Node.js Course Plan with Assignments

1. Introduction to Node.js

Topics Covered:

- Understanding Node.js and its role in web development
- Setting up the development environment (Node.js, npm, IDE)
- Event-driven, non-blocking architecture
- Basic syntax of JavaScript in Node.js
- Installing Node.js and setting up the environment

Exercise:

- Create a simple Node.js application that prints "Hello, world!" to the console.

2. JavaScript Fundamentals

Topics Covered:

- Variables, data types, and operators
- Control flow (if-else statements, loops)
- Functions and scope
- Arrays and objects
- Asynchronous programming with callbacks

Exercise:

- Write a Node.js script that reads a file asynchronously and writes the content to a new file, ensuring each line is on a new line.

3. Events and Callbacks

Topics Covered:

- Understanding events and callbacks
- Using the EventEmitter class
- Implementing event-driven programming

Exercises:

1. Create an event emitter that emits a 'start' event and logs a message when triggered.
2. Implement a function that accepts a callback and invokes it after a 1-second delay.
3. Create a function that takes two numbers and a callback, returning their sum.

4. Node.js Modules and NPM

Topics Covered:

- Creating and using modules
- Working with built-in Node.js modules
- Installing and using third-party packages from npm

Exercises:

1. Create a simple module that calculates the factorial of a number.
2. Build a Node.js application utilizing multiple modules and npm packages.

5. Express.js Framework

Topics Covered:

- Introduction to Express.js
- Building RESTful APIs
- Handling routes, middleware, and request/response objects

Exercises:

1. Create a simple API for a to-do list application.
2. Develop a RESTful API using Express.js to perform CRUD operations on a resource.

6. Database Integration

Topics Covered:

- Introduction to databases (MongoDB, MySQL)
- Connecting Node.js with databases
- Querying and manipulating data

Exercises:

1. Create APIs to store and retrieve data from a MongoDB database.
2. Build a Node.js application that connects to a database and performs CRUD operations.

7. Authentication and Authorization

Topics Covered:

- Implementing authentication with Passport.js or JWT
- Managing user sessions and cookies
- Role-based access control

Exercise:

- Add authentication and authorization to an existing Node.js application.

8. Error Handling and Logging

Topics Covered:

- Handling errors and exceptions
- Logging with Winston or Bunyan

Exercises:

1. Enhance an existing application with proper error handling and logging.
2. Create APIs that return appropriate status codes (200, 401, 404, 500, 502).

9. Testing and Debugging

Topics Covered:

- Unit testing with Mocha, Chai, or Jest
- Writing and running test cases
- Debugging techniques

Exercises:

1. Write unit tests for the to-do list API.
2. Debug an application using Node.js debugging tools.

10. Performance Optimization, Security, and Scalability

Topics Covered:

- Identifying performance bottlenecks
- Implementing caching strategies
- Securing applications

Exercise:

- Optimize a Node.js application by implementing caching and security best practices.

11. Deployment and DevOps

Topics Covered:

- Environment variables and configuration management
- Deployment strategies (PM2, CI/CD, cloud platforms)

Exercises (Optional):

1. Deploy the to-do list application to a production environment.
2. Implement a CI/CD pipeline for a Node.js application.

12. Working with WebSockets (Optional)

Topics Covered:

- Introduction to WebSockets
- Implementing real-time communication using Socket.io

Exercise:

- Create a chat application using WebSockets.

Final Project: Building a Task Management API

Task Breakdown:

1. **Project Setup and Basic API Structure**
2. **Authentication and User Management**
3. **Task Management**
4. **Validation and Error Handling**
5. **Testing and Documentation**

6. **Deployment and Refinement**

Final Evaluation:

- Code quality
- API documentation
- Security implementation
- Testing coverage
- Performance optimizations