**process The** **ProcessObject in Node.js refers to the process object. It is a global object that provides information and control over the current Node.js process. It has various properties and methods that you can use to interact with the process. \*The process object allows you to access information about the command-line arguments, environment variables, process ID, current working directory, and more. It also provides methods for controlling the process, such as terminating the process or scheduling callbacks for specific events. \* By using the process object, you can have more control and flexibility in managing your Node.js applications.**

**Some important properties of the process object include:**- **process.argv: An array that contains the command-line arguments passed to the Node.js process.- process.env: An object that contains the user environment.- process.pid: The process ID of the current Node.js process.- process.cwd(): Returns the current working directory of the Node.js process.- process.platform: Returns a string indicating the operating system platform**. **Some useful methods of the process object include:- process.exit([code]): Terminates the Node.js process with an optional exit code.- process.on(event, callback): Registers a callback function for a specific event, such as 'exit' or 'uncaughtException'.- process.nextTick(callback): Schedules a callback function to be invoked in the next iteration of the** **event loop.**

**• Buffer The buffer module provides a way of handling streams of binary data it works with that data directly in memory. a buffer is a built-in object .The Buffer object is a global object in Node.js, and it is not necessary to import it using the require keyword. Uses of Buffer 1. Accessing Buffer Data: Buffers are essentially arrays of integers, representing bytes. You can access individual bytes using array notation. 2. Manipulating Buffers: The Buffer module provides various methods for manipulating buffer contents, such as copying, slicing, and filling. 3.Converting Buffers: Buffers can be converted to different encodings, such as UTF-8 or Base64, using the toString() method. 4.Buffer and Streams: Buffers are commonly used in Node.js when working with streams. For example, when reading from a file or receiving data from a network, data is often read into or written from a buffer. • Properties & Methods of Buffer 1. alloc(): allloc is used to define a specific size to a buffer object. Syntax: Buffer.alloc(size, fill); 2. bytelength(): byteLength() method returns the length of a specified string object, in bytes. Syntax: Buffer.byteLength(string); 3. compare() : Compares two Buffer objects. Syntax: Buffer.compare(buf1, buf2); 4. copy():The copy() method copies data from one buffer object into another buffer object. Syntax: buffer.copy(target, targetStart, sourceStart, sourceEnd); 5. equals() : Compares two Buffer objects, and returns true if it is a match, otherwise false Syntax: buffer.equals(buf);**