1. Explain what is an error first callback? Please provide a coded example.

An error-first callback is a function in Node.js that determines if an execution is in error or successful, and the function returns either an error variable or the data variable as a result of the execution. The great thing about an error-first callback function is that you don't need to know or program for the exact list of possible error codes. If any error results from the program execution, the error-first callback will capture and display it to the user. It's a great way to reduce the amount of Node.js programming code needed for error processing. The syntax of the error-first callback is as follows:

const ErrorFirstCallback = (error, data) =>

{

if (error)

{

return console.log(error);

}

console.log("Execution was successful");

};

A good programming example of an error-first callback would be reading and/or processing a file from the file system. The syntax of the operation would look something like this:

file.readFile(inputFile, ErrorFirstCallBack);

If the file is not found or you don't have permission to access it or the execution fails midway through, all of these situations have different error codes that will be captured and displayed.

2. Explain what Promises are. Please provide a coded example.

A Promise in Node.js is an asynchronous callback function that results in a return value which is then supplied as a result of the promise's execution. This is unlike normal callback functions that do not have a return value. The promise has a 'then' clause with either two return values normally denoted as 'fulfilled' or 'rejected', and a Promise is typically executed in the background and retuns the execution results when completed. The syntax of a promise is as follows:

var Promise = require('promise');

promise.then(fulfilled, rejected)

A programming example could be connecting to the EmployeeDB and then inserting employees after validating that the username is properlyformatted:

var Promise = require('promise');

MongoDB.connect(EmployeeDB)

.then(valUserCreds(username)

{

if (substr(username,1)=='empl')

{

resolve(db.insert(username))

}

else

{

reject("Username is not properly formatted!")

}

}

3. What is a test pyramid? Please give an example.

A test pyramid in Node.js is essentially a framework that applies to both developers and testers to ensure that high-quality software is developed through proper testing at unit testing (lowest), integration testing (midde), and end-to-end testing (highest) levels. The theory behind a test pyramid is that there should be the most unit tests to cover all software functionality followed by a lesser number of integration tests to cover more specific software functionality that may span other code in the application followed by a lesser number of end-to-end tests to cover the entire software functionality. In this example, there may be 50 unit tests, 20 integration tests, and 5 end-to-end tests for an application.

A picture containing text, screenshot, font, logo

Description automatically generated

4. Using the vulnerability list from OWASP as mentioned in a previous module, research one of the top 2011 vulnerabilities and how the vulnerability could be found in node.js/express.js environment and application.

SQL Injection is a vulnerability that is fairly common when presented with a webpage that has input fields. SQL Injection occurs when a user enters HTML or javascript into an input field instead of valid data, and the HTML or javascript is placed into the database and/or processed by the client or server-side processing. The danger of SQL Injection is that a jackal can attempt to query data from the database or place HTML or javascript into the database record so when the record is called back up, it may attempt to execute the HTML or javascript code between <html></html> tags as if it were a valid part of the webpage launch. A jackal can also enter " or ""=" into userid and password combinations in an attempt to trick or bypass the validation routine to gain access to the system's userids and passwords.

In order to protect against SQL Injection, developers need to add very robust validation routines to their programs to ensure the proper type, format, and length of data is being submitted. For example, if a userid is a standard length of ten characters beginning with 'empl' followed by six numbers, the userid validation routine should only accept data that matches this profile. All other data inputted should be rejected.

5. What roles does NIST (National Institute of Standards and Technology) play in the world of cybersecurity and making sure software is secure?

The National Institute of Standards and Technology (NIST) is a division of the U.S. Department of Commerce that has developed a cybersecurity framework that applies to all businesses, regardless of industry or cyber maturity, to better understand, manage, and reduce cybersecurity risk and protect their networks and data. Their website (https://www.ftc.gov/business-guidance/small-businesses/cybersecurity/nist-framework) outlines the voluntary framework to 1. Identify, 2. Protect, 3. Detect, 4. Respond, and 5. Recover from cyber attacks. Not everyone who is opening a business with e-commerce over the internet is cyber savvy, so the NIST's cybersecurity framework is a great starting point for new and a friendly reminder for existing businesses alike.