Backend Bootcamp - Session 2: JavaScript Refresher

Why JavaScript?

- Used in both frontend and backend (Node.js)
- Universal language for fullstack apps
- Event-driven, asynchronous support
- Large ecosystem (npm, frameworks, tooling)

Scopes and Hoisting

- let and const are block scoped
- var is function scoped (avoid it)
- Hoisting: Declarations are moved to top

```js
console.log(a); // undefined
var a = 5;
```

Data Types and Type Coercion

- Primitive: string, number, boolean, null, undefined, symbol
- Reference: object, array, function
- Type coercion:

```js
'5' + 1 // '51'
'5' - 1 // 4
true + 1 // 2
```

Functions and Closures

- Functions are first-class citizens
- Closures: inner function remembers variables from outer function

```js
```

```
function outer() {
  let count = 0;
  return function inner() { count++; return count; }
}
const inc = outer();
inc(); // 1
```

this Keyword and Arrow Functions

- 'this' refers to execution context
- Arrow functions do not bind 'this'

```js
const obj = {
  name: 'Ali',
  greet() {
    setTimeout(() => console.log(this.name), 1000);
  }
};
obj.greet();
```

Array Methods Deep Dive

- map, filter, reduce, find, every, some, sort

```js
const arr = [5, 2, 8];
arr.sort((a, b) => a - b); // ascending
```

Destructuring, Spread & Rest

- Extract values and copy/extend objects/arrays

```js
const { name, ...rest } = { name: 'Sara', age: 25 };
const clone = [...[1,2,3]];
```

Async JavaScript Advanced

- Promises, async/await, error handling

```js
async function fetchUser() {
 try {
  let res = await fetch(url);
  let data = await res.json();
 } catch (e) {
  console.error(e);
 }
}
```

Class and Prototypes

- Classes use prototype-based inheritance

```js
class User {
  constructor(name) { this.name = name; }
  greet() { return `Hello ${this.name}`; }
}
```

Short Quiz (MCQs)

Backend Bootcamp - Session 2: JavaScript Refresher

1. What does 'const' mean?
   a) Reassignable
   b) Immutable binding
   c) Block scoped only
   d) Function scoped

2. What will [1,2,3].map(n => n * 2) return?
   a) [1, 2, 3]
   b) [2, 4, 6]
   c) undefined
   d) Error

3. What does async/await help with?
   a) Styling
   b) Asynchronous operations
   c) Array manipulation
   d) Loops

4. What does filter() do?
   a) Transforms each item
   b) Removes duplicates
   c) Filters based on condition
   d) Sorts elements

5. What is a closure?
   a) A loop scope
   b) An object inside a class
   c) A function remembering its parent scope
   d) A method that returns a promise

6. What does [...arr] do?
   a) Creates a new array clone

b) Filters nulls

c) Combines arrays

d) Reverses array

7. What is 'this' in an arrow function?

a) Refers to global scope

b) Bound to current function

c) Lexically scoped

d) Undefined by default

8. Which of these will throw an error?

a) let x = 5; x = 10;

b) const x = 5; x = 10;

c) var x = 5;

d) let x;

9. What does reduce() do?

a) Combines array values into one

b) Filters elements

c) Sorts values

d) Reverses array

10. What is prototype in JS?

a) A class method

b) Shared property chain for inheritance

c) Array reference

d) Keyword for scope