HTML, CSS, Javascript

HTML: It is for displaying purpose

CSS: It is mainly for styling purpose

Javascript: It is used to add interactivity to the websites

Editors:

* VS Code
* Live Server – plugin in vscode

HTML elements

body, p, table, div, ol, ul, li, form, h2, h1, h3, h4, h5, h6

HTML5 features

1. Form validation features
2. canvas element
3. progress bar
4. range bar
5. session storage
6. local storage

CSS: It is used to add the styles to the HTML

How to link the CSS with HTML

<link> tag is used

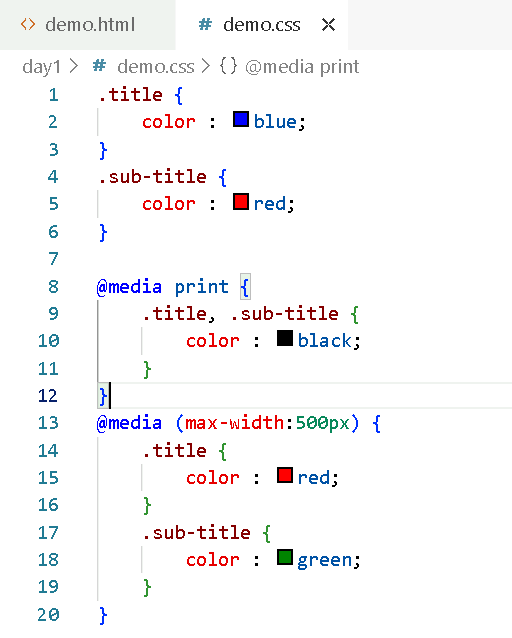
Media Queries

It helps you to create Responsive Web Design, it makes your page fit to all the devices based on the device size

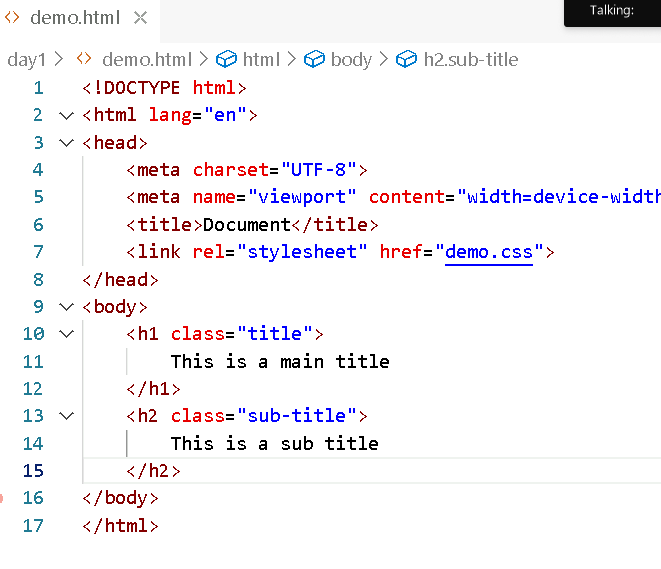
view port: It controls the users visible area so that you don’t have to design the web page differently for different devices

Media Query is a CSS technique to apply the CSS for different device size.

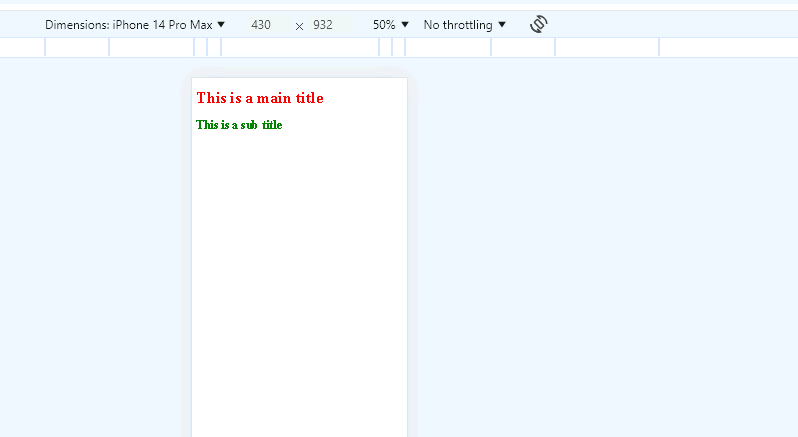
demo.css



demo.html



Output:



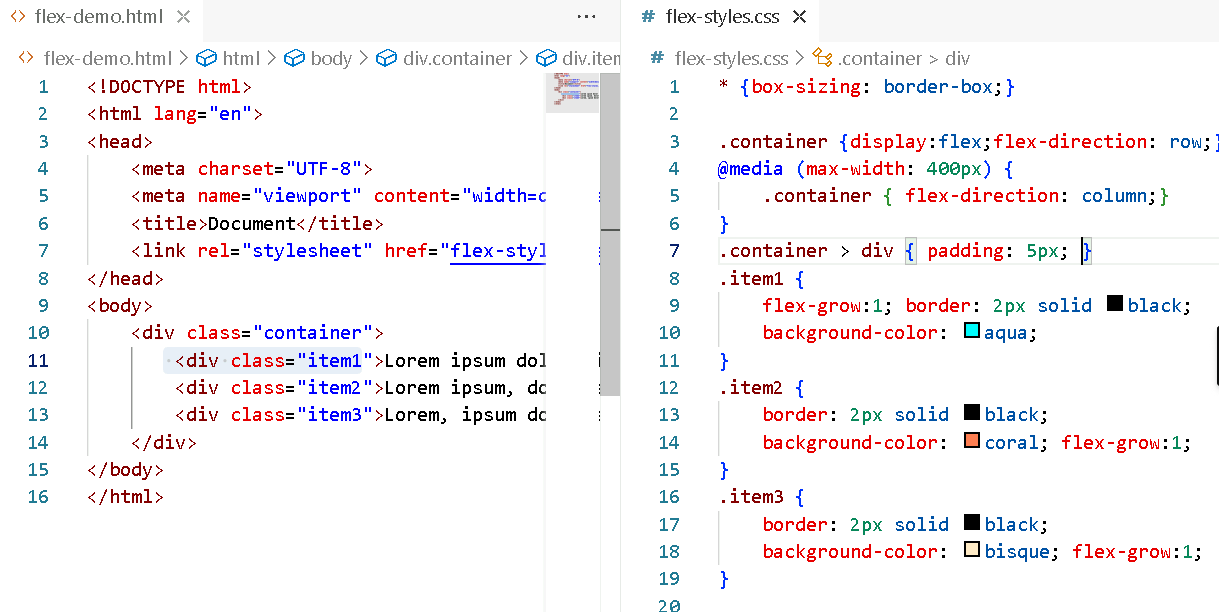
Flex box layout

It is an efficient way to fill the available space to accommodate all kind of devices & screen

Flex container: It can expand or shrink whenever is required

display: flex

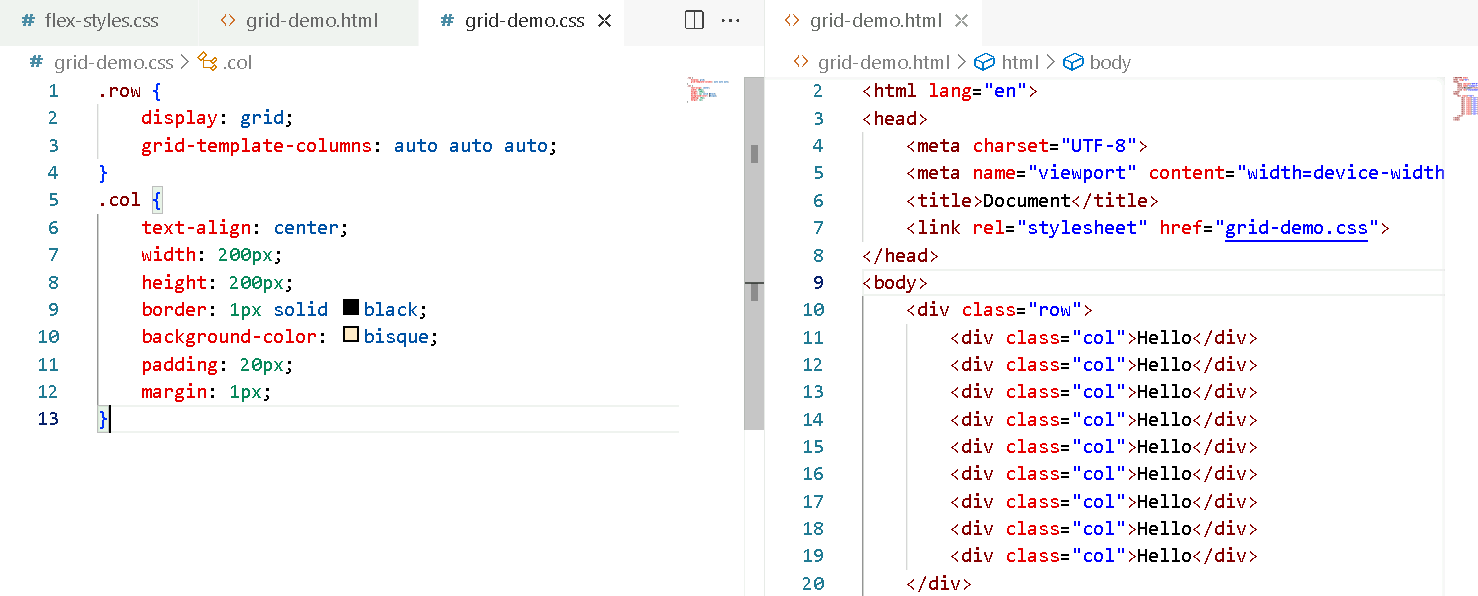
It is the property used to achieve flex layout



Grid layout

When you want to design web pages in row & column layouts then you can use the grid property.

{   
 display : grid;  
 grid-template-columns: auto auto auto;  
}



Javascript

It can modify HTML & CSS at runtime

It can handle events that occur on the web page and can do some task based on the events.

Fundamentals of Javascript

1. functions
2. variables
3. operators
4. arrays
5. loops
6. conditions
7. objects
8. event handling

Functions: These are block of code which are named after the function keyword they can take parameters and also return results to their callers

syntax:

function functionName(arg, arg, arg) { …. statements; return exp; }

Variables: These are the memory they store values in the application

syntax:

var x = 20;

In Javascript there is not datatypes, if a variable store a number, the name variable can also store string

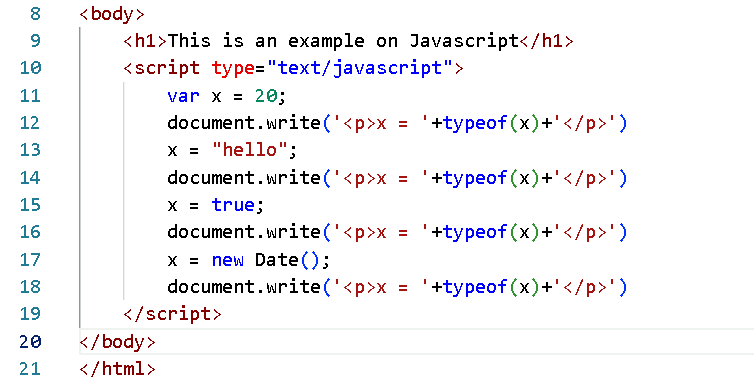
x = “hello”;

x = true;

x = new Date();

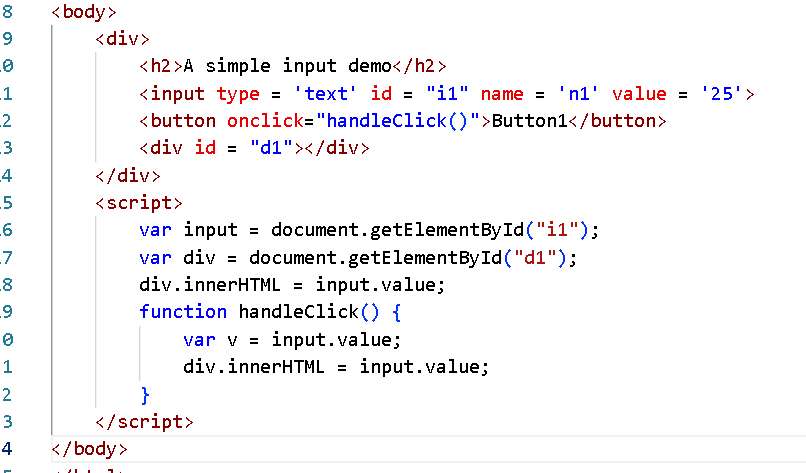
Operators: These are the special symbols which do operations on the data

=, ==, ===, +, -, /, \*, %, ++, --, <=, >=, !=



Event handling

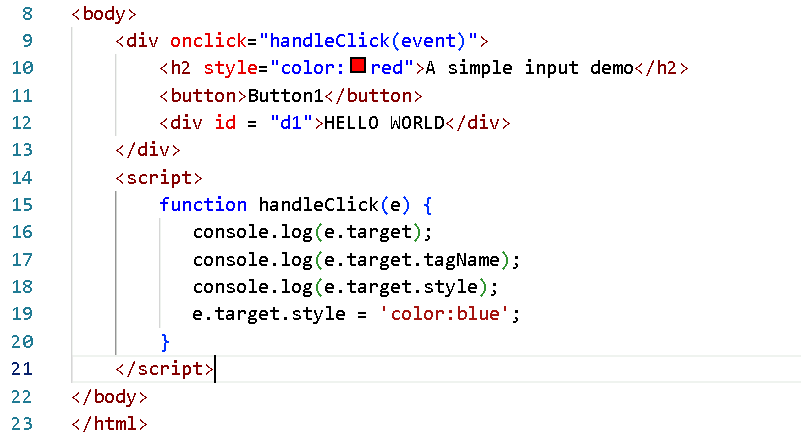
HTML elements generate an event like click, submit, change, blur and so on, when these events occur you an invoke javascript functions



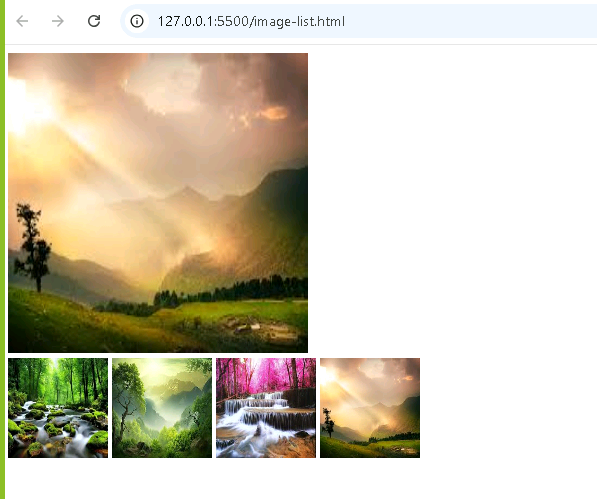
What are the ways you can access the HTML element

document.getElementById(..)  
document.getElementsByTagName(..)  
document.getElementsByClassName(..)  
document.querySelector(..)  
document.querySelectorAll(..)

