1. **Կրթական պարտադիր չափորոշիչների և վերջնական արդյունքի սահմանում՝**

Node.js-ի այս դասընթացում մասնակիցները կբարձրացնեք իրենց JavaScript-ի հմտությունները: Կուսումնասիրեն այնպիսի թեմաներ, ինչպիսիք են ասինխրոն ծրագրավորումը, առաջադեմ երթուղավորումը, միկրոծառայությունների ճարտարապետությունը և անվտանգության լավագույն փորձը: Կտիրապետեն, թե ինչպես օպտիմալացնել կատարումը, իրական ժամանակում հաղորդակցություն իրականացնել WebSockets-ի հետ և տեղակայել Node.js հավելվածները արտադրական միջավայրերում: Այս դասընթացը նրանց կզինի Node.js-ի միջոցով մասշտաբային, անվտանգ և բարձր արդյունավետությամբ վեբ հավելվածներ ստեղծելու համար անհրաժեշտ փորձով:

2. **Դասընթացի կառուցվածքը, դասընթացին վերաբերող տեսական նյութերը (գրականության) հայերեն և անգլերեն լեզուներով:**

Ընդհանուր դասընթացների թիվ	**Շաբաթական դասընթացների թիվ**	**Մեկ դասընթացի տևողություն**	**Ընհանուր դասընթացների ժամաքանակ**	**Տեսական դասընթացների ժամաքանակ**	**Գործնական դասընթացների ժամաքանակ**
19	3 դաս	2 ժամ	38	19	19

Հ/Հ	Դասընթացի թեմա (գործնական/ տեսական)	Ժամերի քանակ		Դասընթացից ստացված արդյունքը
		Գործ.	Տես.	
1	Authentication and Authorization			Project structure Set up MongoDB Register new users Implementing login and logout Remember me User verification Resetting passwords
2	Authentication and Authorization	1	1	Using Passport.js for authorization Implementing local authentication with Passport.js Securiting routes Creating and sending JWT tokens Single Sign-On
3	WebSockets and Real-Time Communication:	1	1	Explore real-time communication using WebSockets and technologies like Socket.io for building interactive applications.

4	Socket.IO	1	1	Broadcasting Namespaces Rooms Chat Application
5	Databases	1	1	Database basics, relational databases, NoSQL databases and key-value stores, ODM, Mongoose, Aggregate
6	Databases			key-value stores, Redis, MySQL, ORM, Sequelize
7	Advanced Express.js	1	1	Middlawares Handling multipart form data with multer Resizing and storing images with jimp

8	Debugging and logging:	1	1	Debugging Tools Effective Logging Strategies winston
9	Testing:	1	1	Learn advanced testing techniques, including unit testing, integration testing, and end-to-end testing.
10	Security:	1	1	WASP Top Ten, Data handling with type and validation Rate limiting against DoS attacks
11	Error Handling Strategies:	1	1	Types of Errors. Custom Errors.

12	Microservices Architecture:	1	1	Dive into building scalable and maintainable applications using a microservices architecture with Node.js.
13	Microservices Architecture:	1	1	Creating a Service Registry Unregister services Creating the service client
14	Performance Optimization:	1	1	Learn how to optimize the performance of Node.js applications, including profiling, load balancing, and using tools like PM2.
15	Scalability and Load Balancing:	1	1	Learn how to scale Node.js applications horizontally and use load balancing techniques to handle high traffic.
16	Docker and Containerization	1	1	Understand containerization with Docker and how it can be used to package and deploy Node.js applications.
17	Continuous Integration and Deployment (CI/CD):	1	1	Explore CI/CD pipelines for Node.js applications,

				automating the testing and deployment process.
18	Performance Profiling:	1	1	Use profiling tools to identify performance bottlenecks and optimize your Node.js applications.
19	GraphQL:	1	1	Learn how to build and consume GraphQL APIs using Node.js.

Students should have some foundational knowledge

1. **JavaScript Fundamentals: Students should have a solid understanding of JavaScript, including variables, data types, control structures (if statements, loops), functions, and basic object-oriented programming concepts.**
2. **Node.js Basics: Familiarity with the basics of Node.js, such as setting up Node.js, using npm (Node Package Manager), and running simple Node.js scripts.**
3. **RESTful APIs: Understanding the concept of REST (Representational State Transfer) and how to make HTTP requests to APIs is important, as many topics in the course relate to creating and consuming APIs.**
4. **Database Fundamentals: Basic knowledge of databases.**
5. **Basic Concepts of Authentication and Authorization: Some prior knowledge of authentication and authorization concepts, including user authentication, role-based access control, and JWT (JSON Web Tokens), will be beneficial.**
6. **Asynchronous Programming: Understanding asynchronous programming in JavaScript, including callbacks, promises, and async/await, is crucial, as Node.js heavily relies on asynchronous operations.**
7. **Basic Web Security: Familiarity with web security concepts, such as common web vulnerabilities and best practices for securing web applications, will be helpful.**
8. **API Design: Basic knowledge of API design principles and RESTful API conventions.**