**Q1. What is Node.js? Where can you use it?**

**⇒**Node .js is an open-source, cross-platform runtime environment that allows developers to create all kinds of server-side tools and applications in JavaScript.

**⇒**It is used for server-side programming, and primarily deployed for non-blocking, event-driven servers, such as traditional web sites and back-end API services, but was originally designed with real-time, push-based architectures in mind. Every browser has its own version of a JS engine, and node.

**⇒USE:**

Node.js can generate dynamic page content

Node.js can create, open, read, write, delete, and close files on the server

Node.js can collect form data

Node.js can add, delete, modify data in your database.

**Q2. Explain callback in Node.js.?**

**⇒**In Node. js, when a function starts reading a file, it returns the control to the execution environment immediately so that the next instruction can be executed. Once file I/O gets completed, the callback function will get called to avoid blocking or waiting for File I/O.

**Q3. What are the advantages of using promises instead of callbacks?**

**⇒**To implement asynchronous code in JavaScript we use callback functions and promises. A callback function is passed as an argument to another function whereas Promise is something that is achieved or completed in the future.

**⇒advantages of using promises**

Improves Code Readability.

Better handling of asynchronous operations.

Better flow of control definition in asynchronous logic.

Better Error Handling.

**Q4. What is NPM**?

**⇒**NPM stands for **Node Package Manager**

NPM is a package manager for the JavaScript programming language maintained by npm, Inc. npm is the default package manager for the JavaScript runtime environment Node. js

**Q5. What are the modules in Node.js? Explain**

**⇒Types of Modules:**

Core Modules

Local Modules/own Modules

Third-Party Modules/NPM module

**⇒Core Modules:** Node.js comes with dozens of built-in modules. These built-in modules are sometimes referred to as core modules. The module system is built around the required function. This function is used to load a module and get access to its contents. require is a global variable provided to all your Node.js scripts, so you can use it anywhere you like. require() function will return a JavaScript type depending on your module.

**Syntex:**

Const module = require(“name of module);

**⇒Local Modules/own Modules:** Putting all code in a single file is not a good idea. If you add more code, you’ll want to stay organized and break your Node.js app into multiple scripts that all work together. For that purpose, we can create local modules in our application.

**Exporting from files:**

utils.js

Const test = () =>{

console.log(“test function”);

}

module.export={

test

}

**Importing files:**

const utility = require('./utils.js');

utility();

**⇒Third-Party Modules:** Third-party modules can be installed from the NPM (Node Package Manager) available online. Firstly we need to initialize the npm using the npm init command before npm can be used. It creates a package.json file in the root directory and it stores all the information about the third-party module that we have installed as a dependency

**Syntex:**

npm install "module\_name"