Summary of useful ES8+ features for Backend Development

Here I’ll be listing the many JS ES8+ features that I think are useful in backend development.

1. Async/Await (ES8)
   1. This allows for writing of asynchronous code in a more concise and readable manner, by replacing the regular .then() and .catch() methods with the simple try/catch syntax.
2. Object.values() and Object.entries() (ES8)
   1. The addition of these two methods allowed for easier access to an object’s inner contents without having to write custom access logic for such scenarios.
   2. The Object.values() returns an array of the object’s values.
   3. The Object.entries() returns an array of a given object’s key-value pairs.
3. Rest/Spread Properties (ES9)
   1. Rest properties allow the collection of remaining properties into a single object.
   2. Spread properties allow the merging and copying of objects
4. Promise.finally() (ES9)
   1. Similar to how the finally block works in a try/catch code structure, the finally() method allows for adding a cleanup logic after the promise is settled (either resolved or rejected).
5. Optional Catch Binding (ES10)
   1. This features allows us to omit the error parameter in the catch block if it’s not required.
6. Nullish Coalescing Operator (??) (ES11)
   1. This provides an easy way to assign default values when null or undefined values are encountered.
7. Optional Chaining Operator (?.) (ES11)
   1. This allows for safe access to nested properties without throwing errors if a reference is null or undefined.
8. Dynamic Imports (ES11)
   1. Dynamic imports feature provides performance improvements by allowing the loading of modules only when needed, instead of preloading all the required modules.
9. Logical Assignments Operators (ES12)
   1. This allows the combination of logical operations with assignment in a more concise manner.
   2. These operator makes the code more readable when assigning values based on logical conditions.
10. BigInt (ES11)
    1. A new primitive type which allows for handling large integers beyond Number type’s limit.