International Journal of Science and Research (IJSR) ISSN: 2319-7064

SJIF (2022): 7.942

# Developing and Building MEVN Stack

# Jagadeesh Kancharana

Abstract: This article provides a comprehensive guide to developing and deploying a MEVN stack application, using a task management project as an example. It details the setup of essential tools and technologies like Node. js, Express, MongoDB, and Vue. js, and explores various methods to automate the build and deployment processes, including the use of Webpack, Gulp, and Docker. The article aims to streamline the development process and ensure the successful deployment of scalable web applications. This paper provides a comprehensive guide to building a MEVN stack application and deploying it in a production environment. The guide covers the essential steps involved in development, packaging, and deployment, highlighting various methods to streamline and automate the processes. The development section begins with setting up the required prerequisites, including Node. js, Express, Mongoose, MongoDB, and various tools such as Vue CLI and Postman. An example project, specifically a simple task management application, is used to demonstrate the practical implementation of the stack. This project involves creating, retrieving, editing, and deleting tasks through API calls to a Node. js server, storing data in MongoDB. Several approaches for building and packaging the application are explored. The manual implementation section walks through the process of manually building the Vue, is application, placing the static assets in a production - ready directory, and running the Node. is server. However, this method is time - consuming and prone to errors, leading to the exploration of more efficient techniques. The paper then introduces Webpack, a module bundler that simplifies the build process by combining the server code and signific assets into a single file. This approach reduces the need for manual installation of dependencies, but still requires some manual steps. To achieve full automation, the guide demonstrates how to use Gulp, a toolkit that automates repetitive tasks. By setting up a Gulpfile, developers can automate the entire build process, including cleaning directories, building the Vue. js code, bundling the server code, and zipping the final package. This method significantly improves productivity by eliminating manual steps and reducing the potential for errors. Finally, the paper discusses the use of Docker for packaging and deploying the MEVN stack application. Docker allows developers to containerize the application, creating an isolated environment that includes all dependencies and configurations. This container can then be deployed on various container platforms, ensuring consistency across different environments. The purpose of this article is to guide developers through the process of building, automating, and deploying a MEVN stack application for efficient web development.

**Keywords:** MongoDB, Express. js, Vue. js, Node. js, Server - side, Type - Script, Bootstrap, MEVN stack, web development, Node. js, Vue. js, Docker

There are so many ways we can build VueJS apps and ship them for production. One way is to build the VueJS app with NodeJS and MongoDB as a database. The MEVN stack is popular because it allows developers to use JavaScript across all components. The four things are MongoDB, VueJS, Express, and NodeJS. This stack can be used for a lot of uses cases in web development.

Developing the application is one part and you need to package it based on your deployment needs once the development part is completed. There are so many ways we can package and ship

MEVN Stack to production: manual, with webpack, with Gulp, etc. This article explores these approaches in detail.

- Prerequisites
- Example Project
- MEVN Stack Development
- Manual Implementation
- With Webpack
- Packaging With Gulp
- With Docker
- Summary
- Conclusion

## 1. Prerequisites

There are some prerequisites for this post. You need to have a NodeJS installed on your machine and some other tools that are required to complete this project.

- NodeJS
- Express Framework
- Mongoose
- MongoDB

- VSCode
- Postman
- nodemon
- dotenv
- Vue CLI
- Typescript
- BootstrapVue
- gulp. js
- Docker

**NodeJS:** As an asynchronous event - driven JavaScript runtime, Node. js is designed to build scalable network applications.

**Express Framework:** Express is a minimal and flexible Node. js web application framework that provides a robust set of features for web and mobile applications.

Mongoose: elegant MongoDB object modeling for node. js

**MongoDB:** MongoDB is a general - purpose, document - based, distributed database built for modern application developers and for the cloud era.

**VSCode:** The editor we are using for the project. It's open - source and you can download it here.

**Postman:** Manual testing your APIs **nodemon:** To speed up the development

If you are a complete beginner and don't know how to build from scratch, I would recommend going through the below articles. We used these projects from this article as a basis for this post.

