# Transforming Inventory Management System using MEAN Stack

**Ritu Sharma[#1], Anushka Bajpai[#2], Ankur Maheshwari[#3], Ashwini Sharma[#4], Gaurav Gupta[#5]**

*[#]Department of Computer Science & Engineering, Inderprastha Engineering College, Plot No 63, Site IV, Sahibabad, Ghaziabad, 201010, India*

---------------------------------------------------------------------***---------------------------------------------------------------------

***Abstract—*** By offering more personalized service to customers, the retail company's objective is to increase returns through customer happiness and loyalty to the store. However, if a retail store doesn't have enough inventory, it becomes prone to lose a potential consumer. The tribulation of the shop is that they do not have the proper inventory control device in guiding and handling their sale and inventory level of the shop. By proposing a Sale &amp; inventory management system to the vendors as a substitute for antique guide approaches, the task targets to equip the platform with more bendy functions to the shop. We provide the vendors and SMEs with a web application using the MEAN stack for developing a sales and inventory control system. The MEAN stack allows us to provide a higher user interface and a devoted database to work precisely.

***Keywords—*** **ERP, Sales & Inventory Management, Khata-Books, MEAN stack, MongoDB**

## I. INTRODUCTION

The retail industry is one of the industries that is developing at a rapid pace wherein the number of retail businesses maintains on increasing once in a while to meet the call for customers of exact regions. There are varieties of retail stores for the customers to select from according to their convenience, starting from hypermarkets to mini markets. The maximum number of the shops are based in residential areas and streets. Essentially, retail continues to sell a wide range of wholesale or personal stop-off products and services. For this reason, the character of retail enterprise required an effective control of the stock stage intending to meet the clients' demands.

The traditional store keeps their income and stock details in spreadsheets which aren't powerful anymore when the dimensions of the store are larger. This is because additional items are stored in huge amounts, as a result, monitoring the revenues made with the stock level in the shop would be complicated and time-consuming for the store. The situation worsens when the retailers lack the proper technique to decide the gadgets bought with the asset of their clients. With the recent development of the digital generation, developers worldwide have in no way a great and free time installing new Internet packages. Better than the traditional 'khata books', it enhances the quality and reach of data and inventory management, reduces time and redundancies, and creates opportunities to expand the business. It makes a truly ambitious system for resource management.

The MEAN stack era turned into a device used to create a sales and stock management system. It's still one of the sturdy full-stack technologies used to increase the efficiency of most trafficking websites. With the growing potential and demand for mobile operations, such systems must incorporate adaptable designs and web-based methodologies into their operations. While many systems still use rich client architectures, our inventory management systems use web-based front ends that simply require a regular browser environment to operate. This paper appears at the 4 implied stack additives (MongoDB, ExpressJS, Angular & NodeJS) and the way they move nicely collectively to build an adequate inventory management system.

## II. PROBLEM

Inventory is one of the critical departments that need to be properly controlled to make sure day by day enterprise activities run smoothly. But, many shops do not realize the significance of inventory management as they are not ready to move with computerized devices in walking their enterprise. As a result, the safety level of all information, documents, and some things associated with daily transactions and inventory may become very low. When several documents are maintained for each item and each supplier, it consumes plenty of time and eventually becomes ineffective for future references. Furthermore, due to bad sales and stock management, many additionally face hassle in figuring out the amount sold for each item in keeping with the day and available stock level of the item units. With an Inventory management system, the challenge for establishments forms the execution stage, which can be even more complex and risky, especially for SMEs. Factors that fail include, among others are high customization efforts, poor business process reengineering, poor consultant quality, and the lack of top management support.

There are many inventory management systems available in the market, but either they are cost-ineffective or are complex to use and understand which is not convenient for shop owners. Our application comprises the databases that enable information storage and retrieving of each transaction and information about the inventory of each gadget within the save, control the product releases and garage, and summarize the factor of sales. It would generate a quicker improvisation of work with much less time and effort.