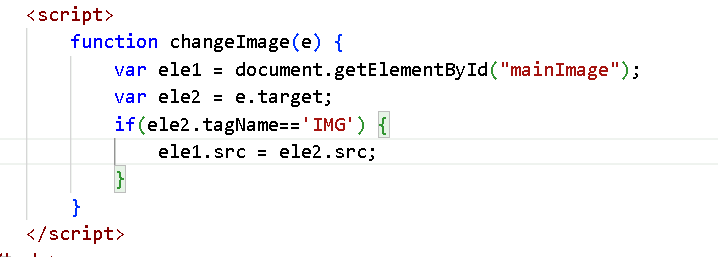
We can handle event and identify the element generate an event using event.target



Activity:



When you click on the smaller images then the top image must be replaced by the image you click



Form validation

Though you can do validations in HTML5, you can use Javascript to perform any particular validation which may not be possible with HTML5

ex: Stopping user to enter a number or a special character in a firstname & lastname text box



Day 2 Agenda

* Modern Javascript syntax
* Introduction to PWA
* Difference between PWA vs TWA vs Native app
* Principles of PWA
* Core Technologies of PWA

Modern syntax of Javascript i.e, ES6 and later features

* ECMA Script classes
* New keywords like let, const, super, extends, class
* Template strings
* Arrow functions
* Default parameters
* Destructuring
* Rest & Spread
* Generators
* Promises
* Async/Await
* Optional Chain

How to create an object in Javascript

1st way

In a literal style

employee = { id : 100, name : “Raj”, salary : 45000 }

2nd way

Creating a function constructor and instantiating the object

function Employee(id, name, salary) { this.id = id; this.name = name; this.salary=salary }   
emp = new Employee(100, “Raj”, 45000);

Suppose you want to add an extra function to an object then you must use a prototype of an object and add the function

Employee.prototype.display = function() {   
 console.log(this.id, this.name, this.salary)  
}

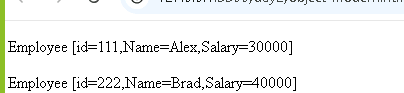
emp.display();

New syntax introduces the concept of classes & constructor to initialize the properties

class Employee {   
 constructor(id, name, salary) { this.id = id; this.name = this.name; this.salary=salary; }  
 display() { console.log(this.id, this.name, this.salary); }  
}



Output:

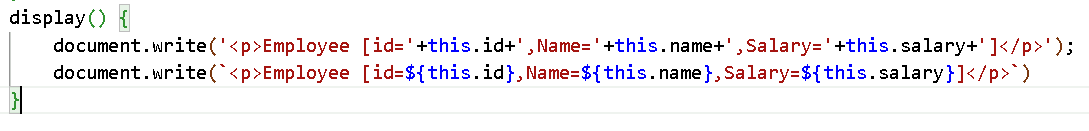


Template String literals

It helps in using javascript expression inside the string easy way without using + operator, you can use the backtick character to represent the string & using ${expression} inside the backtick to use the javascript expression.

“<p>Hello “+name+”</p>”;

`<p>Hello ${name}</p>`



Activity: Declare an array of employee objects and print each employee object property in a table

var employees = [ new Employee(123, “Raj”, 45000), new Employee(456, “Vijay”, 5000) ]

using back tick character print id, name & salary in a table by iterating the above array.

Arrow functions:

Arrow functions are the simplest form writing the callback functions, they reduce the code when used in place of callbacks

|  |  |
| --- | --- |
| Callback Function | Arrow Function |
| function(a, b) {  return a + b;  } | (a, b) => { return a + b }  or  (a, b) => a + b; // by default returns expression |
| function(a) {   console.log(a) } | a=>console.log(a) |

items = [2, 3, 1, 4, 6];

items.forEach( function(value, index) { stmt; } );

for(x = 0; x < items.length; x++) { items[x] }

items.forEach( (value, index) => stmt );

array also have map function:

It is for transformation, it takes callback as a parameter but returns a new array once all the elements are iterated

newItems = items.map( function(value, index) { return value; } );

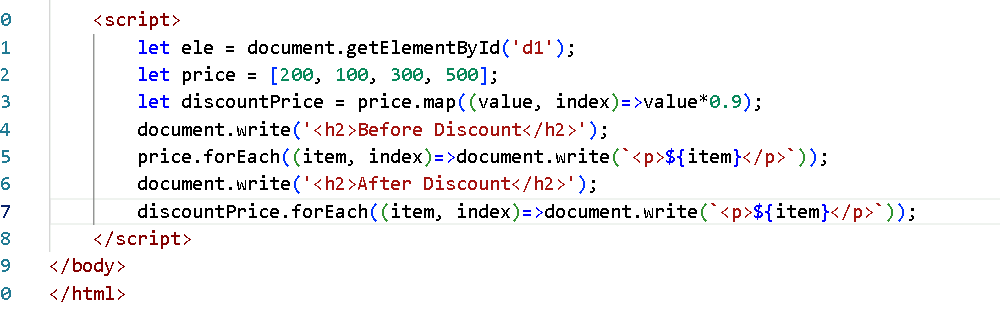
newItems = items.map( (value, index) => value );

let & const: Both are used to create block scoped variables, this can be used as a replacement to the var

const x = 20;

let y = 30;

var z = 40;



Array Destructuring & Object Destructuring

Destructing is used in order to access the data in a simplest way in a single line.

let items = [2, 3, 1, 5];

let [a, b, c, d] = items;

Object Destructuring

let employee = {id:100, name: “Alex”, phone: 9993399393};

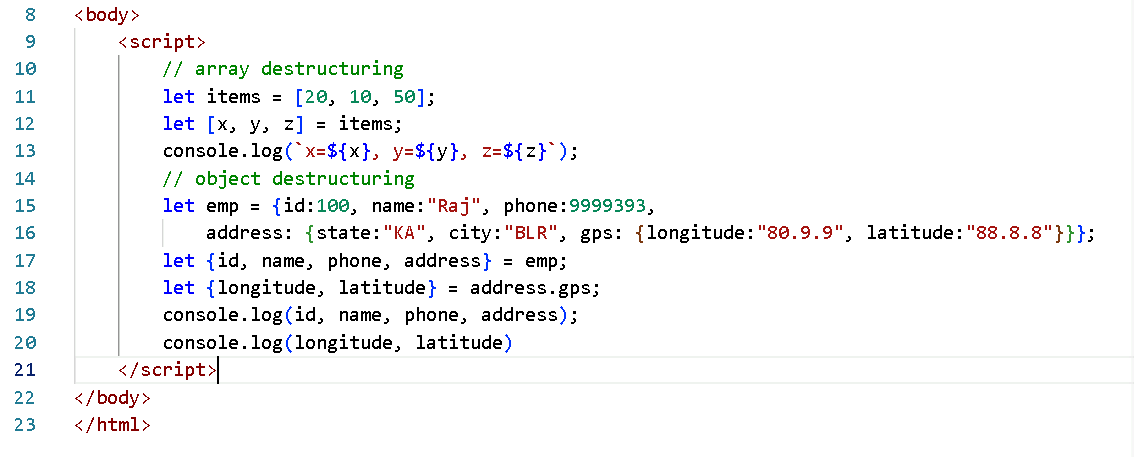
let { id, name, phone } = employee;

id = 100  
name = Alex  
phone = 9993399393

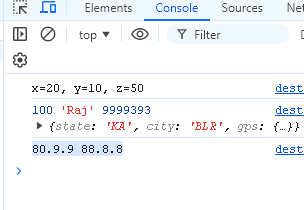
let employee = { id:100, name:Raj, address: {state: “KA”, city:”BLR”, geo : { long : 99, lat: 88 } }}

let address = employee.address;  
let id = employee.id;  
let geo = address.geo;

${employee.address.state}  
${employee.address.geo.long}



Output:

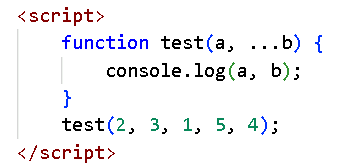


Rest & Spread operator

Rest & Spread are used when you want to take 0 or more parameters or when you want to spread multiple values to multiple variables.

// rest operator

function test(a, …b) {   
}

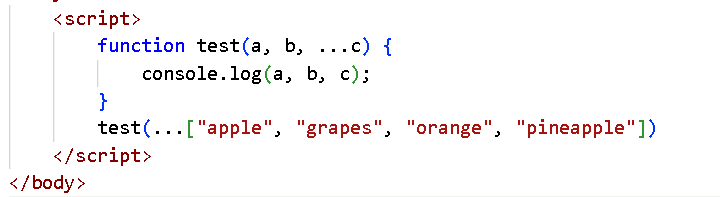


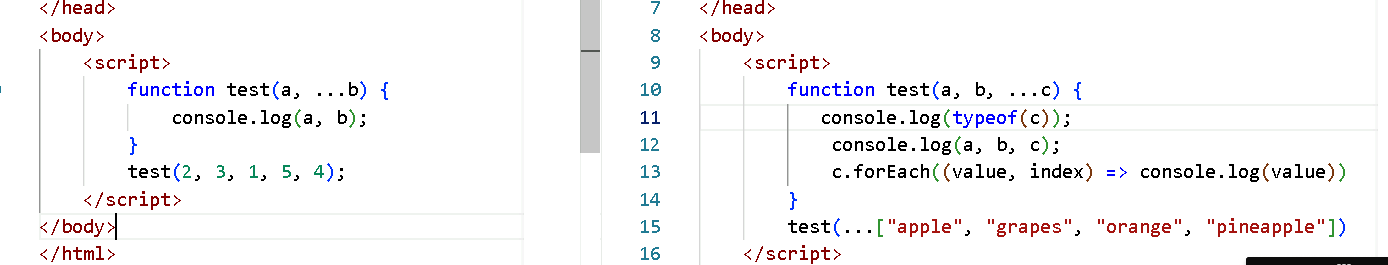
Spread operator

When you want to distribute values to multiple parameters you can use spread

let items = [“apple”, “orange”, “grapes”, “pineapple”]  
test( …items );

function test( a, b, …c ) {   
  
}





Create a function with a single parameter of rest & pass the array through spread operator & iterate each element & add them to a sum & find maximum & minimum

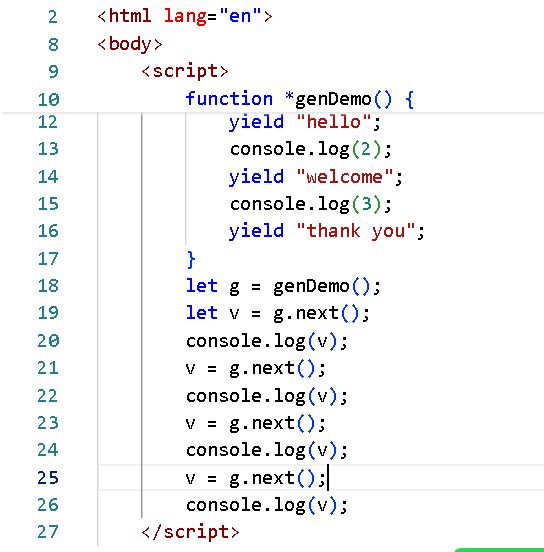
calci(…[10, -1, 5, 8, 20]);

function calci(…items) {   
 // print sum of all the values  
 // print the maximum value  
// print the minimum value  
}

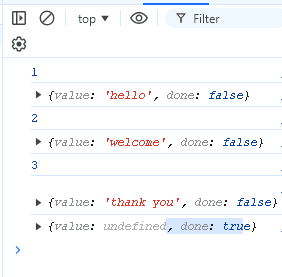
Generators in Javascript

It is a special function which can return multiple values in installment before completing the function

function \*genDemo() {   
 yield result1;  
 yield result2;  
 yield result3;  
}



Output:



Promises

These are objects that handles the asynchronous operations like calling backend API’s or accessing database and so on, Promises will have 2 states

1. resolved: When the promise is successful
2. rejected: When the promise fails

We use .then( callbackFn ) and .catch( callbackFn ) to identify whether the promise resolved or rejected

function abc() { returns a promise }

abc().then( value=> stmt ).catch(value=>stmt)

Best examples of promises are methods like

fetch(..): It returns a promise object when you supply a URL of any api

axios.get(..): it returns a promise object when you supply a URL of any api  
axios.post(..), axios.put(..), axios.delete(..) and so on.

aysnc/await: it helps you to wait for the promise to be resolved/reject and a function can return that resolved/rejected promise through the async function

Optional Chain: It takes care of handling undefined properties when you try to access nested properties



Since address is undefined on the second iteration the ?. will not access state & city on the second iteration

Progressive Web Application

A Progressive web application is a website that works like a mobile/desktop application. you can open it in your browser but it behaves like an app installed in your mobile phone / desktop.

Ex: Twitter Lite, Uber Lite, Spotify, Pinterest

Difference between traditional web application and progressive web application.

Both can be accessed using browsers but the main difference would lie in how they function, and how user interact with them.

|  |  |
| --- | --- |
| Traditional Web application | Progressive Web application |
| It will not have any offline capabilities or an application like behaviour | It is an enhanced version of a web application, it combines web sites with native mobile apps features, it provides an app like features such as offline access, push notifications & you can install it on the device |
| If the connection is lost you can’t access any pages, you see error like you need internet connection | If the connection is not there or you have a poor internet connection, you can still access the web site |
| You cannot install web applications | PWA’s can be installed directly from the browser to the users home screen similar to the native applications, here users don’t have to download from app store/play store |
| Traditional web applications don’t support push notifications | PWA’s can send push notifications, similar to native apps |
| Look & feel will be like websites | PWA’s look and feel will be like native apps |
| Performance can vary and can be slow | PWA’s are optimized for performance, with faster loading and low data usage, due to caching & background updates through service worker |
| Most of the web applications don’t have access to device features like GPS, camera or notifications | PWA’s will have access to many native device features like gps, camera, push notifications |

PWA vs Native applications

Both serve similar functionalities

|  |  |
| --- | --- |
| PWA | Native apps |
| They are platform agnostic and work across all the devices | Native apps are built specifically for a particular platform, developers need to create separate versions of native app for different platforms |
| PWA’s are easier to develop mainly because they use standard web technologies (HTML, CSS & Javascript) | native apps need separate development process for each platform |
| Performance is good | Performance is much better, because they are build using platform specific code |
| Offline capabilities are allowed | Offline capabilities allowed, but it will have full access to the device features |
| It is accessed over the browser | They installed first on the device |

When to choose PWA vs Native Apps

Whenever you need a cross platform solution & a low cost for the company & a quicker development you can always go for progressive web applications.

