**Cognizant Digital Nurture 4.0**

**Week-7**

**Name -Brishti Das**

**Superset id. -6363729**

**React Handson Exercise 9**

ES6 and JavaScript Concepts

# Features of ES6

* let and const: Block-scoped variable declarations.
* Arrow Functions (=>): Shorter function syntax and lexical `this`.
* Classes: Syntactic sugar over prototypes for OOP.
* Template Literals: ${} syntax for string interpolation.
* Destructuring: Unpack values from arrays/objects.
* Default Parameters: Set default values for function arguments.
* Rest and Spread Operators (...): Handle variable arguments and copy/merge arrays/objects.
* Modules (import/export): Modular coding system.
* Promises: Handle asynchronous operations.
* for...of Loop: Iterate over iterable objects.
* Map and Set: New data structures.
* Symbol: Unique and immutable primitive value.

# JavaScript let

* let is a keyword used to declare block-scoped variables.
* Cannot be re-declared in the same scope.
* Can be updated, but not re-declared.
* Example:  
  let x = 10;  
  if (true) {  
   let x = 20;  
   console.log(x); // 20  
  }  
  console.log(x); // 10

# Difference between var and let

* var is function-scoped; let is block-scoped.
* var allows re-declaration; let does not in the same scope.
* var is hoisted and initialized with undefined; let is hoisted but not initialized (Temporal Dead Zone).
* var creates properties on the global object; let does not.

# JavaScript const

* const declares block-scoped, read-only constants.
* Cannot be re-declared or re-assigned.
* Must be initialized at declaration.
* Objects and arrays declared with const can have their contents modified.

# ES6 Class Fundamentals

* Classes are syntactic sugar over JavaScript’s existing prototype-based inheritance.
* Syntax:  
  class Person {  
   constructor(name) {  
   this.name = name;  
   }  
   greet() {  
   console.log(`Hi, I'm ${this.name}`);  
   }  
  }

# ES6 Class Inheritance

* Use extends and super() to inherit from another class.
* Example:  
  class Dog extends Animal {  
   speak() {  
   console.log(`${this.name} barks.`);  
   }  
  }

# ES6 Arrow Functions

* Shorter syntax for writing functions.
* Do not have their own 'this'.
* Cannot be used as constructors.
* Do not have 'arguments' object.
* Example: const add = (a, b) => a + b;

# Set and Map in ES6

* Set: Collection of unique values. No duplicates allowed.
* Map: Collection of key-value pairs. Keys can be any type.
* Set Example:  
  const mySet = new Set();  
  mySet.add(1);  
  mySet.add(2);  
  mySet.add(1); // ignored
* Map Example:  
  const myMap = new Map();  
  myMap.set('name', 'Alice');  
  myMap.set(100, 'Number Key');