Node.js handles asynchronous operations through the libuv library. By default, the libuv provides four worker threads for Node.js that are only used when blocking asynchronous operations are performed. Node.js uses callbacks which are a function passed into another function as an argument, which is then invoked inside the outer function to complete some kind of routine or action. In addition, Node.js uses promises which are a proxy for a state or value that is unknown for the moment when the promise is created but could be determined later. Finally, Node.js uses async-await which is a method in which the parent function is declared with the async keyword and inside it await keyword is permitted. The async and await keyword enables asynchronous, promise-based behavior so that code could be written a lot cleaner and avoid promise chains.

Asynchronous operations are beneficial because they allow things to happen independent of the main program flow. For instance, this allows the program to load large files while other lines of code are running. In general asynchronous operations are beneficial because it allows improved application performance because it allows tasks to run together, avoiding delays. It allows for enhanced scalability, by spreading work across many servers or processes. It allows for better user experience by making the user interface more responsive and interactive because it allows users to continue using the app while background tasks run. Furthermore, allows for resource optimization by ensuring efficient use of system resources by avoiding idle periods and maximizing resource use.