If you need top performance and full access to device features then you can use native apps

Principles of PWA

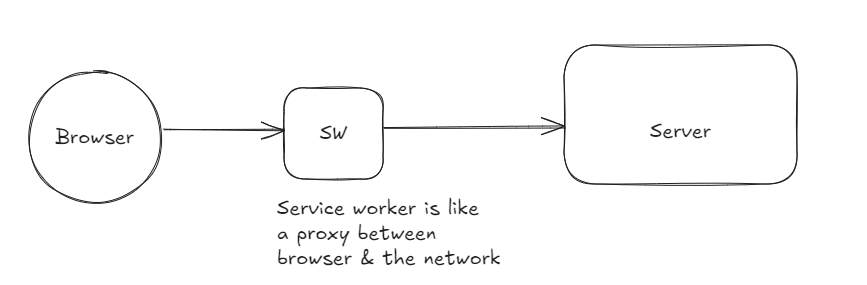
* Progressive Enhancement
* Response Design
* Connectivity Independent
* UI must be like an application
* HTTPS
* Installable

Core PWA technologies

Below are the key technologies to create PWA

* Service Worker
* HTTPS
* IndexDB or Local Storage
* Push Notifications
* Caching Strategies
* Background sync

Service Workers



It is a background program which is a backbone of PWA, it acts as a proxy between the browser & the network, it takes care of running in the background

* intercept network requests
* cache resources
* offline support

Phases of service worker

* Register
* Install
* Activate
* Fetch
* Wait Time

Register:

In this phase the service worker will be registered which is a javascript file, this will allow browser to find the service worker

Install:

This phase comes after registration, here you can perform pre-caching (html, css, js, images)

Note: The install event occurs only when the service worker file is updated or its new

Activate

This phase comes after the installation phase, at this point of time the service worker takes control of the pages

Fetch

This comes after activated phase, this can intercept the network requests using the fetch event

First & foremost thing is to check whether the browser supports the service worker not

if(‘serviceWorker’ in navigator) {   
  
} else {   
  
}

index.html

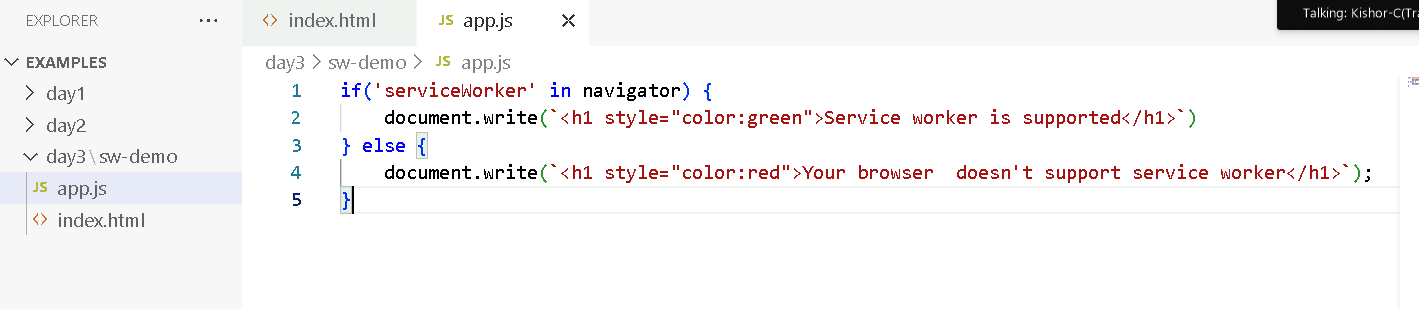


Output:



Create a Javascript file named app.js and write the service worker supporting statement in it and add that javascript in your index.html

app.js



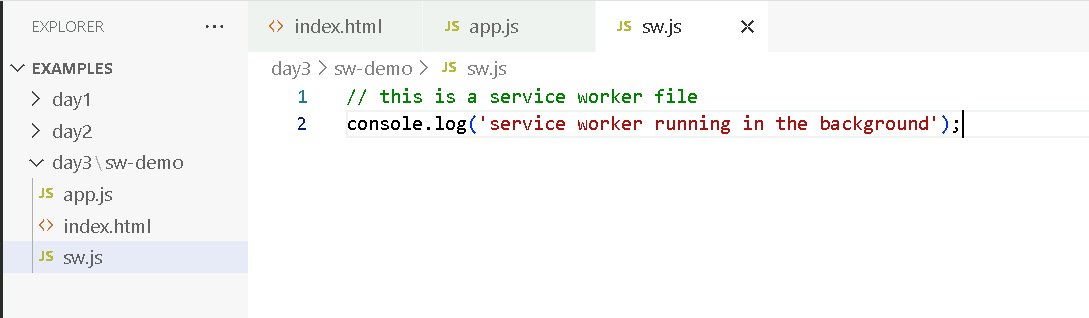
Registering the service worker

We must create one javascript file that will be running in the background to control the web pages

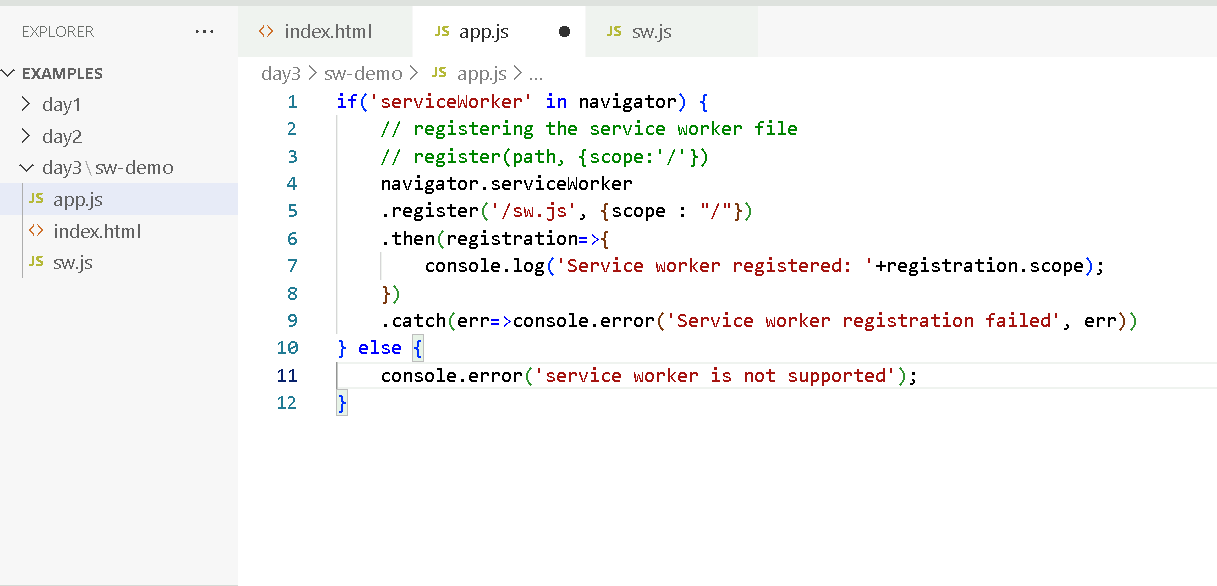
navigator.serviceWorker.register(‘/path/service-worker-file.js’, { } ).then( … ).catch(…)

Create a file called sw.js and register that as a service worker using above statement

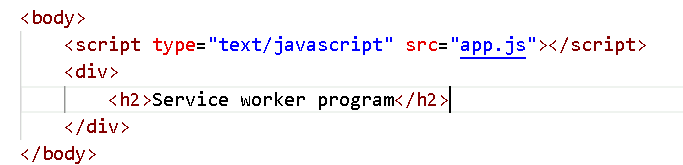
sw.js



app.js

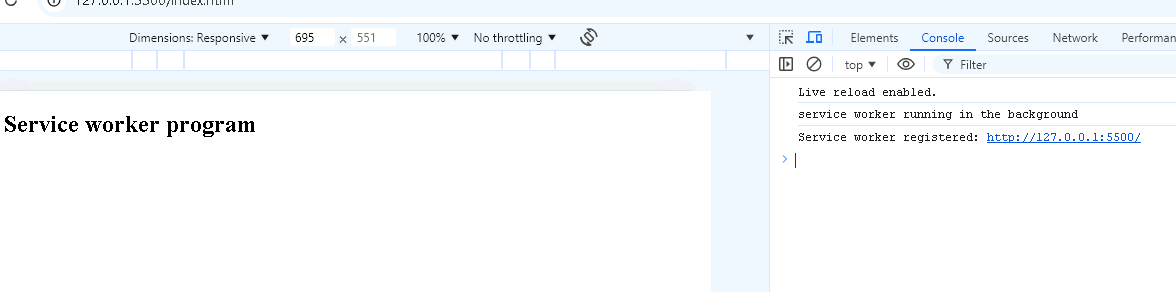


index.html

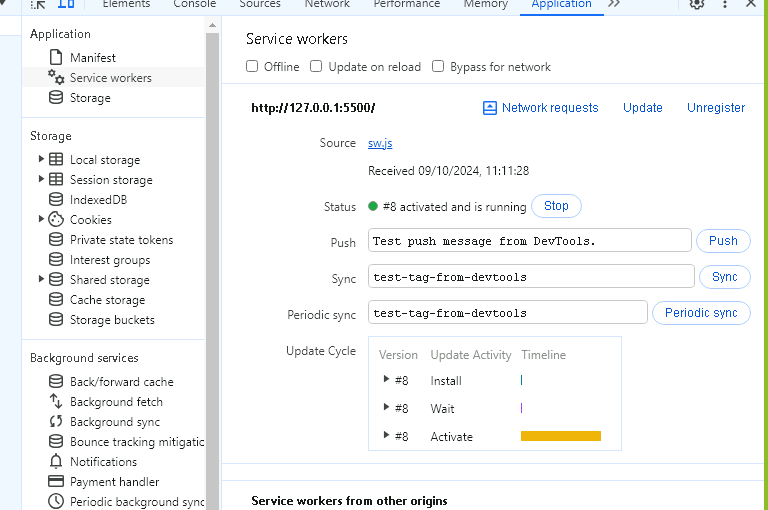


Output:

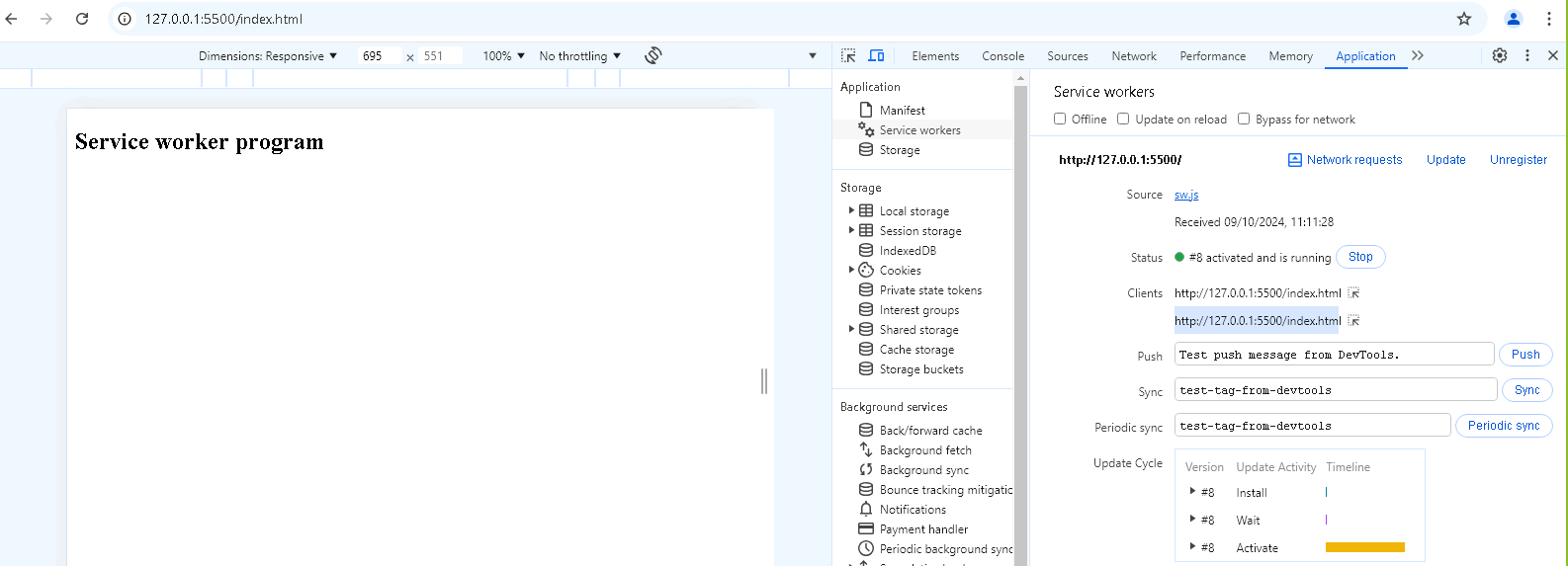
In the browser -> application -> you must see the sw.js registered and also console output



Goto application tab in the developer tools to see the sw.js file



Open another tab and send the request to the application and observe the application tab about the number of clients using the application



You can notice 2 clients connected to the application, but we have only once the sw.js is loaded

What happens to service worker when you refresh the browser

It doesn’t reload again because its already registered, if you make changes to the service worker file then it will be again loaded

Note: You will see the updates only after closing all the tabs

You can check if the page currently has the service worker using controlled

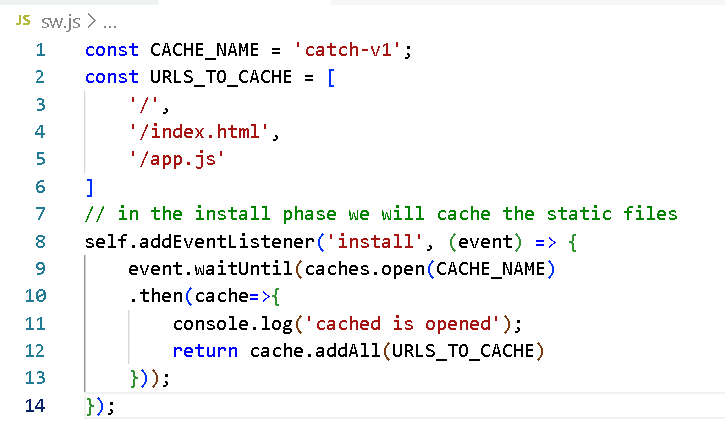
if(navigator.serviceWorker.controller) {

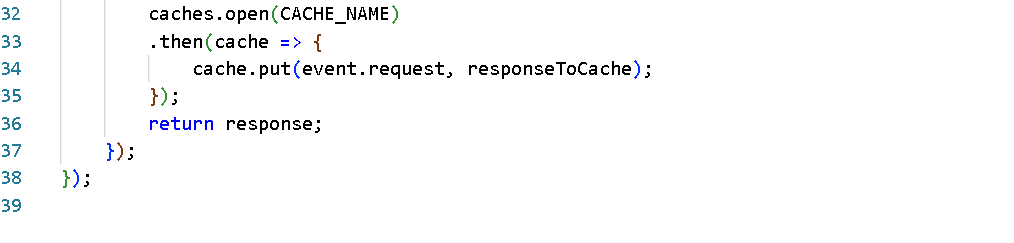
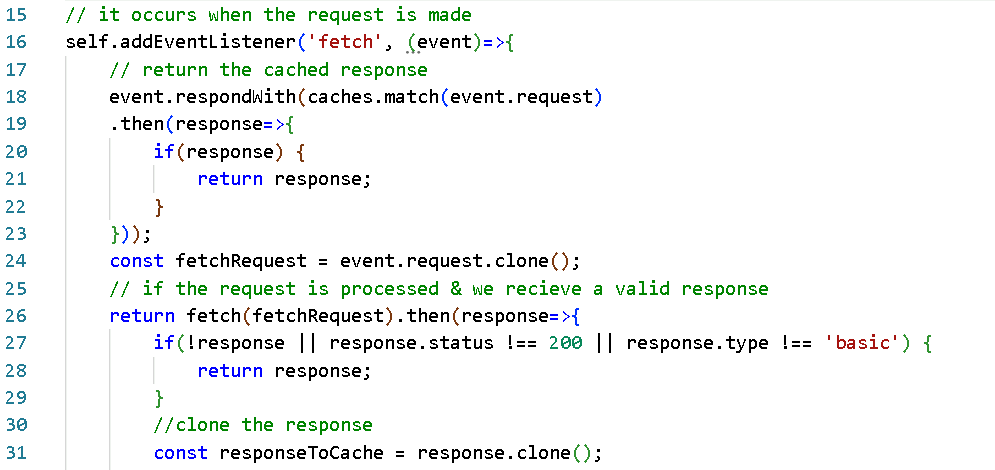
console.log(‘page has the service worker’);