As the concept of a sales and inventory management machine is to lessen paperwork and useless approaches of coping with stock, this gadget is inclined to help make the proper decision within the system, dealing with inventory aligned with the sales level in the shop.

## III. Concepts and Usage

Day after day, millions of people across the globe engage in innumerable sales transactions, yielding a steady flow of value that acts as the footing of our economies.

In general, a sale is a transaction between two parties in which the buyer obtains actual or intangible products, services, or assets in exchange for payment. As a result, each party must give up something to gain something beneficial. Inventory refers to the raw materials, work-in-process items, and finished goods that make up a fraction of a company's assets sold. Hence, It explains why a business requires inventory for making sales of products in exchange for money, which generates profits.

High and low inventory are two kinds of issues businesses encounter while managing inventory levels. Because of the high costs of inventory storage, devaluation, and deterioration, keeping a large inventory for an extended period is usually not a good decision for a firm. On the other hand, Low inventory is not worthwhile because the company starts losing valuable sales and market share.

"*Inventory management systems are the rule that applies to analyzing which items are selling and which are cluttering up shelf space for businesses, small businesses, and suppliers*.", according to Tim Crosby (2012) in his paper on 'How Inventory Management Systems Work'. The strategy balances a retailer's financial necessity to keep as little stock as possible with the objective of verifying that customers always have plenty of what they desire (Tim Zierden, 2009). Thus, modern inventory management systems must have the ability to track sales and available inventory, connect with suppliers in near real-time, and collect and incorporate additional data such as seasonal demand.

Because the goal of a sales and inventory management system is to reduce paperwork and inefficient ways of dealing with stock, this technology is willing to aid in making the best option within the system, managing inventory per the store's sales level.

## IV. Advantages of Sales and Inventory Management System

As the advantages of switching to modern inventory control system proven by many businesses, Donal Reimer (2008) in his study, identified the following benefits of Sales and Inventory Management System:

### A. *Inventory management increases profitability*

Forecasting, controlling, and overseeing inventory boosts the store's sales and production, increasing profitability. Moreover, inventory reliability enhancements will diminish the cost of correcting costly mistakes. Expense management will be improved, with rapid access to current and historical pricing, bridge product codes, and tools for controlling purchase operations generated by the system (Zipkin, P.H., 2000).

### B. *Inventory management improves cash flow*

Better cash flow and, eventually, larger profitability result from purchasing the proper inventory in the right amount to fulfill customer demand while also reducing slow-moving, outmoded inventory.

### C. *Inventory management improves decision-making*

Rapid, reliable data collecting allows for real-time business intelligence across all sections of the store. Not only that but issues and events coupled with the system allow for proactive issue detection and resolution.

### D. *Inventory management increases customer satisfaction*

By keeping the correct products in store for clients, you can anticipate seasonal promotions and shifting marketing conditions. The business is in a direct relationship with customer satisfaction. If it becomes possible to track the growth of sales by any means, this increases the customer's confidence, business confidence, and boosts economic growth.

## V. Formulation of web application

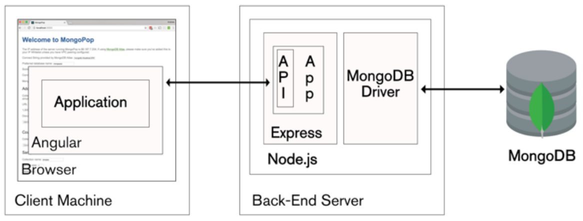
Stack web development tasks to analyze and understand MEAN's current technologies dynamics and challenges (Mongo Db, Express Js, Angular, Node Js). MEAN Technologies and their ease of use properties for the modern developer are used to make FULL-STACK architecture viable. The MEAN stack consists of MongoDB as a database, Express as a server system, Angular as the front-end, and Node Js as a JavaScript server-side environment. MongoDB is a more versatile and adaptive data storage layer. Node.js gives a better nexus for running the server, whereas Express.js aids in website development standardization. AngularJS provides a simple approach to adding interactive functions and AJAX-driven rich components to client-side applications. This all comes together to create a transparent, logical system transporting data from the user to the disc farm and back again.