How To Develop and Build Vue. js App with NodeJS

How to Build NodeJS REST API with Express and MongoDB

How to write production - ready Node. js Rest API — Javascript version

# 2. Example Project

Here is an example of a simple tasks application that creates, retrieves, edits, and deletes tasks. We actually run the API on the NodeJS server and you can use MongoDB to save all these tasks.

← → C (0)	localhost.8080	I			ô \$	🖈 🇐 Updata 🚦
		MEVN St	ack Example	e		
	ToDo List					
	Task		Assignee			
	Create a Task		Assignee			
	Status:					
					~	
	Create					
	Tasks					
	Task Id	Task Name	Assignee	Status		
	62129522077477e8b041848c	sdfsdf131232133123	sdfsdfsdf55	To Be Done	Edit Delete	

As you add users we are making an API call to the nodejs server to store them and get the same data from the server when we retrieve them. You can see network calls in the following video.

				66				twork > cache No throttlin	
	MEVN Sta	ack Example		• © ± ±	γq	D Preserve	og   [] Disable	cache No tryotor	0.4.40
				Filter		0 in	vert 🗌 Hide dat	URLs	
								WS Wasm Manh	est Other
ToDo List				Has bit	22 ma	es 🗋 Blocked	Fequests 3		10
Task		Assignee							
Create a Task I		Assignee							
Status:									
			~						
Create									
Create									
Create									
Create									
Create									
Create									
Create									
Create									
Create						Bac	cardina networi	activity	
Create							cording network		
Create					F		uest or hit # R	to record the rel	iced.
Create					,			to record the rel	ced.
Create					,		uest or hit # R	to record the rel	iced.
Create					,		uest or hit # R	to record the rel	ced.
Create					,		uest or hit # R	to record the rel	load.
Create					,		uest or hit # R	to record the rel	oad.
Crane					,		uest or hit # R	to record the rel	oad.
Create					,		uest or hit # R	to record the rel	icad.
Craw					,		uest or hit # R	to record the rel	oad.
Create					F		uest or hit # R	to record the rel	oad.
Craw					,		uest or hit # R	to record the rel	oad.
Create					,		uest or hit # R	to record the rel	oad.
Create					,		uest or hit # R	to record the rel	oed.

## **Network Calls**

Here is a Github link to this project. You can clone it and run it on your machine.

// clone the project
git clone https: //github. com/bbachi/mevn - stack - example.
git

// Vue Code
cd ui
npm install
npm run serve

// API code
cd api
npm install
npm run dev

# 3. Project Structure

Let's understand the project structure for this project. We will have two packages. json: one for the VueJS and another for nodejs API. It's always best practice to have completely different node\_modules for each one. In this way, you won't get merging issues or any other problems regarding web and server node modules collision. It's easier to convert your MEVN Stack into any other stack later such as replacing the API code with microservices and serving your UI through the NGINX web server.



# 4. Project Structure

If you look at the above project structure, all the Vue app resides under the **ui** folder and nodejs API resides under the **api** folder.

## 1) Set up a MongoDB Atlas

The core of MongoDB Cloud is MongoDB Atlas, a fully managed cloud database for modern applications. Atlas is the best way to run MongoDB, the leading modern database.

Let's create your MongoDB Account here. You can either log in with any of your Gmail accounts or you can provide any other email address to create the account.

Join MongoDB in supporting organizations that are fighting for racial justice and equal opportunity.	
kin ten 4	
	1
	acial justice and equal opportunity.

## 2) MongoDB Atlas login

Once you log in with your account you will see the dashboard below where you can create clusters.



#### **MongoDB Dashboard**

Let's create a cluster called todo - cluster by clicking on the build a cluster and selecting all the details below.

our needs. For more information, check		ar options, but feel free to customize your clus	ne i
Cloud Provider & Region	Cloud Azure	AWS, N. Virginia (us-east-1)	~
★ Recommended region ① AUSTRALIA Sydney (ap-southeast-2) ★	NORTH AMERICA	EUROPE	
ASIA	N. Virginia (us-east-1) 🖈	∎ ∎ Ireland (eu-west-1)★	
Mumbai (ap-south-1)			
Cluster Tier	M0 Sandbo	x (Shared RAM, 512 MB Storage)	>
Additional Settings		MongoDB 4.2, No Backup	>
		Cluster0	