}

Adding the resources into the cache

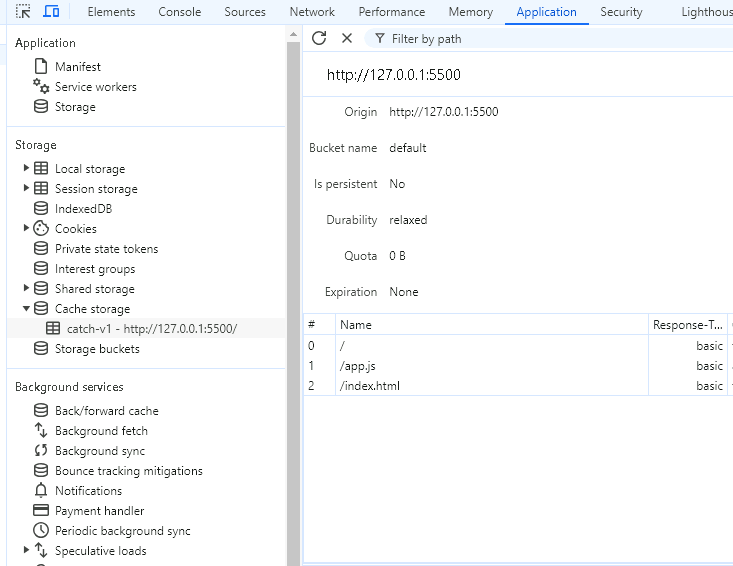
sw.js





Output:

You can stop the live server & still you could able to open the page because index.html & app.js are cached.



There are two types of strategies

1. cache first
2. network first

Cache first: Here the service worker checks the cache first if the resource is available in the cache, if available it returns the cached version else it fetches from the network and stores in the cache for future use

Network first: Here service worker makes a request over the network first, if the request fails it will fallback to the cached version of the resource

event.respondWith( fetch(event.request).then(..)

When to use these strategies

Network first: can be used in the scenarios where live data is required like news

Cache first: for static assets like scripts, stylesheets

Day 4 Agenda

* Push Notification
* Indexed DB

Push Notification:

Server --> Send -> push notification (event occurs like received mail/messages) -> client (subscriber receives the notification)

This can be implemented using below programs

1. Service Worker: Background program/script that will be running to handle push event and other service worker jobs(install, activate, fetch and etc)
2. Notification API at the client side: If the service worker gets a push event then, service-worker will use Notification API to display the notification
3. Push API at the server side: This API is used to send messages to the client (browser)

Note: By default browser wouldn’t allow notifications for any application we must enable it

Things to create

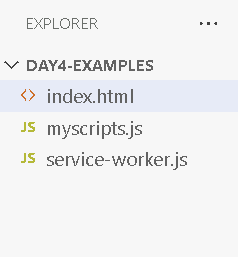
We need to write the client programs

1. myscript.js: Takes care of having logics related to registering service worker & enabling the notifications, requesting permissions to enable the notification
2. index.html: To load the myscript.js
3. service-worker.js: To have service worker code

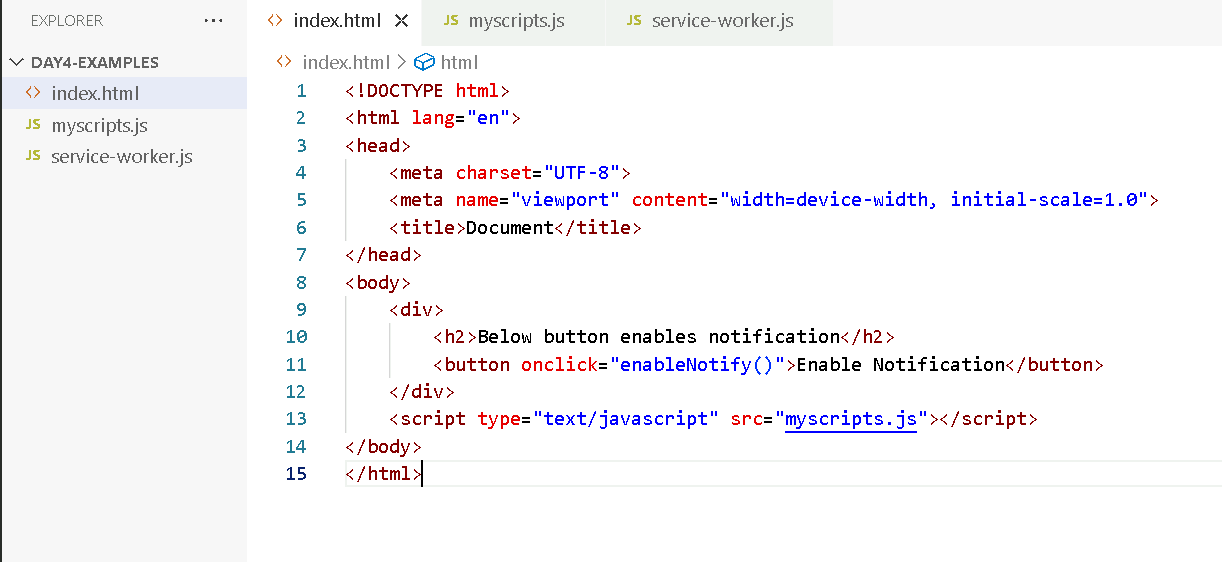
We need to write a server program

1. server.js: This will allow clients to subscribe & generate a notification, to allow clients to subscribe it needs a private & public key, so that client & server will use the same public key to interact
2. We will be generating private & public keys using web push library, it generates a key called VAPID keys (Voluntary Application Server Identification for Web Push)

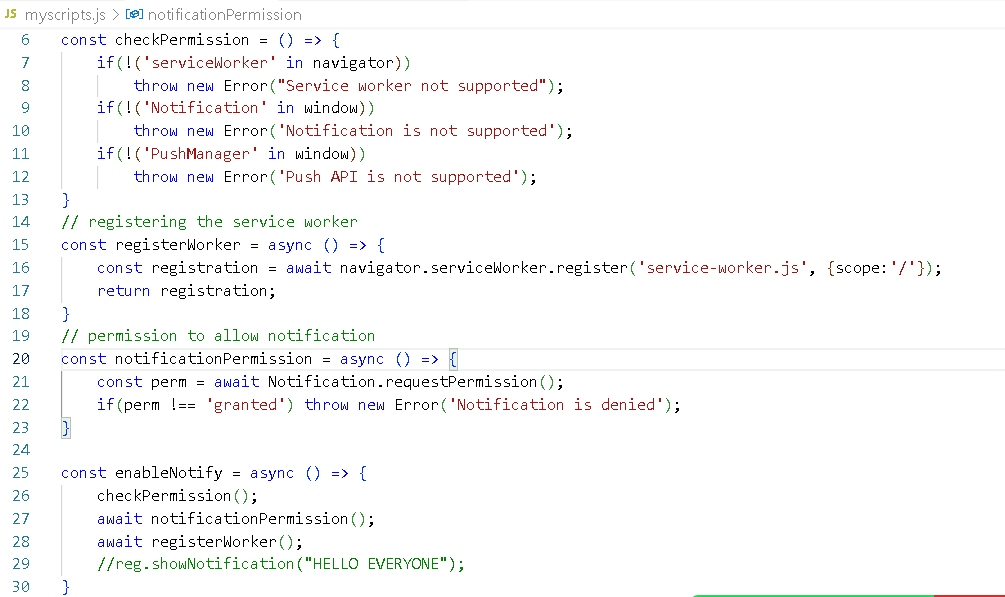
Create client programs as per the below structure



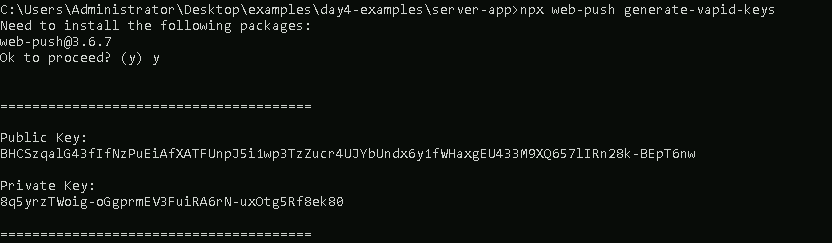
index.html



myscripts.js



Generate a vapid key using web-push and keep those private & public keys to use them in the client & server program

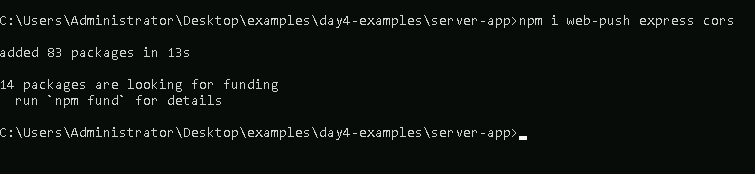


Note: Copy the public & private keys in notepad

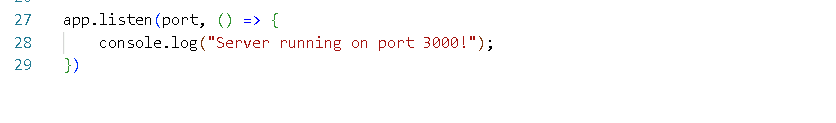
Service worker will have a subscription code, handling the push event, to subscribe to the server it must use public key that must be assigned to the applicationServerKey property which expects the public key in some string encoding format.

Server program

We must use apiKeys like private & public keys, then express, cors & web-push library



app.js



Steps:

1. Enable Notification
2. Send request to /send-notification
3. Observe the notification in the browser.

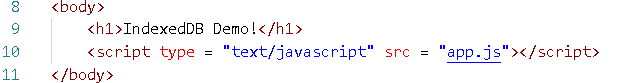
IndexedDB

It is a database available at the client side, it can store large amount of data-set, usually it stores in document format like keys & values

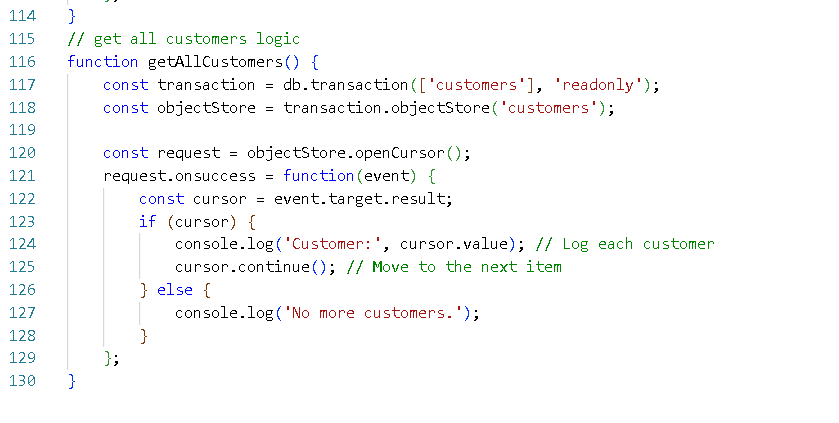
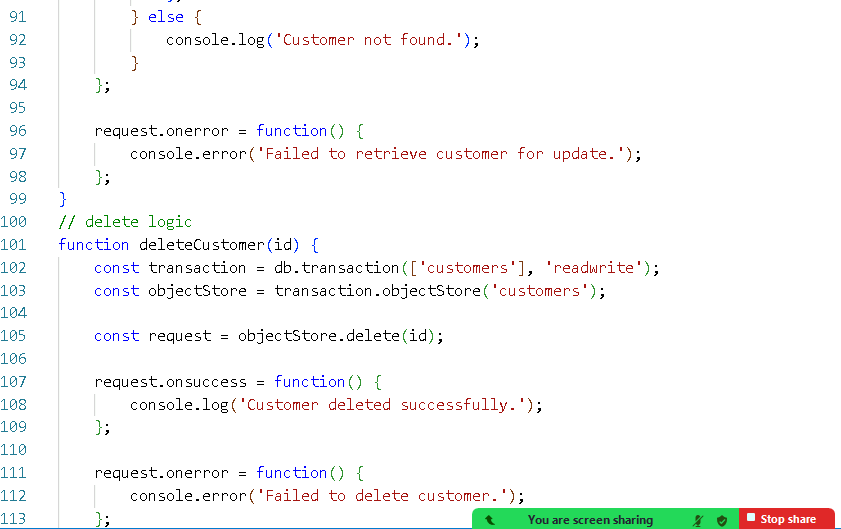
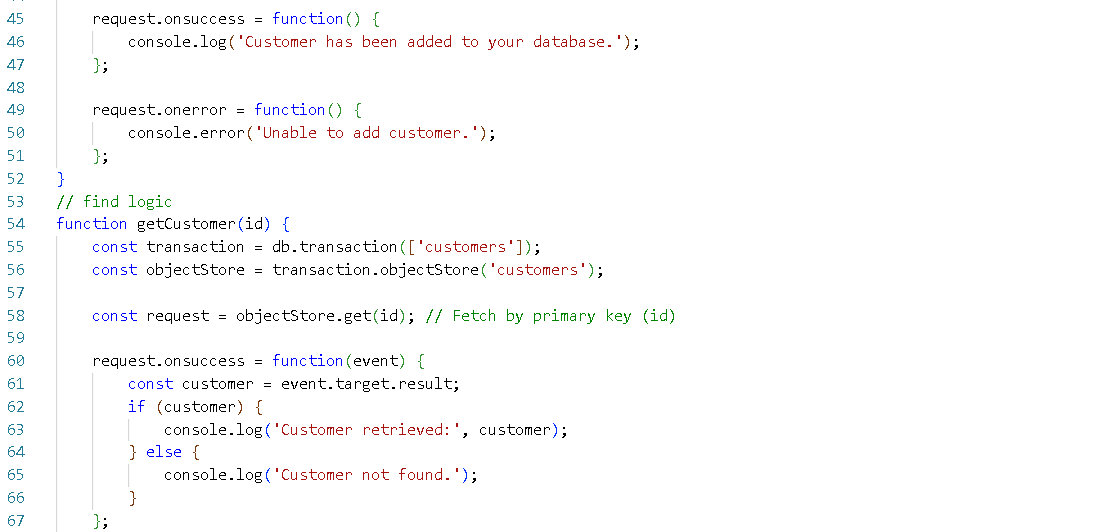
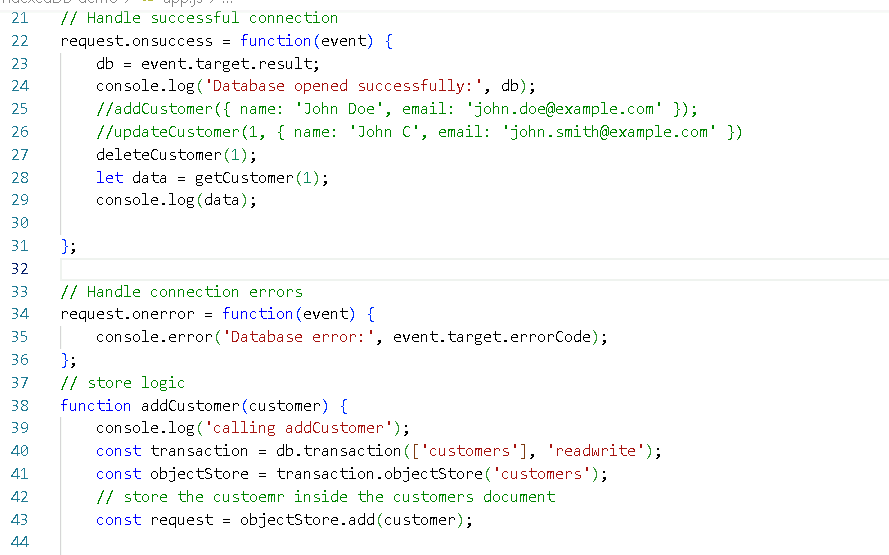
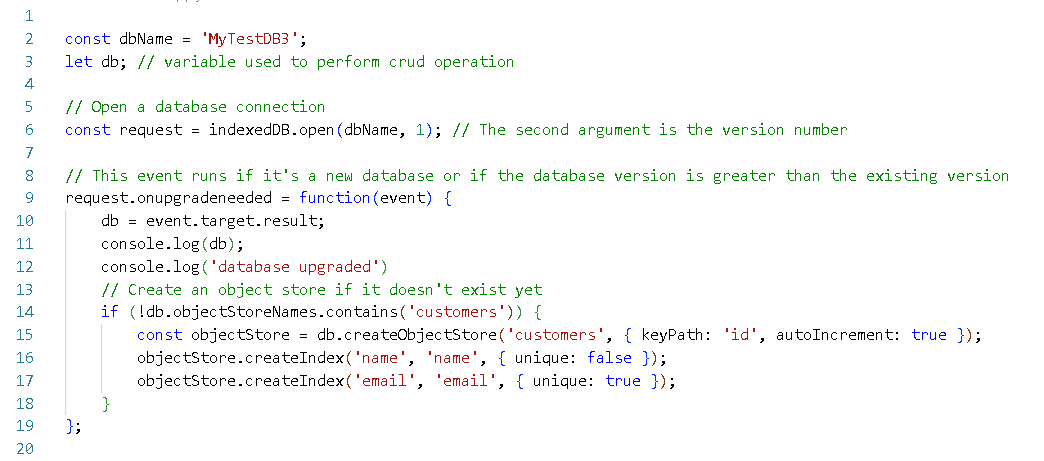
Use cases:

* Offline – progressive web-apps
* Storing audios/videos
* Form data

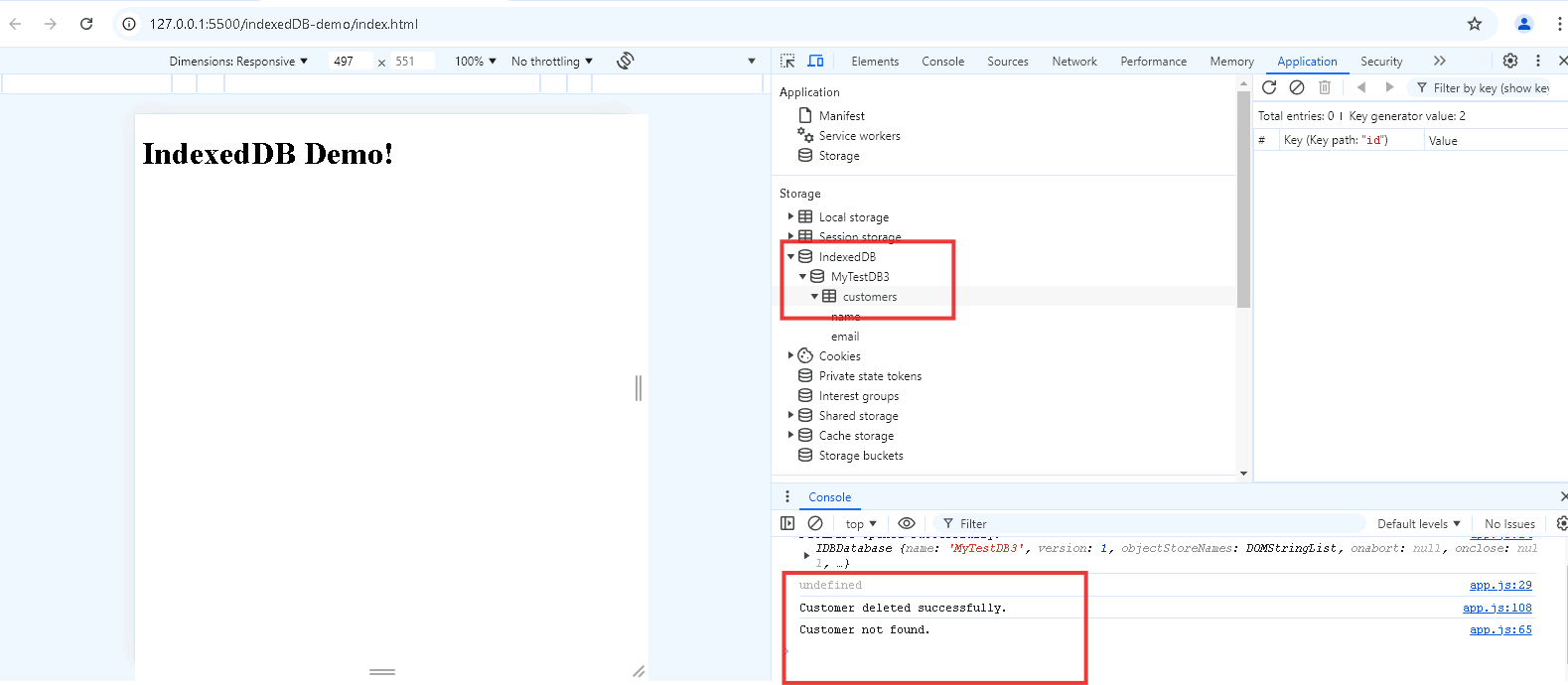
index.html



app.js



Output:



Day 5 Agenda

Configuring PWA for Angular / React

Angular / React

These are Javascript technologies helps to develop single page applications

Angular is a framework, which uses HTML, Typescript for development, it provides all the features required to develop SPA like commands to create components, commands to start, stop applications, libraries to configure routers and so on.

Angular is created by Google

React.js is a library which uses HTML & JSX (Javascript XML)

React is created by Facebook

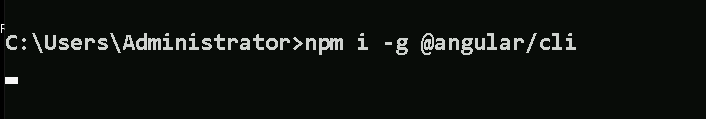
Both the companies have provided tools to quickly create applications.

Angular

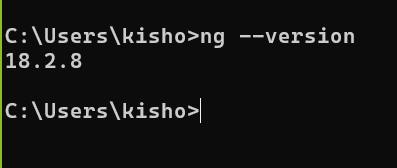
Software requirements for Angular

* Node.js
* Angular CLI: It is a tool kit helps to create & run angular application.

Installing angular cli

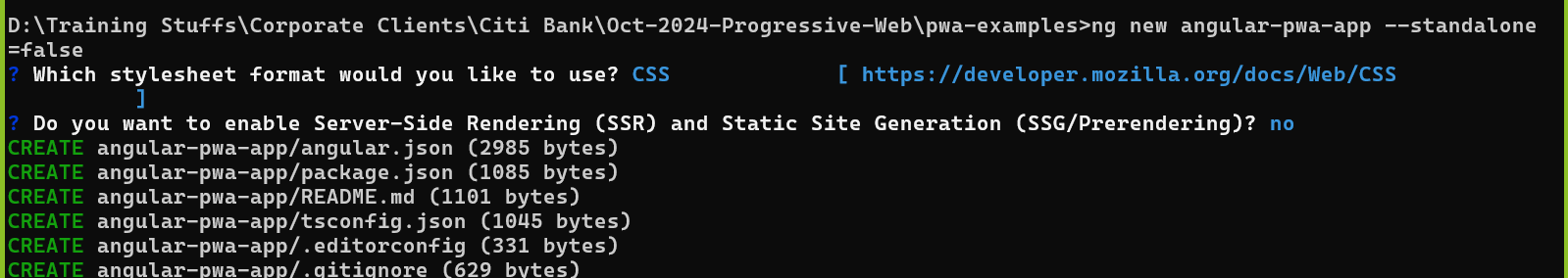


Verifying the installation



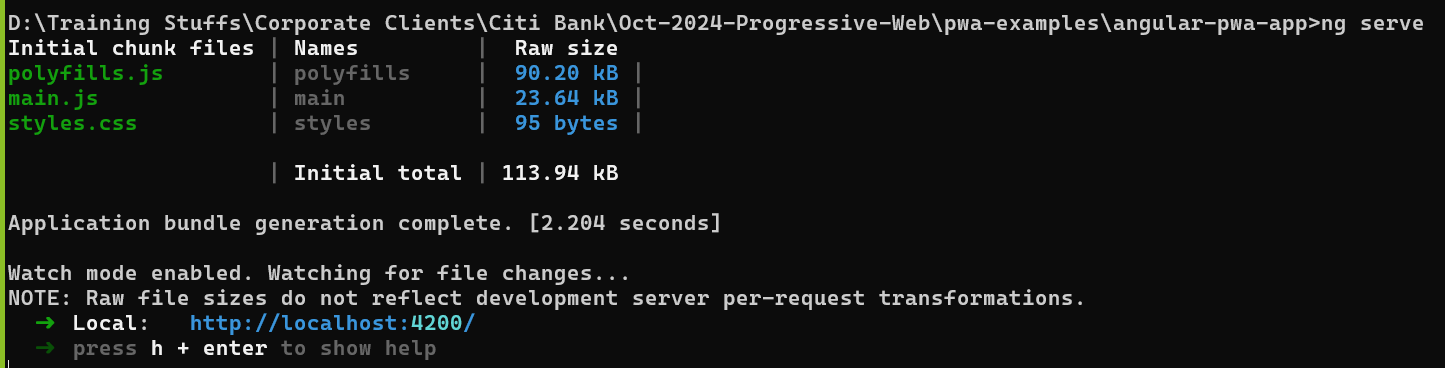
Angular Project:

ng new app-name –standalone=false

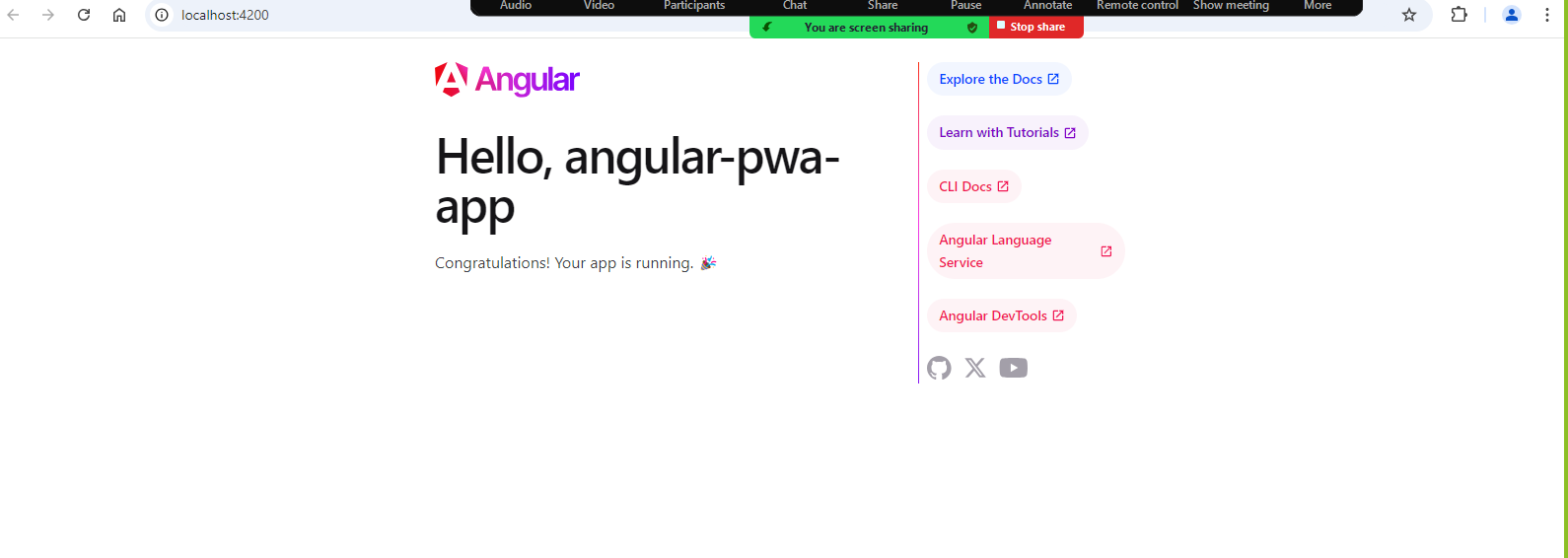


Change the directory to navigate to the project folder

Then use npm start or ng serve command to launch the application.



