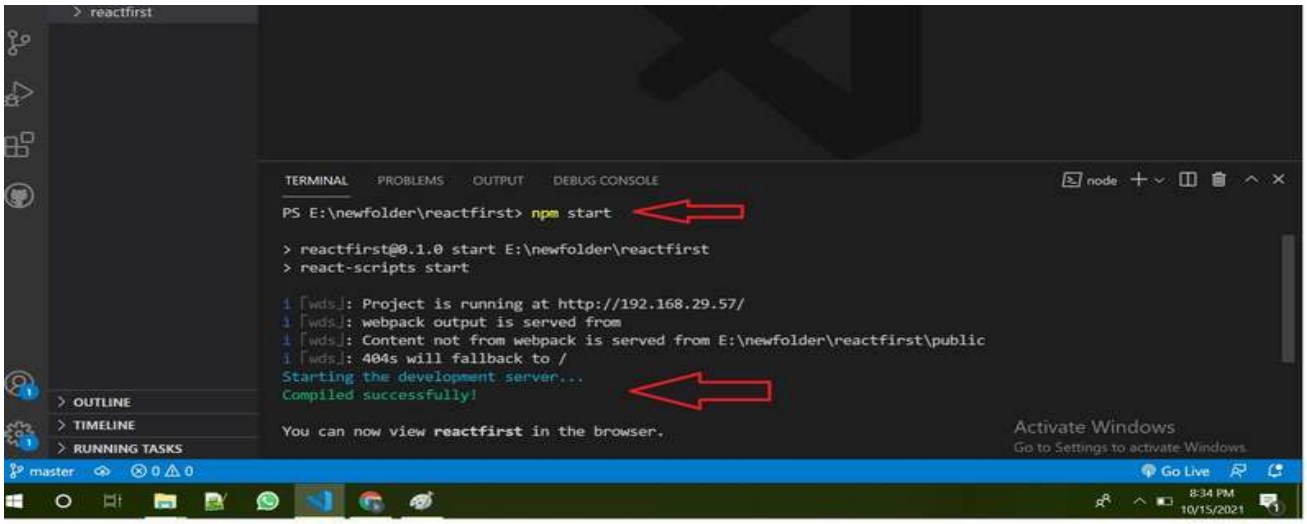
NOTE: Due to npm naming restrictions, names can no longer contain capital letters, thus type your app's name in lowercase.

```
C:\Users\sa552>create-react-app FirstReact
Cannot create a project named "FirstReact" because of npm naming restrictions:

  * name can no longer contain capital letters

Please choose a different project name.
```

Step 6: Now open the IDE of your choice for Visual studio code and open the folder where you have installed the react app new folder inside the folder you will see your app's name reactapp (In our example). Use the terminal and move inside your app name folder. Use command cd reactapp (your app name)



Step 7: To start your app run the below command : npm start



Once you run the above command a new tab will open in your browser showing React logo as shown below :

Congratulation you have successfully installed the react-app and are ready to build awesome websites and app

Last Updated : 06 Feb, 2023

1.7 JSX

The render function specifies the HTML output of a React component. JSX(JavaScript Extension), is a React extension which allows writing JavaScript code that looks like HTML. In other words, JSX is an HTML-like syntax used by React that extends ECMAScript so that HTML-like syntax can co-exist with JavaScript/React code. The syntax is used by preprocessors (i.e., transpilers like babel) to transform HTML-like syntax into standard JavaScript objects that a JavaScript engine will parse.

JSX provides you to write HTML/XML-like structures (e.g., DOM-like tree structures) in the same file where you write JavaScript code, then preprocessor will transform these expressions into actual JavaScript code. Just like XML/HTML, JSX tags have a tag name, attributes, and children.

Why use JSX?

It is faster than regular JavaScript because it performs optimization while translating the code to JavaScript.

Instead of separating technologies by putting markup and logic in separate files, React uses components that contain both. We will learn components in a further section.

It is type-safe, and most of the errors can be found at compilation time.

It makes easier to create templates.

Nested Elements in JSX

To use more than one element, you need to wrap it with one container element. Here, we use div as a container element which has three nested elements inside it.

1.7.1 JSX Attributes

JSX use attributes with the HTML elements same as regular HTML. JSX uses camelcase naming convention for attributes rather than standard naming convention of HTML such as a class in HTML becomes class Name in JSX because the class is the reserved keyword in JavaScript. We can also use our own custom attributes in JSX. For custom attributes, we need to use data- prefix. In the below example, we have used a custom attribute data-demo Attribute as an attribute for the <p> tag.

Example

```
import React, { Component } from 'react';
class App extends Component{
  render(){
    return(
      <div>
        <h1>hello world</h1>
       <h2>hello world1</h2>
      </div>
    );
  }
}
export default App;
```

Example

hello world

hello world1