#### **Creating a Cluster**

#### Here is the cluster we created below.

Access Manager + Support Billing			All Clusters	Bhorgay
😌 Alfas 😂 Realm 🖉 Ch	arts		م	Q. E
AROA/S-0RG - 2020-04-08 > STICKERBUTUER				
lusters			Create	e a New Cluste
Find a cluster				
SANDBOX				
e todo-cluster Version 4.2.11	Operations R: 0 W: 0	100.0%	Logical Size 0.0 B	512.0 M
CONNECT METRICS COLLECTIONS ···				
CLUSTER THER	Last 6 Hours	0	Last 6 Hours	•108
M0 Sandbox (General)	Last e Hours		Last 6 mours	
REGION AWS / N. Virginia (us-east-1)	Connections 0	500	Enhance Your Experience	
TYPE Replica Set - 3 nodes		-	Enhance Your Experience For dedicated throughput, richer metrics and enterprise sec options, upgrade your cluster now!	urity
LINKED REALM APP			Upgrade	
None Linked	Last 6 Hours			

#### todo - cluster

You can click on the connect button to see the details about connecting to the cluster. You need to create a user and Allow Access from anywhere for now.

Step correction security         Choose a connection method         Connect           Nor need to incure your MongOED Alas cluster below you can use A. Stet which users and IP addresses can access your cluster need.         You can't connect IP address           Vac and connection IP address         Add a connection IP address         Addresses can access and user recurity permission below.           Image: Static group fermula access and user recurity permission below.         Add a connection IP address         Addresses (addresses)           Contact a Database User         Mony Access from Anywhere         Mony Access from Anywhere           Useration         Passender         Succession for this project.           Useration         Passender         Succession for the security permission for this project.           Image: Succession (State Database User)         Passender         Succession (State Database User)						
Add a connection IP address     Add a Different IP Address     Address     Add a Different IP Address     Add a Different IP Address     Add a Different IP Address     Addres     Address     Address     Add					IP addresses can a	access your
Add those Corrent IP Address     Ador a Different IP Address     Allow Access from Anywhere       Create a Database User     This final user will have adatabation (if permissions for this project. Korey your corrential handy, you'll rend them for the next step.       Usersone     Passened       infmin     Immin Sector	You can't connect yet. Set	up your finewall access and us	ser security per	mission below.		
This first user will have attaAdmin ( <sup>2</sup> ) permissions for this project. Krep you credential hardy, you'll read them for the net step. Username Passeod	-		Address	Nov Access fro	n Anywhere	
Keep you credential handy, you'll need them for the need step. Username Passwood						
admin				ect.		
Create Database User		P		R <sub>4</sub> Autopromite		
					Create Databa	ase User
	Close				Choose a connect	tion method

#### **Connecting to cluster**

Manager 👻		×	
Atlas	Connect to todo-cluster		
deploying you	Setup connection security Choose a connection method Connect		
ers	You need to secure your MongoDB Atlas cluster before you can use it. Set which users and IP addresses can access your cluster now. Read more C		
cluster	You're ready to connect. Choose how you want to connect in the next step.		
ox lo-cluster 14.2.11	Add a connection IP address An IP address has been whitelisted. Add another whitelist only in the IP Whitelist tab.		
R TIER ndbox (Genera N. Virginia (us	Create a Database User A MongoDB user has been added to this project. Net your? Create one in the MongoDB Users tab. You'll need your MongoDB user's credentials in the next step.		Enha
s Set - 3 node REALM APP Jinked	Close Choose a connection method		ated through options,

#### **Connecting to cluster**

You can see three ways of connecting to the cluster on the next screen.



#### Ways of connecting

We will see all these three ways to connect to the cluster in the next sections.

#### Create a Database

We have created a cluster and it's time to create a database. Click on the collections to create a new database as below.

Atlas @ Realm @	Charts		A EL 4
Clusters Q Find a cluster.			Grazile à New Glutter
saveaux     etodo-cluster Version 4.2.1	Operations R: 0 W: 0	- 105.05	Lapital Size 0.0 B
CONNECT METRICS COLLECTIONS	Last 6 Hours	0	+10 B
REGION AWS / N. Vrginia (us-east-1) TYPE Replica Set - 3 nodes	Connections 0	580 mit	Enhance Your Experience For dedicated thosupport, rober metrics and enterprise security eptors, upgrade your classer neal
UNKED REALM APP None Linked	Last 6 Hours	•0	Upgrade

#### Collections

Click on the Add My Own Data Button to create a new database.