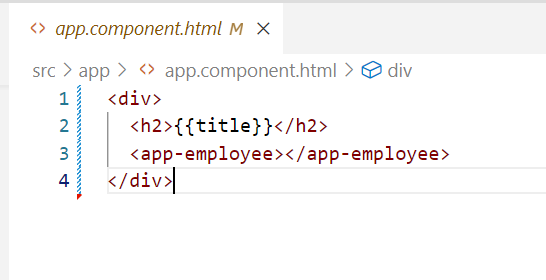
You can click on the link to view the page



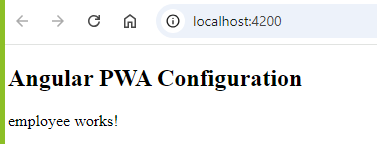
Generating the component

ng g c employee

Adding the employee component in app.component.html

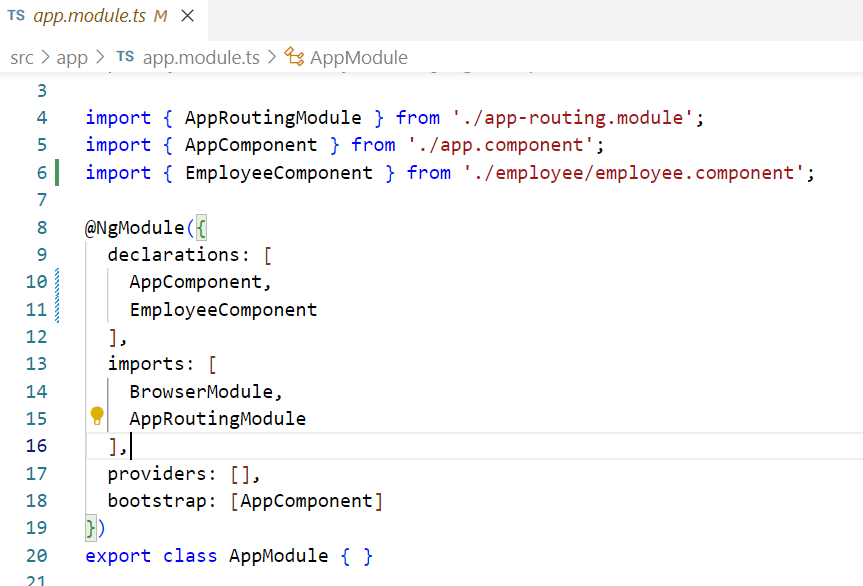


Output:



How to configure PWA in Angular

Angular Framework provides a library called @angular/pwa using which angular creates all the necessary files to PWA features & configures in the root module app.module.ts



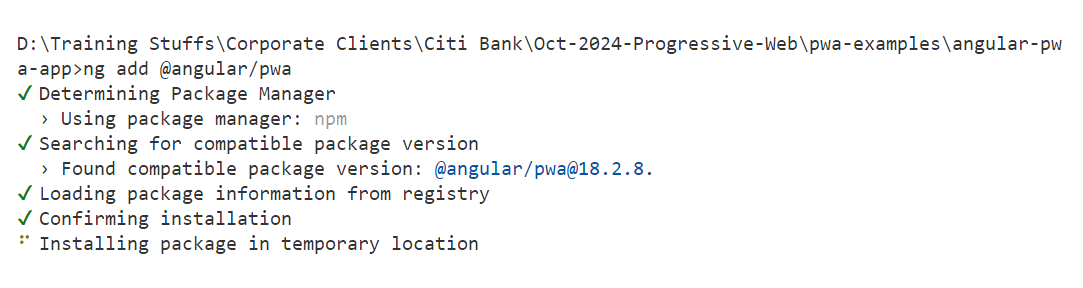
Currently the app.module.ts has no PWA feature, the moment you add @angular/pwa you get some code that specifies PWA is configured

Steps

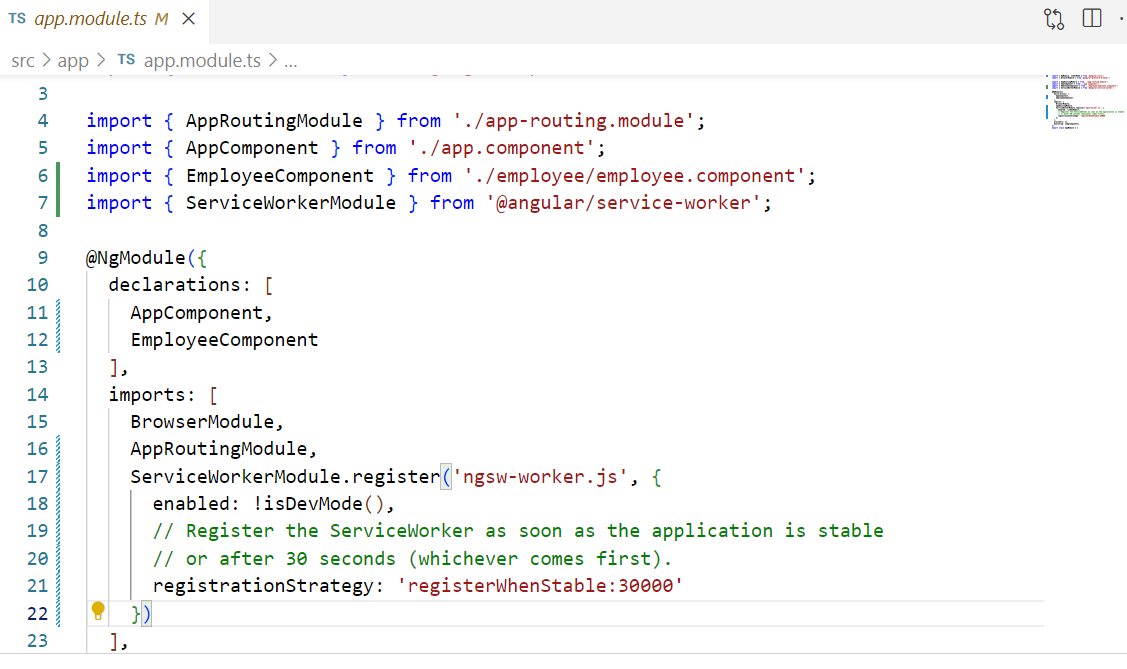
1. Adding PWA using ng add @angular/pwa

The above does all the things required to enable PWA like

* mainifest file: Defines PWA metadata like name, icons, start page
* ngsw-config.json: Configure angular service worker
* add necessary packages for service worker and updates angular.json



Updates app.module.ts to register the service worker file



index.html will be configured with manifest file

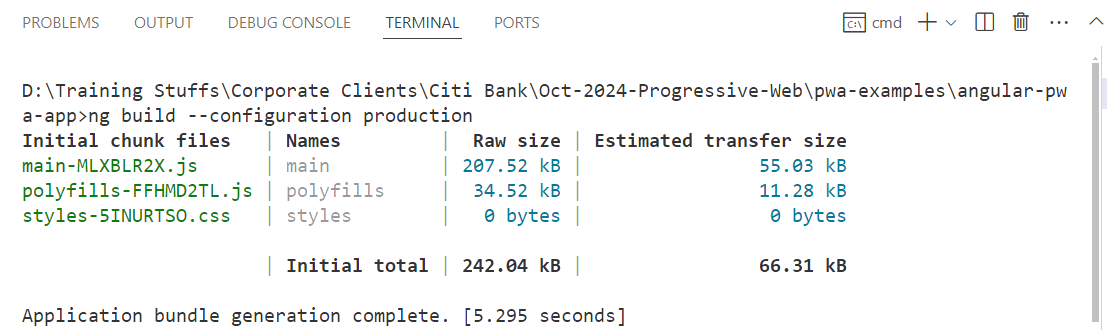


manifest.webmanifest will have PWA meta-data: That is referred by the index.html

Note: ng serve is used to launch the application, however service worker works only in real server, hence you must build the project & launch using any real server like http-server

What happens when you build

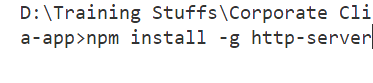
All the components are optimized in a single js file & it will be loaded in the index.html



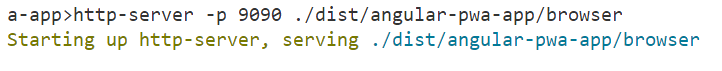
This project you built is inside dist folder of the app, which you can deploy in any server

We can install http server & deploy the dist folder

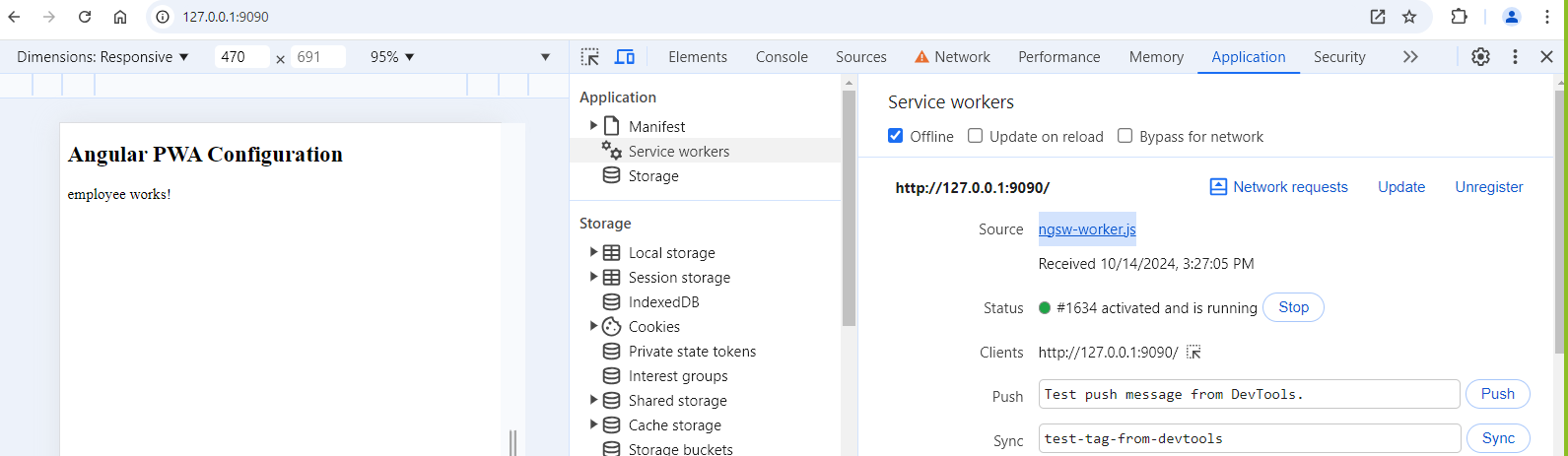
Installing http-server



Then you need to deploy dist/app-name/browser



Goto developer tools -> application-> you can observer service worker registered.



ngsw-worker.js is the service worker file that is registered

You can stop the server and try to access the web page, it will be still available

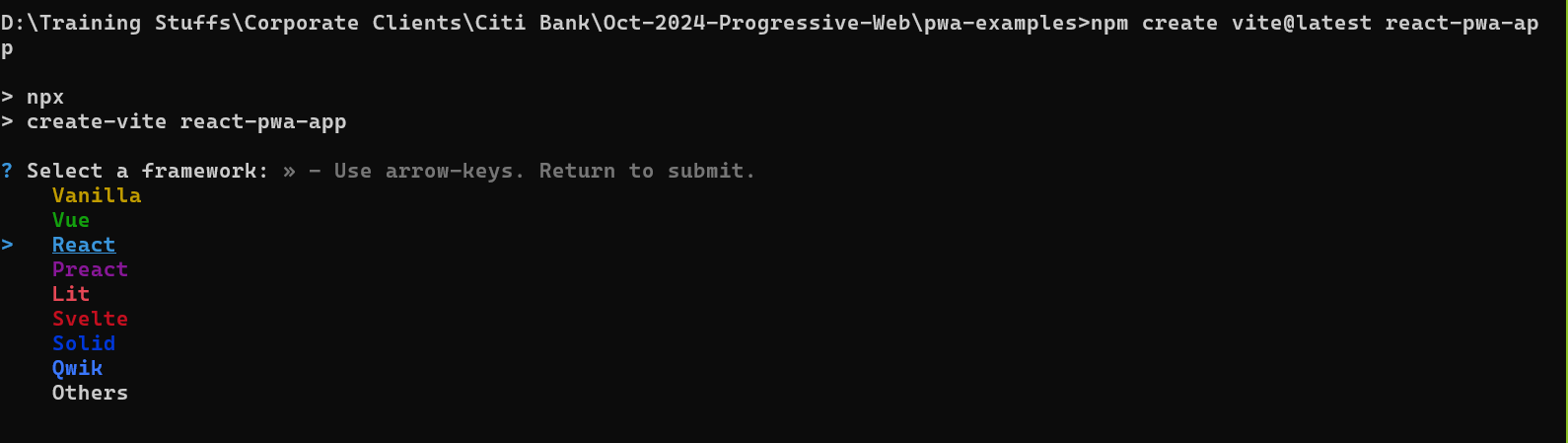
How to configure PWA in React

React application is created using same npm command, but it has a community called vite to develop the react application.

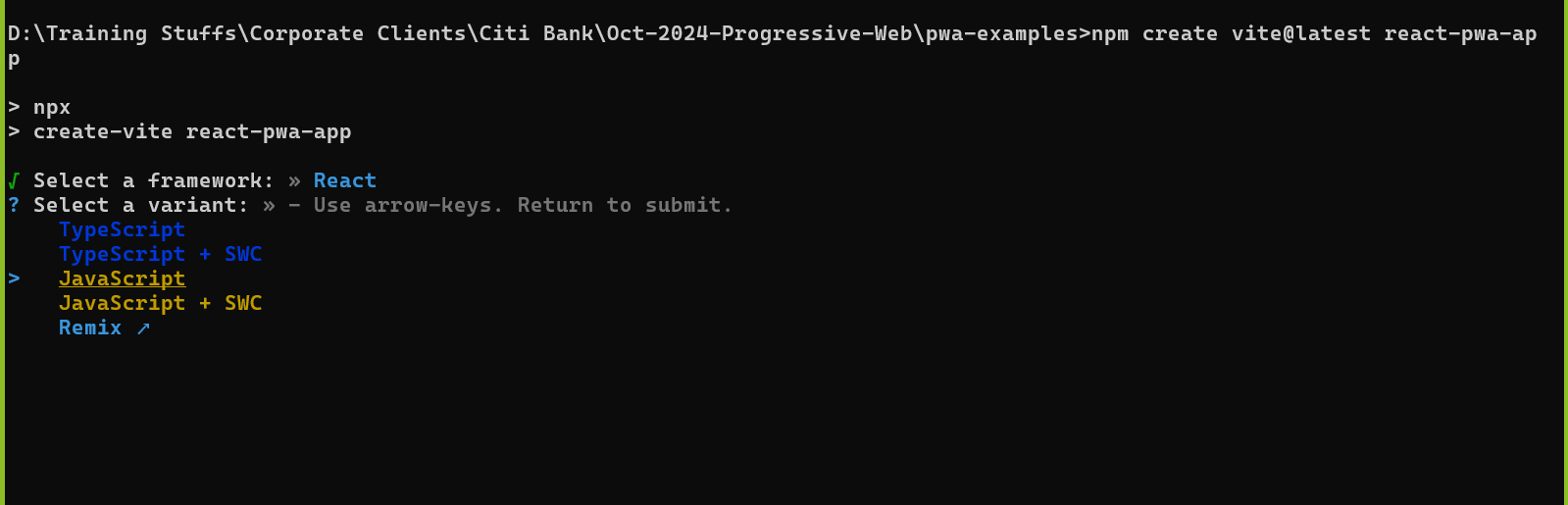
Steps

1. Create react project with vite: npm create vite@latest app-name
2. Navigate to the project
3. Install a plugin for pwa: npm install vite-plugin-pwa –save-dev
4. You must add some configuration code in vite.config.js file
5. Build the application
6. Through the server you must deploy the build