In many systems, MEAN is extremely easy for both the back end and the front end, but the distinct languages for the front and back ends are written in a single language for server and client-side execution.

### A. *Architecture of MEAN stack*

When built one on top of the other, the MEAN architecture is made up of four separate JavaScript technologies. The architecture is built such that each component does a specific purpose while working in tandem with the other layers in the stack to reach a common goal. In the MEAN architecture as depicted in the picture below, the client initiates requests, which flow from the first to the final layer. Appropriate replies to those requests then flow back up to the client through all the levels.

The user interface, which is designed using the Angular frontend framework, is the first point of contact for the client. The server-side engine, Node.js, receives requests sent through the interface. After that, the Express.js middleware framework sends a request to the database MongoDB. Express.js gets the answer from the database in the form of data. Finally, Node.js returns this response to the client via the user display:



fg. 1

### B. *User Interface*

For any application, clarity is essential. When it comes to sales and inventory or a POS system, the data comes in PetaBytes, and dealing with such a quantity of data can be disorganized. Such problems culminate by using more stable and single-language scripting framework stacks. Such stacks allow a code to run from client to server.

Angular is such a frontend framework for creating single-page applications (SPAs) that load on a single page. It continues to streamline as a result of the utilization of Ajax call techniques and multiple page loading. Two-way data binding between views and models is available with Angular. This is important for developers in terms of timetables, and HTML makes it much easier to place tasks on a template (Adam and Colin, 2014).

### C. *Walkthrough of an Inventory Management system*

This paper references an application developed through the MEAN stack that deals with sales and inventory management. The tour starts with a user signing up for the system. When the user signs up with a unique ID, he gets to a simple dashboard. Then they can access the product details page, customer details page, and sales page. These pages are made using Angular, and NodeJS connects them to the database. The entire process is divided into four phases. The initial phase starts with a user generating a request. When a client generates a request, AngularJS processes it first. The request then moves on to phase 2, which is NodeJS. The request to the database is made in phase 3 by ExpressJS. MongoDB retrieves the data and returns it to ExpressJS as a response. Then ExpressJS returns a response to NodeJS, which NodeJS then passes on to AngularJS to display the result.

For all respective pages, the format remains the same where users can add data and modify or delete it as required. The details are updated directly through the database and no extra implementation is required on the client's machine.

### D. *Database*

The previous database storage methods are inconvenient. Shopkeepers, for example, use khatabooks to keep track of their daily sales, more goods are available in bigger quantities, tracking sales based on inventory levels in the shop becomes more complicated and time-costly for the merchant. To deal with such problems, RDBMS came into action. It is a database management system that allows data to be stored more flexibly. RDBMS guarantees that data is saved in the form of rows and columns, limiting redundancy and replication. Developing further from it, here we are using MongoDB to create the database for this application. MongoDB characterizes a Web-based application as a database that holds data. MongoDB is a database that focuses on documents. MongoDB Inc. created a NoSQL database. It's a free and open-source database that works with schemas and JSON-like documents.

MongoDB is an application data platform built on many basic architectural principles that helps you fulfill the needs of modern apps. The document data architecture provides a straightforward yet powerful query interface, which allows developers to construct transactional, operational, and analytical applications quickly and easily. A multi-cloud global database that gives developers the freedom to operate their apps anywhere and the flexibility to shift between private and public clouds as needs change — all without changing a single line of code.A unified experience for contemporary apps that spans the cloud to the edge, including database, search, and the data lake, with extensive and integrated developer services.

It offers several unique characteristics that set it apart from other databases in terms of application and design. This has a complicated structure and is hence widely used to build scalable systems.