#### **Inserting the Document**

A todo-cluster							endak AW:
Overview Real Time	Metrics	Collections	Profiler	Performance Advisor	Online Archive	Command Line Tools	
DATABASES: 1 COLLECTIONS: 1						Let VISU	ALIZI
+ Create Database	tas	ks.todos					
Q, NAMESPACES	COLL	CTION SIZE 08	TOTAL DOCUME	NTS: 0 INDEXES TOTAL SIZE: 4KB			
- tasks	Find	Indexes	Schema	Anti-Patterns 🕜 Aggrega	tion Search Inde	ixes	
todos							
	( CO	("filter":	"example"}				
	QUERY	RESULTS 1-1 OF	1				
		_id: ObjectId!" name: "monel" description: "No createBate: 2020 createBate: 2020 createBate: 2020 updatedBy: "firm	ano 1 descripti 5-88-08T05:00:0 5t last" 20-08-08T05:00:	on" 0.000+00:00			

## Add My Own Data

I have given a database name as tasks and the collection name is todos.

Atlas 🔗 Realm	S Charts	Create Database			
IS ORG - 2020-08-08 > STICKERBUTLER	R > CLUSTERS	DATABASE NAME			
odo-cluster		tasks			4.
erview Real Time	Metrics	COLLECTION NAME	Online	Archive	Command Lin
SES O COLLECTIONS O		todos			
		Capped Collection Before MongoDB can save your new database, a collection name must be specified at the time of creation.			
		Cancel Greate	ita		
		- Find: run queries and interact	with docume	nts	
		- Indexes: build and manage in	dexes		
		- Aggregation: test aggregation	n pipelines		
		- Search: build search indexes			
		Load a Sample Dataset Ad	d My Own Data		



You will see the below dashboard once the database is created. We have a database with empty collections.



#### **Document Inserted**

#### 1) Connect With Mongo Compass

We have seen three ways we can connect to this cluster and read the collections. Let's connect to the database with Mongo Compass. The first thing we need to do is to download and install Mongo Compass from this link.

Let's get a connection string from the Atlas dashboard as below.

S Atias	Connect to todo-cluster		
BHARGAVS CRG - 2020-04-08	Setup connection security     Choose a connection method     Connect		
Q Find a cluster	Choose a connection method View documentation (3		
SANDBOX	Get your pre-formatted connection string by selecting your tool below.		
e todo-cluster Version 4.2,11	Connect with the songo shell Interact with your cluster using MongoDB's interactive Javascript interface	>	
CONNECT METRIC	Connect your application Connect your application to your cluster using MongoDB's native drivers	Ś	_
M0 Sandbox (Genetic REGION AWS / N. Verginia (us	T Overseer year approximent to your means using monitorio of manne writero.		
TYPE Replica Set - 3 node:	Connect using MongoDB Compass Explore, modify, and visualize your data with MongoDB's GUI	>	Enhance Your Experie nted throughput, richer metrics or options, upgrade your cluste
None Linked	Go Back	Close	Upgrade

#### **Connect with MongoDB Compass**

Replace the password with the password that you created above.

#### **Empty Collection**

Let's insert the first document into the collection by clicking the button insert document

## International Journal of Science and Research (IJSR) ISSN: 2319-7064 SJIF (2022): 7.942



#### **Connection String**

Let's connect to the database with the connection string

<ul> <li>Favor</li> <li>Receiption</li> <li>DEC 14</li> <li>cluster</li> </ul>		New Connection (Statement) Fill in connection fields individually	
DEC 14 Cluste	ints		
DEC 14		Pia in connection news mannously	
cluste	4 3030 10 38 88		
	er0.dc3nz.mongodb.net	Click edit to modify your connection string (SRV or Standard ①)	New to Compass a
todo-	/ SECONDS AGO -cluster.zp.kr.mongodb.r	$mongodb+erv: //admin:^{****}  to do-cluster.zpikr.mongodb.net/test?authSource=admin8$	If you don't already h one for free using Me
		EDIT CONNECT	CREATE FREE CLU
			How do I find my o
			If you have an Atlas of Click the 'Connect' by you wish to connect.
			See example
			How do I format my
			See example

**Connect with Connection String** 

You can actually see the same collection in the MongoDB Compass as well.