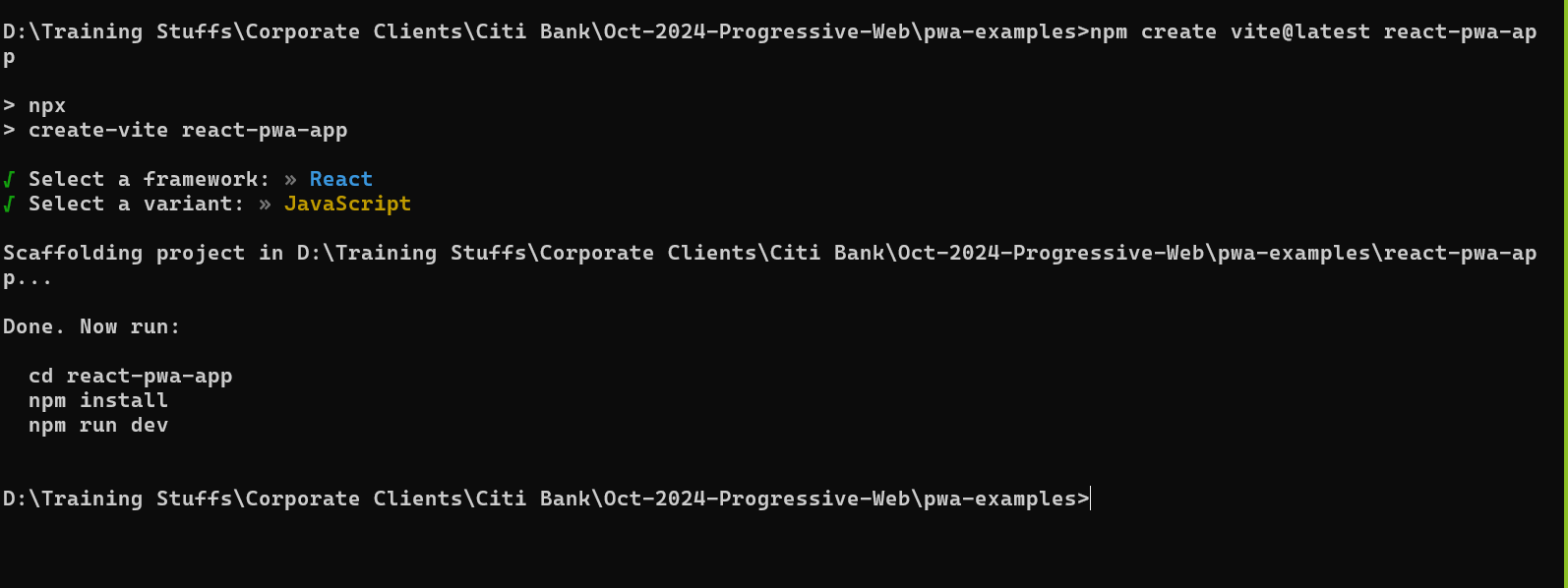
Creating react with vite



Select React

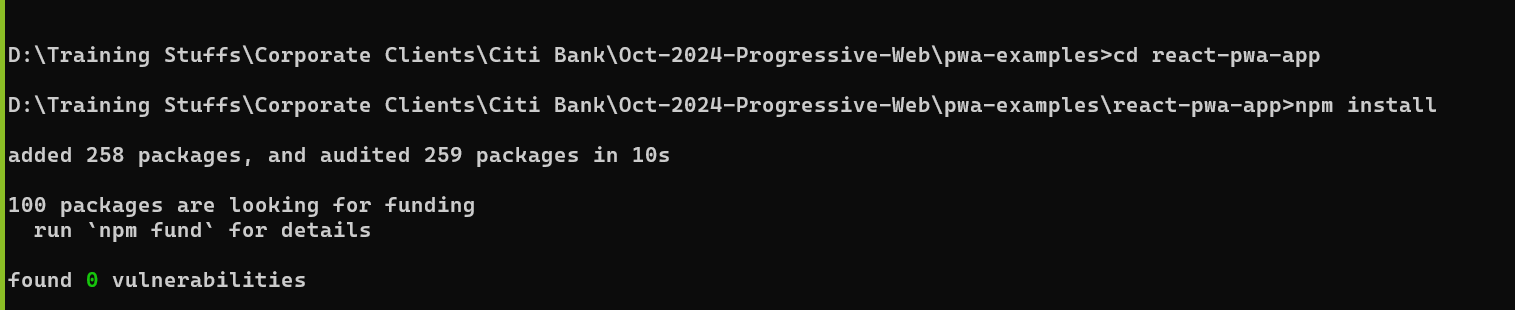


Select Javascript

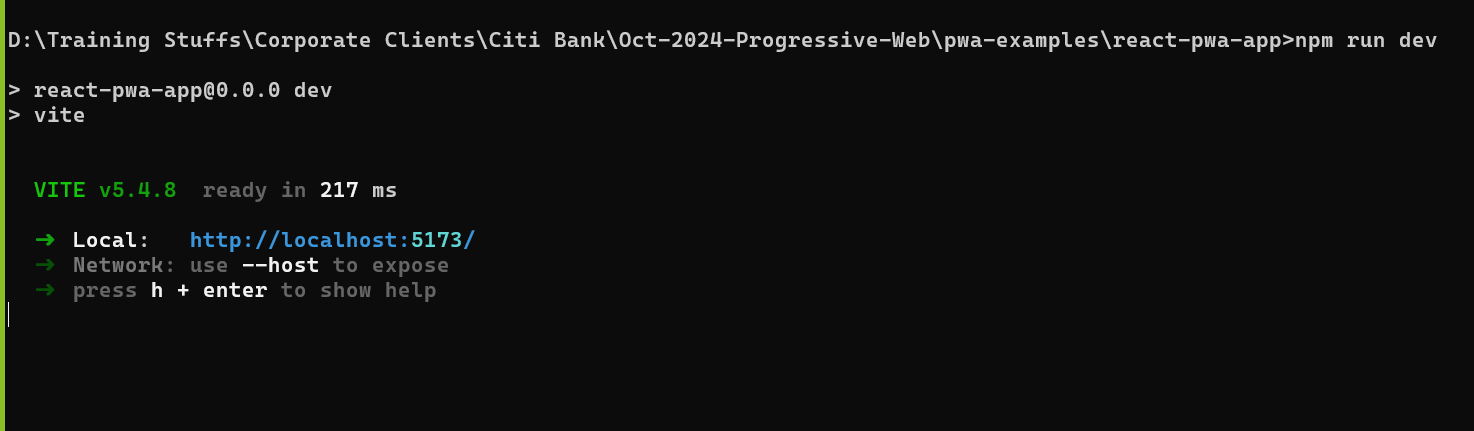


You will see the project created, you must navigate to the project & then install the libraries required for react

i.e., cd & npm install & then npm run dev



npm run dev



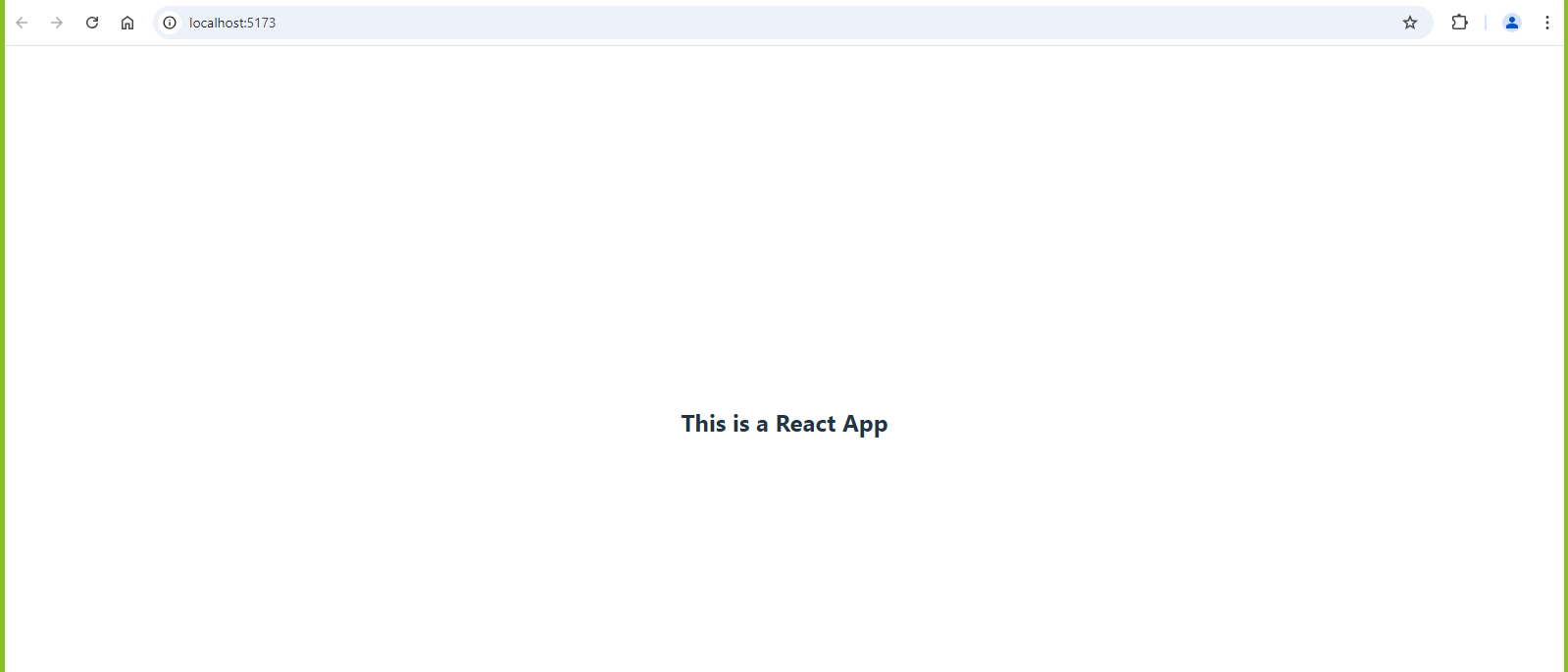
Stop the server & visit the page, where it says to internet connection

Go to the project/src/App.jsx

Add just an hello world message in the return statement of function App



re-run the npm run dev to see the above message

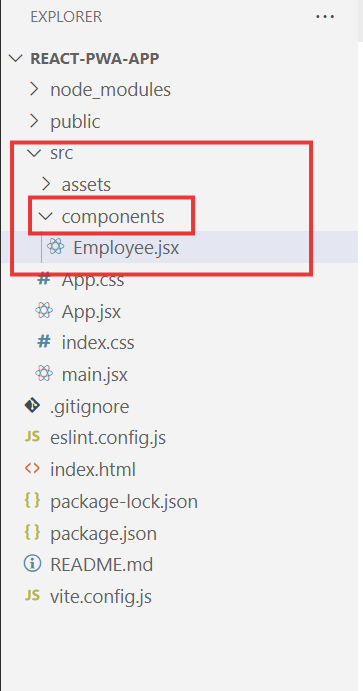


But the above app is still not configured for the PWA, we must add some configuration for the react app to enable PWA

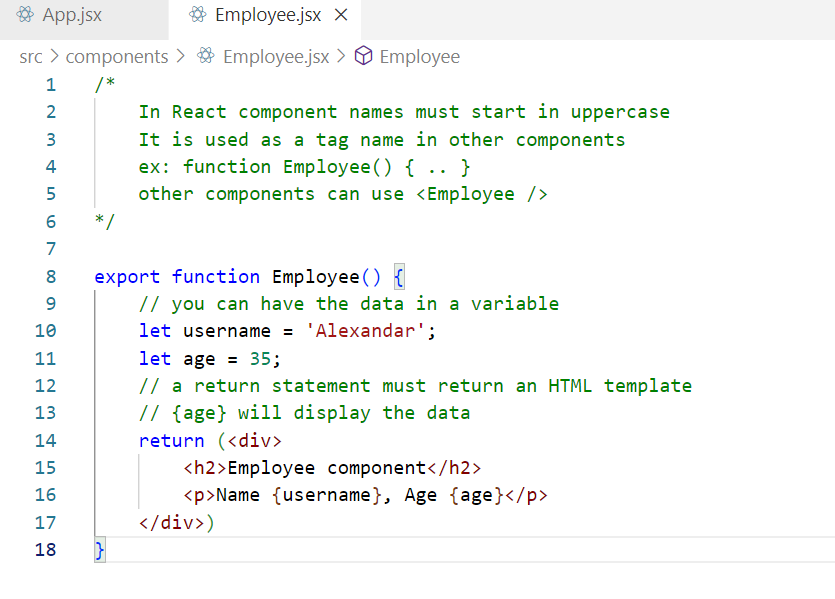
How to create a component?

Unlike Angular react doesn’t have any commands to create components you must create components using code itself.