## *E. Backend Processes: NodeJS & ExpressJS*

This article intends to offer a framework for application developers that includes software packages such as Node.js and MongoDB, as well as APIs that allow them to connect their application to their database without having to worry about server-side code. Additionally, these APIs may be used to construct drivers for a variety of platforms, including Android,.NET, and iOS. An application developer should be able to link their apps and websites to a database server more comfortably and straightforwardly. This backend server can handle many user requests and has a high concurrency rate. It should also take data privacy and security into mind. In JavaScript, Node.js is a relatively new and popular technology.

It's a framework for simply creating fast, scalable network applications based on Chrome's JavaScript engine. We want to build a framework for a database backend server that can allow us quickly connect our frontend to MongoDB by leveraging Node's fundamental features and connecting it to MongoDB, one of the fastest and most scalable databases available. Express.js is a Node.js web framework. It includes several features that enable developing web applications faster and easier than if only Node.js were utilized. The connect Node.js middleware module, which uses the HTTP module, is the foundation of Express.js. As a result, Express.js will work with any connect-based middleware.

It comes with built-in HTTP server libraries, allowing developers to design their web server as well as highly scalable online applications. For each request, Node.js produces an event with event handlers. The association handler is queued for execution when an input/Output action happens, and a callback function emits an event when the Input/Output operation is done. As a result, Node Js handles I/O asynchronously and doesn't stop script execution, enabling the event loop to react to additional requests. After the Node module has been added to the current script, it gives a public API that may be accessed.

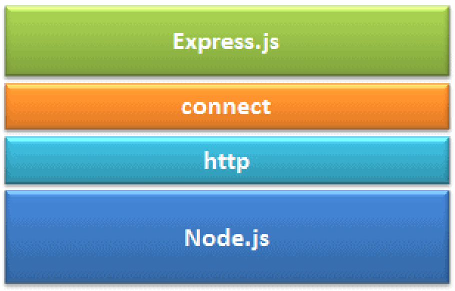


fig. 2

ExpressJS helps to develop Node.js web applications quickly and easily. It's easy to set up and personalize. It allows the creation of application routes using HTTP methods and URLs. It also includes several middleware modules that can be used to execute additional requests and responses activities. Simple to interface with a variety of template engines, including Jade, Vash, and EJS, it allows you to specify a middleware for handling errors.

## *F. MongoDB (M), ExpressJS (E), AngularJS (A), NodeJS (N) - MEAN Stack Implementation.*

The installation of Node.JS and Express.JS was crucial in laying the groundwork for the application. The following commands were used on Ubuntu version 16.04 command line interface (CLI) to install Express.JS after setting up the development environment.

```
npm update -g express

npm update -g express-generator

express hrtc

cd hrtc

npm install
```

### *1) App.js Startup file*

The app.js file was used to start up or initialize. The starting point of a web app is the app.js file. In most cases, apps listen for requests on a certain port. app.js generates a Node.JS signaling server as well as a port for the application to listen on. It also contains handler methods for various RESTful routes and activities as dependencies.

*2) Package.json*

This file is used to record and remember the internal dependencies of the program, as well as their versions.npm install –d" was used to implement package.json. This program installs all of the dependencies provided in the Node Package Manager (NPM).

```
{
 "name": "node demo",
 "version": "1.0.0",
 "description": "",
 "main": "StartNodeServerTest.js",
 "scripts": {
  "test": "echo \"Error: no test specified\" && exit
1"
 },
 "keywords": [],
 "author": "",
 "license": "ISC",
 "dependencies": {
  "body-parser": "^1.19.0",
  "cors": "^2.8.5",
  "express": "^4.17.1",
  "mongodb": "^4.2.0",
  "mysql": "^2.18.1",
  "nodemailer": "^6.7.1",
  "nodemon": "^2.0.15"
 }
}
```

*3) Express.js*

The Express.js file starts with the express() function which creates an Express application. The express() function is a top-level function and is exported by the express module.[24]

