1: What are the differences between var, let and const?

JavaScript allows you to declare variables using one of three different keywords, var, let, and const. While all these allow you to declare variables, there are some differences in how each of these keywords are used.

“Var” is a keyword that is globally scoped, meaning the variable can be initialized and used anywhere throughout the document. Whereas “let” is a more narrowly scoped keyword. Let is typically used within a given block of code , such as a function or a loop, making the variables locally scoped. Meaning that any variables declared by using let within that code block are limited to that code block. You could set a variable by writing let sum = …. Inside a function and that variable would be limited to only that function. However, you could also have a variable with the same name elsewhere in the document with a global scope. Each declaration would hold their own unique values. The locally scoped variable would remain and be used inside of that function, loop, if statement, etc. But the globally scoped variable could be used in other algorithms in other areas of the document.

The “const” keyword indicates a constant variable. That is the value assigned to that variable cannot be changed later in the code base. Const is used for values that need to remain unchanged, such as a birthday.

Source: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/let

2: What are some of the features that are new in JavaScript with ES6?

The latest revision of the JavaScript language brings many enhancements and new functionalities. Many of these can be used to create shorter, more effective code when developing applications or web pages. Others create better ways of getting something done through your code. The following is a list of most of the new features included in the ES6 update:

- arrows
- classes
- enhanced object literals
- template strings
- destructuring
- default + rest + spread
- let + const
- iterators + for..of
- generators
- unicode
- modules
- module loaders
- map + set + weakmap + weakset

- proxies
- symbols
- subclassable built-ins
- promises
- math + number + string + array + object APIs
- binary and octal literals
- reflect api
- tail calls

Source: https://github.com/lukehoban/es6features