Best practices for implementing an application based on MEAN stack

MEAN stack is a combination of technologies/frameworks put together to work seamlessly and efficiently. MEAN is an abbreviation for “Mongo, Express, Angular, NodeJS” and is targeted to create applications using a single technology – JavaScript.

MEAN stack has been around in the market for a long time now, with ever increasing popularity within the open source community. Adoption rate of the MEAN stack is growing with the adoption of its underlying technologies. This technical paper will present key guidelines/best practices in creating an application using the MEAN stack components – **M**ONGO, **E**XPRESSJS, **A**NGULAR & **N**ODEJS.

mongoDB. | NodeJs
express EXPRESSJS | ANGULAR

Some MEAN stack use cases including but not limited to – frontend development, backend development, mobile app development.

Mongo

MongoDB has emerged as a database for modern application with an increasing adoption rate. MongoDB is often the go-to solution for a wide variety of use cases. Having said that, the issues associated with it include very tightly coupled systems, highly transactional systems with predefined data models, etc.

Mongo Atlas

This is a very niche cloud service for MongoDB that supplies a basic database to get you started in seconds. It has loads of features including – data lake, stitch, charts, search, analytics, ops manager, connectors, triggers, and a lot more, that enables an “Always On”, “Fully Automated and Elastic” database engine served as DBaaS.

Mongo Compass

Database development is made easy with the help of tools that are used to browse and manipulate data within. Mongo Compass is the easiest way to explore and manipulate data. Key features include – Schema visualization, Server Performance, Geospatial data visualization, and Visual Explain charts. With all these features built in, Mongo Compass is the only tool that you will need to work on your document data.

ExpressJS

A fast, unopinionated, minimalist web framework for Node.js. The statement holds true from every feature’s perspective. It is a minimalistic library that helps to create backend applications efficiently.

While creating backend applications, below are some of the guidelines that should be followed if we want to achieve a collaborative development working environment.

Folder structure

- **Config:** to keep the configuration of the entire application. To be used only for non-sensitive configurations
- **Controllers:** to keep all your routes hierarchy and configuration
- **Helpers:** To be used only for utility functions across your apps
- **Middleware:** placeholder for your application’s middleware
- **Model:** the place where the data layer is managed

Debugging & Error Handling

Inherent debugging provided by Express will, by default, provide you with the information on the routes being executed along with their performance metrics

Use a promise-based routing mechanism to handle errors efficiently

Implement a global error handler as a middleware to centralize your logic

Define nested error handler middleware if required

Performance

High availability should be enabled so that your application can handle the load

When in production, use NODE_ENV to “production”

Use a process manager so that the application is always running. “pm2” is a leader in this and is a recommended solution as well

Always use compression in production deployment

Use of a caching server (ngins) in case the application is in high need of caching

Security

- Always keep your package.json updated with the latest version of the dependencies you use
- Always secure your application with TLS1.2 at the very least
- Remove unwanted headers. Express recommends “helmet” that takes care of this specifically. Manual update is also possible. Affected headers includes - Content-Security-Policy, X-Powered-By, Strict-Transport-Security, X-Download-Options, Cache-Control, X-Content-Type-Options, X-Frame-Options, X-XSS-Protection.
- Use cookies securely. Use a module provided by express to handle the same.
- Implement expiry and renewal of token if used
- Enforce token blacklisting
- No hardcoding of any URI & credentials

Angular

There is no doubt that Angular is one of the key frameworks/platforms in UI development. The goal is to encapsulate all the required functions under one umbrella. Angular serves as a single framework to build web and mobile applications.

Below are key guidelines and best practices to follow when it comes to building applications using Angular.

Always use a UI framework

Recommended UI frameworks:

Angular Material – Fast, consistent, customizable, and optimized for Angular. A well thought and designed library which is in line with Material design principles.

Zurb Foundation – Significantly advanced framework which is customizable to the level you want it to be at

Semantic UI – If we want to stand apart from the monotonous/standard design of Material and bootstrap.

Opt for out-of-the-box modules

Routing, security, internationalization, and server-side rendering – Angular supplies all of these out-of-the-box. You will barely come across a situation where you will need a specific functionality that does not exist in the framework. Hence, use OOTB features and keep dependencies to a minimum.

Keep it Asynchronous

Since Angular is based on the Node.js framework, the key is to follow the underlying architecture principle. Always use async features wherever you can to speed up the application. Make your flow to adapt this and both UI and performance will be rich.

Server-Side Rendering (SSR)

This is extremely useful for specific use cases where you want the first load time to be faster. Use of a framework helps to overcome several hurdles in this area. “Nestjs, a progressive Node.js framework for building efficient, reliable and scalable server-side applications” is one of the key mentions.

Node.js

We all tend to use modules created by others without learning the full scope of the native language features. While it is tempting to use convenient modules, being efficient in your application design requires the use of native modules. This applies to Node.js as well. A few modules worth mentioning include:

Crypto -
a comprehensive cryptography module that can support any of your encryption requirements

Cluster -
a module to enable high availability for Node.js applications from the ground up. This will give you the control which open source libraries may not be able to

Assertion testing -
building test cases for your scripts is easy with this module

Utilities -
a host of OOTB utilities functions which the platform provides

Conclusion

Considering its increasing popularity in the open source community, MEAN stack is a recommended solution to build cutting-edge web and mobile applications.

About the Author

Abhishek Tanwar

Enterprise Architect, DMTS Senior Member, Communications BU, Wipro Limited.

Abhishek Tanwar is an Enterprise Architect leading the Cloud Transformation program in T-Mobile Netherlands as part of the Communications BU. He has ~15 years of experience working with various service providers across Europe and South-East Asia in the B/OSS and Digital domains.

● **Wipro Limited**
Doddakannelli, Sarjapur Road,
Bangalore-560 035,
India

Tel: +91 (80) 2844 0011
Fax: +91 (80) 2844 0256
wipro.com

Wipro Limited (NYSE: WIT, BSE: 507685, NSE: WIPRO) is a leading global information technology, consulting and business process services company. We harness the power of cognitive computing, hyper-automation, robotics, cloud, analytics and emerging technologies to help our clients adapt to the digital world and make them successful. A company recognized globally for its comprehensive portfolio of services, strong commitment to sustainability and good corporate citizenship, we have over 175,000 dedicated employees serving clients across six continents. Together, we discover ideas and connect the dots to build a better and a bold new future.

For more information,
please write to us at
info@wipro.com