THE NODE.JS MATURITY CHECKLIST

FOLLOW THESE STEPS TO ADOPT NODE.JS IN YOUR COMPANY

1. BUILD A CULTURE THAT EMBRACES LEARNING

Introducing Node.js will test your company's ability to adopt new technologies.
Make sure your environment strives when it comes to new challenges!

Read more:
7 Tips to Create a Company Learning Culture
Creating a Learning Culture is a Must-Have to Gain Competitive Advantage
How to Build a Culture of Learning (and Why You Need to)

2. DO POSTMORTEMS

As you move your application to production, you will run into problems
no matter how great your team is. Make sure you never make the same mistake twice.

For this reason, do postmortems what everyone reads, so you as a team can learn from it.

Learn writing postmortems:
How to write a postmortem
Postmortem template

3. GET YOUR JAVASCRIPT KNOWLEDGE TO THE NEXT LEVEL

As Node.js programs are written using JavaScript, knowing the language is a must to get started with it.

Great resources to learn JavaScript:
JavaScript basics
You Don't Know JS (book series)

4. HOLD REGULAR INTERNAL TRAININGS

If your company just started with Node.js, it is important to get things right!

Without trainings, your colleagues can waste weeks on figuring out the best practices
- and could still end up using bad ones.

Need help in Node.js trainings? Get help from RisingStack!

5. SET UP AN AUTOMATED DELIVERY PIPELINE

When building even just MVPs, setting up a continuous integration / delivery pipeline is a must.
Running tests and code quality checks automatically can save you a lot of manual labor.

Resources to get started with automation:
Node.js CD Basics
Shipping Node.js with Docker

Tools that help:
Codeship
Circle CI
Jenkins

6. DO IMMUTABLE DEPLOYMENTS

Immutable infrastructures usually consist of data and everything else.

The everything else part is replaced on each deploy.
Not even security patches or configuration changes happen on production systems.

But why? Going back to older versions is easy, testing in isolation is possible,
and it simplifies change management.

Great resources to learn containers:
Getting started with Docker
Getting started with rkt

7. TAKE ADVANTAGE OF CACHING

With caching - whether, it is on the application or load balancer level -
you can make tremendous performance improvements, which will eventually result in happier users.

Resources on caching:
Redis caching introduction
Caching with NGINX
HTTP caching headers explained

8. EMBRACE VERSION CONTROL

The heart of your business is the application you develop and its codebase.

It is something you want to handle with great care.
One of the most popular version control software today is Git, which is a distributed VCS.

Learn Git:
Getting started with Git
Interactive Git tutorial

9. UPDATE YOUR DEPENDENCIES REGULARLY

Modules on npm are updated regularly, introducing bug fixes or new features. Make sure you update your dependencies on a weekly basis.

Tools that help:
npm outdated
Greenkeeper

10. USE CODE LINTING

Code linting is a static analysis that is used to find code patterns with problems or parts that don't respect a certain style guide.

As JavaScript is dynamic and loosely-typed, it is especially prone to programmer errors.

Tools to check out:
ESLint
Standard

11. LEARN CLEAN CODING

"Even bad code can function. But if code isn't clean, it can bring a development organization to its knees."

"Writing clean code is what you must do in order to call yourself a professional. There is no reasonable excuse for doing anything less than your best."

Resources to get started with Clean Coding:
Clean Coding with JavaScript
Clean Code: A Handbook of Agile Craftmanship

12. HAVE A GREAT TEST COVERAGE

You can think of tests as safeguards for the applications you are building.
But tests are much more than just safeguards - they provide a living documentation for your codebase.

Get started with testing:
Node.js unit testing tutorial
tap test runner for Node.js
Sinon - standalone test spies, stubs and mocks

13. EMBRACE THE SECURITY MINDSET

Take security seriously, and make security a part of your workflow!

Most companies don't invest enough in security, as usually it is not that visible
and decision makers don't feel the pain for the lack of it.

However, if your company gets exposed, you'll hurt a lot.

Get started with security:
Node.js Security Checklist
Node.js Security Tutorial

14. SECURE YOUR DEPENDENCIES

You are what you require.
Even if your code is safe, your dependencies may have known security isssues.

Read more:
Controlling the Node.js security risk of npm dependencies

Tools to verify your dependencies:
Trace by RisingStack
Snyk
Node Security Platform

15. HAVE PROPER LOGGING

Logging is a must if you want to spot and understand problems in your application easily.
Without logs, you cannot even tell what's going on.

Resources on logging:
How to Get Node.js Logging Right
Comparing Winston and Bunyan

16. START PERFORMANCE MONITORING

Knowing when your application stops or worse, crashes is crucial for building a successful business with Node.js.

Node.js monitoring:
Node.js Monitoring Done Right
Trace by RisingStack

17. SET UP ALERTING WITH ON-CALL SCHEDULING

Your team must be notified immediatly if your service goes down or even slows down, so you can react on it.

Blogposts on on-call schedules:
How ServerDensity handles on-call

Tools that help:
OpsGenie
PagerDuty

18. HAVE AUTOMATED BACKUPS

Your business depends on your data - make sure you have at least daily backups.

Tools that help:
Attic
rsync
Linux Backup Solutions