The ExpressJS framework is a key component of the MEAN stack. This framework's features are basic, yet they're important in the overall design of a web application. ExpressJS can be used for both routing and middleware. ExpressJS includes over 20 middleware components, including a logger, session support, and cookie parser, among others. One of the most common uses of the app. use() is to utilize it to plug into ExpressJS middleware by passing it as an argument. It will be in charge of routing based on URL paths, managing sessions via cookies, parsing incoming requests (for example, JSON), and routing the request to a handler, which is a JavaScript callback function that can be programmed inline or in a separate module (Madhanasekaran, 2015).

The express.static() function is a built-in middleware function in Express. It is based on serve-static and serves static files.

```
const express = require('express');
const app = express();
app.use(express.static(path.join(__dirname,'public')));
app.get("/api/:name/:age",(req,res)=>{
res.json({name:req.params.name,age:req.params.age });
})
```

*4) MongoDB*

In contrast to the old connection class Db, which has acknowledgments turned off, the new connection class MongoClient acknowledges all writes to MongoDB. The prefix mongodb:/ indicates that this is a string in the standard connection format. With MongoClient.connect, one can use the URL format. MongoClient picks the best default settings for options wherever possible.[7]

```
var MongoClient=require('mongodb').MongoClient;
var url = "mongodb://localhost:27017/";
var Database = "mydb";
var con=MongoClient.connect(url, function(err, db) {
 return db.db(Database);
});
```

## VI. Discussion

Due to the lack of implementation of MongoDB, ExpressJS, AngularJS, and Node.JS (MEAN stack) new technologies as a single stack, the importance of this study become apparent when components communicate internally using a single common language. This will negate the advantages of utilizing the same language throughout the stack. In terms of developer experience, speed, agility, M-V-C support, scalability, open-source, and cloud deployment, MEAN is cost-effective. These benefits are bigger as compared to other technologies. Developers may simply install apps directly on the server using Node.js, eliminating the requirement for a separate standalone server. The overheads associated with the client-server design are reduced as a result.

The practical experience in designing the emailing subsystem utilizing MEAN stack technologies has been described in this paper. The new idea highlights the differences in the stack of the most prevalent web application technologies, such as LAMP and earlier technologies, that have been noticed. When compared to alternative options, these components provide better interaction convenience. The study's user interface findings revealed that users may simply connect from any location. The software provides users with flexibility, simple interactions, and the needed level of service.

The study found that apps built using the MEAN stack technology are simple to create, install, and use, as well as being highly engaging.

## VII. RESULT

Inventory Management System is a straightforward web-based online application that is best suited for small businesses. It has all of the essential components for a small business. Our team has succeeded in developing an application that allows us to edit, insert, and delete items as needed. This application is ideal for small businesses with a limited number of warehouses. Despite its limitations, our team is convinced that using this technology would be beneficial to the firm. With modern cutting-edge technologies like MEAN stack, we also ensure the proper functioning and a better user interface.

## VIII. CONCLUSIONS

This paper started by summarizing the relationship between Sales and Inventory. The management and concepts were discussed concerning problems and advantages. The common advantages of proper inventory management were illustrated. With the reference of a Web-based self-developed application, the MEAN stack was defined.

LAMP as well as its variants are historical technologies that have spawned a plethora of interesting and useful applications all across the world, and this cannot be overstated. JavaScript, on the other hand, is a leader in web development in today's technologies. Any stack can be used according to the programmer's needs and priorities, the MEAN stack has yet again proved to be a dependable solution for quick, extensible, and real-time applications. The MEAN stack is a modern cutting-edge, and robust technology that overthrew web development platforms. NodeJS' s non-blocking approach made concurrency simpler. With the JSON data format, the MongoDB document-oriented, No-SQL database has demonstrated higher performance, flexibility, and scalability in new technologies. Express.JS was created based on Node.JS to give developers a simple framework to work. Google created AngularJS to promote the MVC framework for developing SPAs and swiftly creating stunning interactive user interfaces.