src/components/Employee.jsx



Using function you can create a component in the Employee.jsx

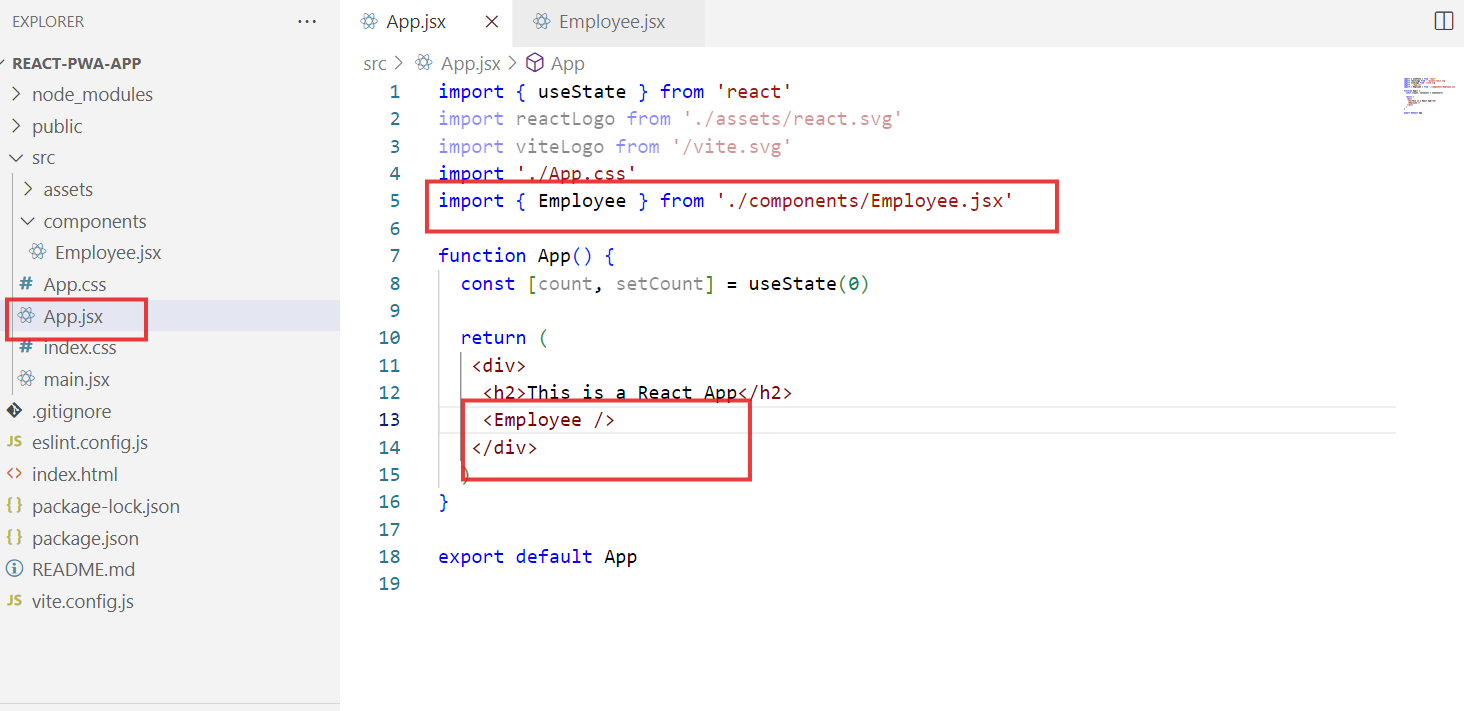


This is a child component which can be added in the root component

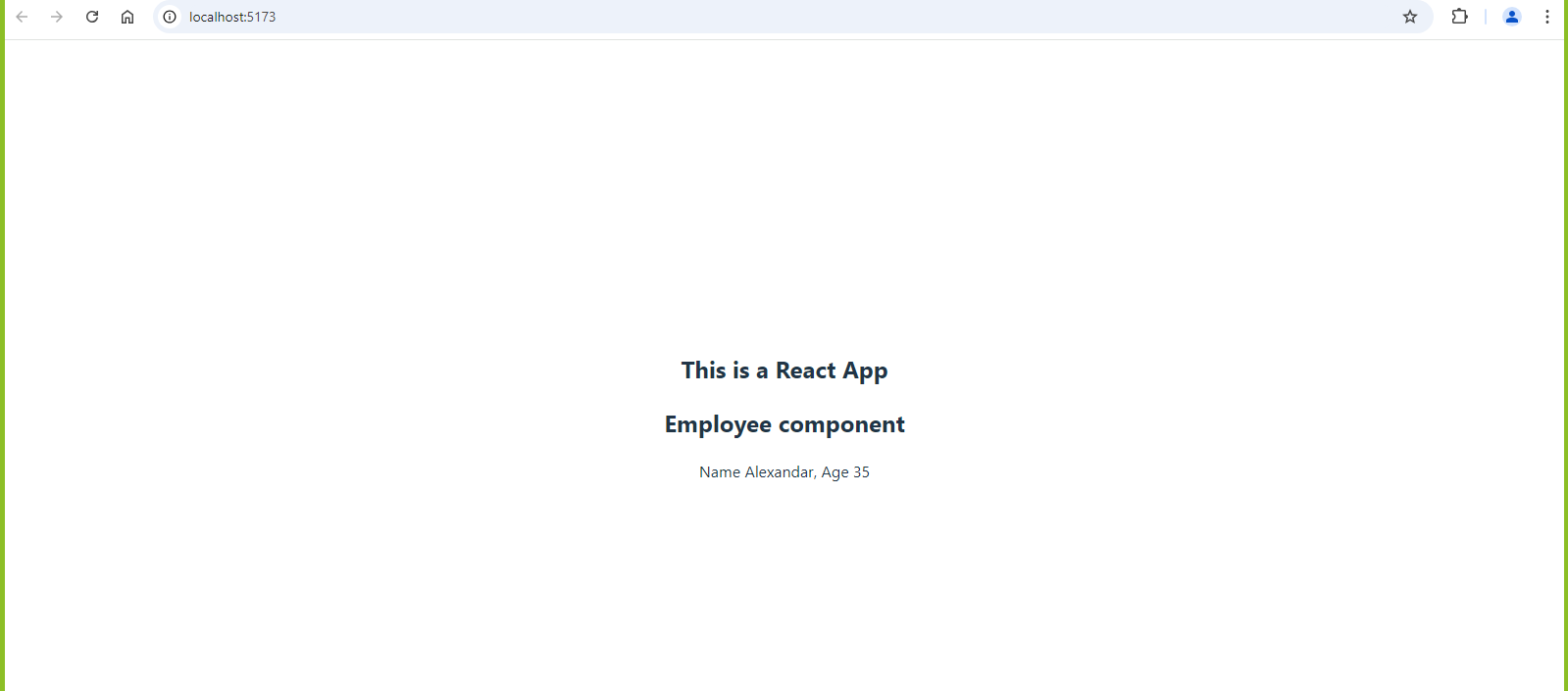
Note: App.jsx is the root component in React.js, however in Angular the root component was AppComponent

Let us add the Employee component into the root component

src/App.jsx



Output



Adding PWA feature to the existing react application

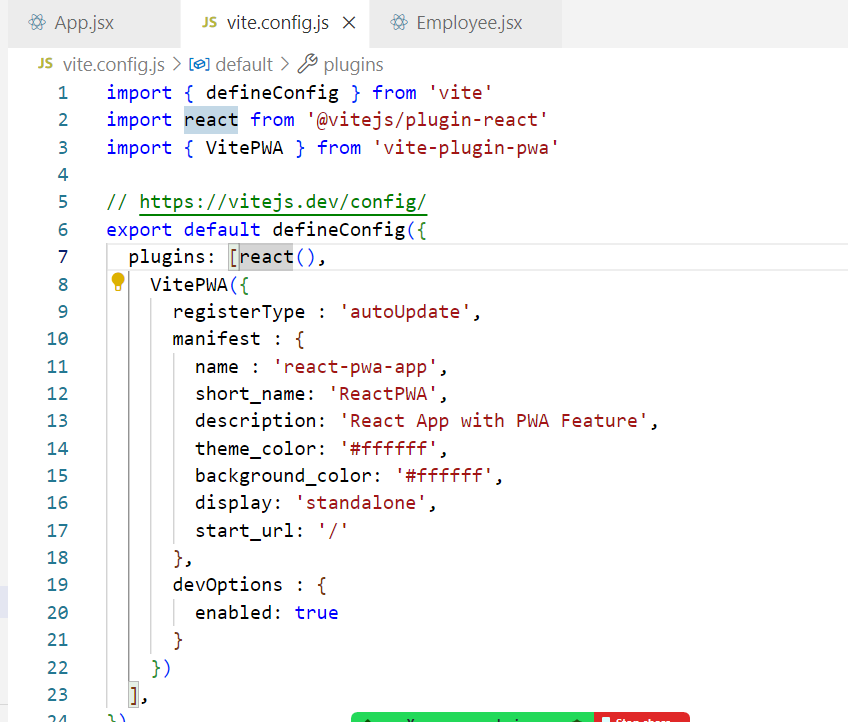
We need to perform 2 steps

1. Install vite plugin pwa
   1. npm install vite-plugin-pwa –save-dev
   2. configure vite.config.js to have pwa

Installing Vite Plugin PWA



Configure the PWA plugin in vite.config.js file

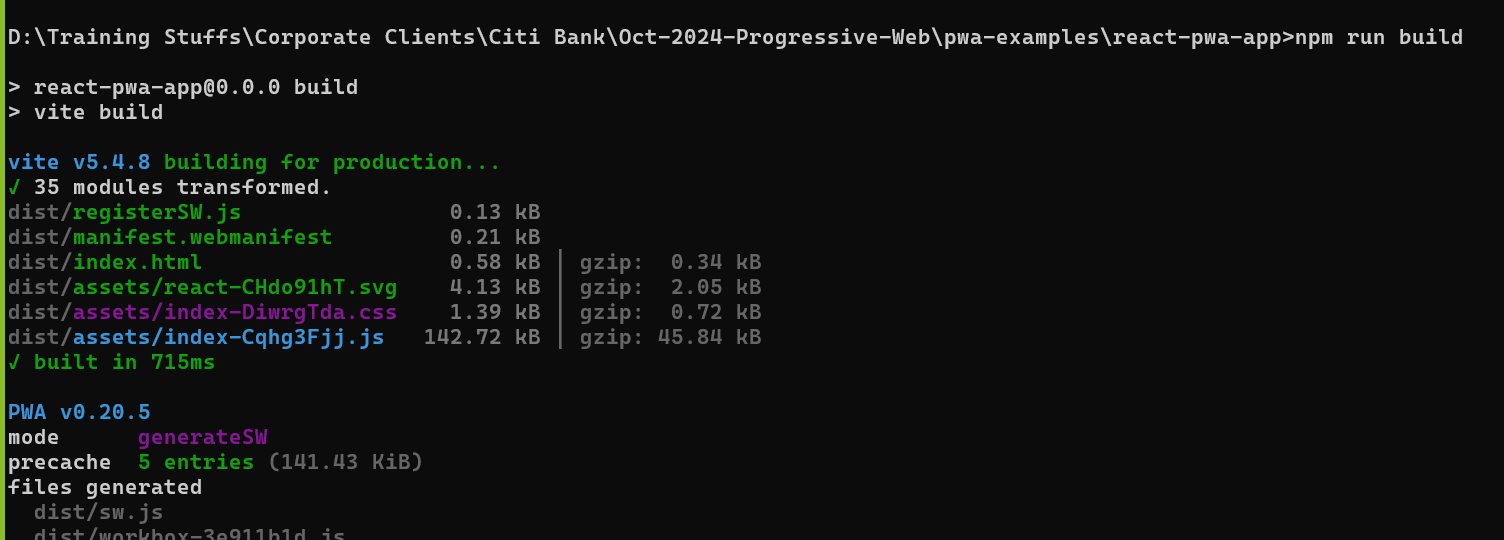


Now you must build the project & deploy the application in a real server, because service worker doesn’t work in the development server.

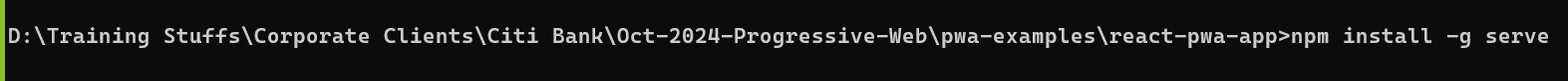
You can install a lite server called serve specifically to test react applications to mimic the production server.

Firstly we must build using npm run build command (creates a dist folder) , then install the server using npm install -g serve, then deploy using serve -s dist

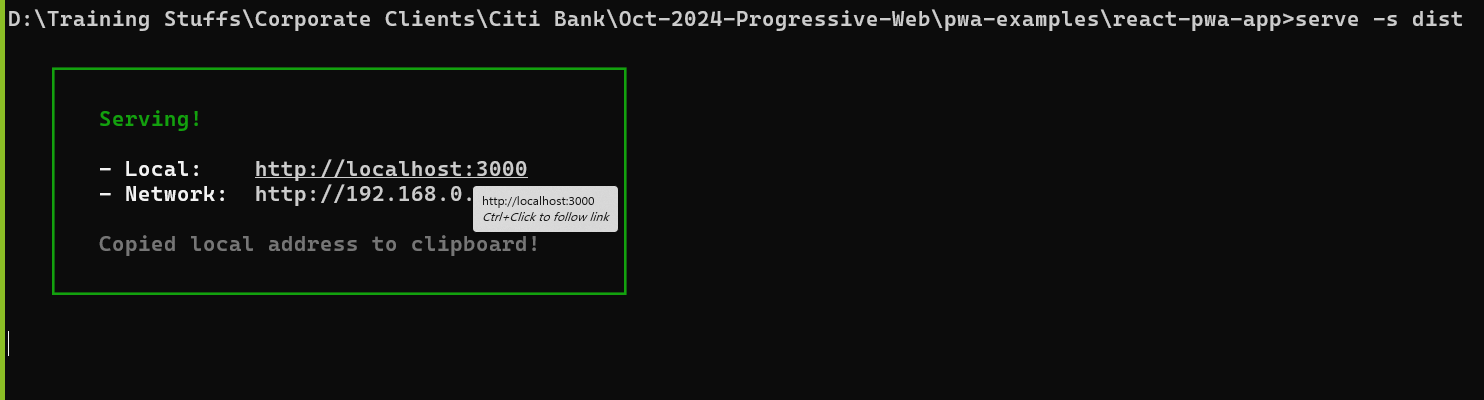
npm run build



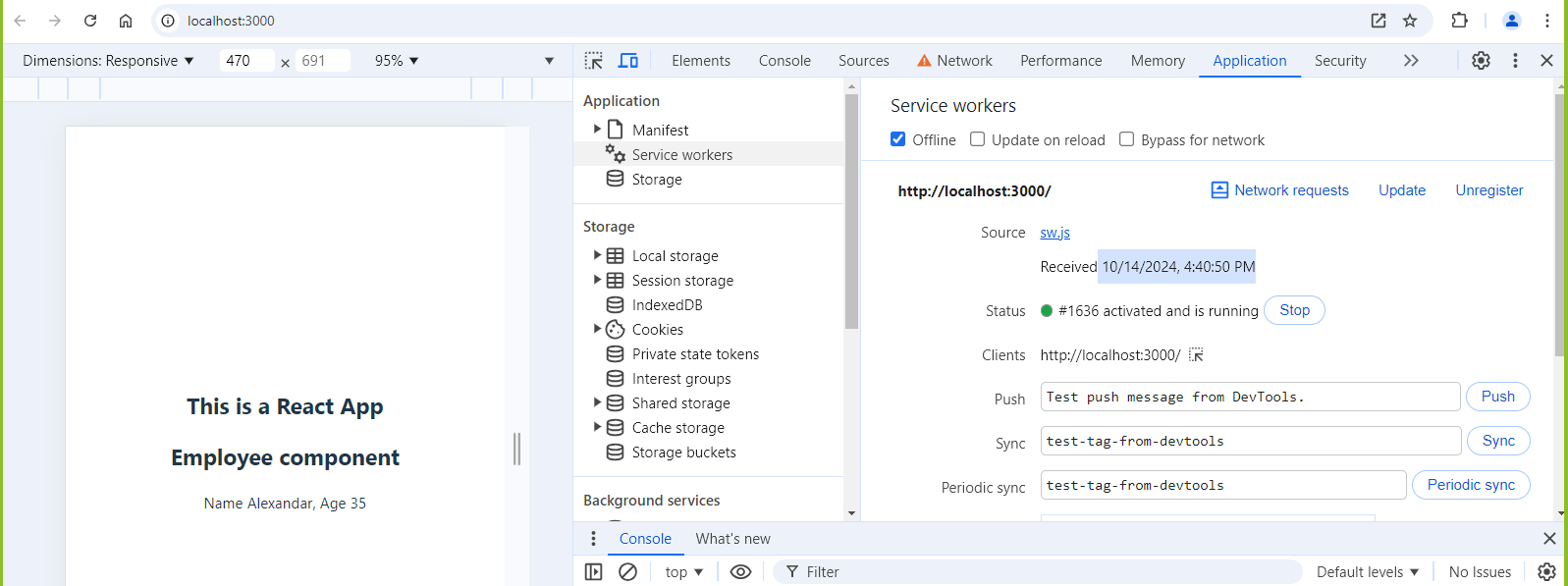
npm install -g serve



Deploy the dist



You can now see the application & also the service worker then offline features



You can still access the app even if you are offline