## IX. REFERENCES

[1] Alfred, A. (2014). Node.js: Introducing the MEAN Stack.

[2]. Adam, B. and Colin J., I. (2014). Full Stack JavaScript Development with MEAN. Retrieved on

November20,2017http://pepa.holla.cz/wp-content/uploads/2016/11/mean1.pdf

[3]. Bretz, A., & Ihrig, C. J. (2015). Full stack JavaScript development with MEAN.

[4]. Burns, N., & Grove, S. K. (2009). The practice of nursing research :appraisal, synthesis, and generation of evidence. St. Louis, Mo: Saunders Elsevier.

[5]. Perrenourd, M. (2015). Learning web development with the MEAN stack..

[6]. Bojinov., V. (2015). Design and implement comprehensive RESTful solutions in Node.js.

[7]. Dickey, J. (2015). Write modern web apps with the MEAN stack: Mongo, Express, AngularJS, & Node.js. San Francisco, CA: Peachpit Press.

[8]. Dirolf, M. (2010). Binary JSON. Retrieved from: http://bsonspec.org

[9]. Edim, A., E. and Bakwa, D., D (2017). A Peer-To-Peer Architecture For Real-Time Communication Using Webrtc. Journal of Multidisciplinary Engineering Science Studies (JMESS) ISSN: 2458-925 3(4).

[10]. Elrom, E. (2016). AngularJS SEO. Pro MEAN Stack Development, 197-219. doi:10.1007 /978-1-4842 - 2044-3_8

[11]. Elrom, E. (2016). CSS, Bootstrap, & Responsive Design. Pro MEAN Stack Development, 131-164 doi: 10. 1007/978-1-4842-2044-3_6

[12]. Fhala, B., & Chrispinus, E. O. (2017). Learning path: MEAN : create MEAN stack apps.

[13]. Grover, R. (2015). Building Apps with MEAN Stack: The Benefits of the MEAN Stack.

[14]. Haviv, A. Q., Mejia, A., & Onodi, R. (2016). Web application development with MEAN:Unlock the power of the MEAN stack by creating attractive and real-world projects : a course in three modules.

[15]. Ihrig. J. C., and Bretiz, A. (2015). Full Stack Javascript Development with MEAN.

[16]. Kent Beck. (1999). Extreme Programming Explained: Embrace Change. Addison-Wesley Longman Publishing Co, Inc, Boston, MA, USA.

[17]. Wikipedia(2013), Inventory Management Software Retrieved 22 Jun 2013 from http://en.wikipedia.org/ wiki/ Inventory_management_software

[18]. Anton Dolinsky (2007), Barcodes, sales and inventory control Retrieved 22 Jun 2013 http://www. almyta.com/Inventory_ManagementHistory_4.asp.

[19]. August-Wilhelm Scheer and Frank Habermann. 2000. Enterprise resource planning: making ERP a success. Commun. ACM 43, 4 (April 2000), 57–61.

[20]. Muscatello, J.R. and Chen, I.J., 2008. Enterprise resource planning (ERP) implementations: theory and practice. International Journal of Enterprise Information Systems (IJEIS), 4(1), pp.63-83.

[21]. Al-Mashari, M., Zairi, M. and Okazawa, K., 2006. Enterprise Resource Planning (ERP) implementation: a useful road map. International Journal of Management and Enterprise Development, 3(1-2), pp.169-180.

[22]. Ngo, C.J., Chang, J. and Chung, S., 2020. Software Documentation and Architectural Analysis of Full Stack Development.

[23]. Dunka, Bakwa & Emmanuel, Edim & Oyerinde, Yinka. (2018). Simplifying Web Application Development Using-Mean Stack Technologies. 04.

[24].“5.X API,” *Express 5.x - API Reference*. [Online]. Available: https://expressjs.com/en/5x/api.html.