**MongoDB** Compass

#### 2) Building API

We have configured MongoDB in the previous section, it's time to build the API. I would recommend you go through two articles posted in the prerequisites section. Let me put those here as well.

How to Build NodeJS REST API with Express and MongoDB

How to write production - ready Node. js Rest API — Javascript version

The starting point of the API is the *server. js* file in which we define all the routes and import the express. Here is the file where the nodejs server runs on port 3080 and starts listening for the incoming requests.

1	<pre>const express = require('express');</pre>	
2	<pre>const bodyParser = require('body-parser');</pre>	
3		
4	<pre>const taskController = require('./controller/task.controller')</pre>	
5		
6		
7	<pre>const app = express();</pre>	
8	<pre>const port = process.env.PORT    3080;</pre>	
9		
10	<pre>app.use(bodyParser.json());</pre>	
11		
12	<pre>app.get('/api/tasks', (req, res) =&gt; {</pre>	
13	<pre>taskController.getTasks().then(data =&gt; res.json(data));</pre>	
14	});	
15		
16	<pre>app.post('/api/task', (req, res) =&gt; {</pre>	
17	<pre>console.log(req.body);</pre>	
18	<pre>taskController.createTask(req.body.task).then(data =&gt; res.json(data));</pre>	
19	});	
20		
21	<pre>app.put('/api/task', (req, res) =&gt; {</pre>	
22	<pre>taskController.updateTask(req.body.task).then(data =&gt; res.json(data));</pre>	
23	});	
24		
25	<pre>app.delete('/api/task/:id', (req, res) =&gt; {</pre>	
26	<pre>taskController.deleteTask(req.params.id).then(data =&gt; res.json(data));</pre>	
27	});	
28		
20	ann mat/// (non nor) -> (	
		The v
		<i>P</i> V
32		
33		
34		
35	<pre>app.listen(port, () =&gt; {</pre>	
36	<pre>console.log(`Server listening on the port \${port}`);</pre>	
37	})	
serv	erjs hosted with 🤎 by GitHub	view raw
	an 🖡 - Chanalain - Chanalain - 🔊 - Martina - Sana	
	server.js	

We have defined 4 routes for CRUD operations. Notice that we are using four different HTTP methods for creating, updating, reading, and deleting operations. The request comes to these routes and each route calls the respective method in the controller class. You can read the body of the incoming requests in the req object defined in each route. The result of these methods is promise based so you need to use *then* method to read and send back to the client with the method res. json ().

Here is the controller class in which we are calling the service class with async/await. The async/await is the cleaner way of reading promises. You don't need async/await here since we are directly returning the result of the service class.

	<pre>const taskService = require('/service/task.service');</pre>
1	
2	<pre>const logger = require('/logger/api.logger');</pre>
3	
4	class TodoController {
5	
6	<pre>async getTasks() {</pre>
7	logger.info('Controller: getTasks')
8	<pre>return await taskService.getTasks();</pre>
9	}
10	
11	<pre>async createTask(task) {</pre>
12	<pre>logger.info('Controller: createTask', task);</pre>
13	<pre>return await taskService.createTask(task);</pre>
14	}
15	
16	<pre>async updateTask(task) {</pre>
17	<pre>logger.info('Controller: updateTask', task);</pre>
18	<pre>return await taskService.updateTask(task);</pre>
19	}
20	
21	<pre>async deleteTask(taskId) {</pre>
22	<pre>logger.info('Controller: deleteTask', taskId);</pre>
23	<pre>return await taskService.deleteTask(taskId);</pre>
24	}
25	}
26	<pre>module.exports = new TodoController():</pre>
	Task Controller

Let's look at the service class in which we call the repository to interact with the MongoDB data.



You need to know how to configure Mongo Connection in the NodeJS before looking at the repository so that you can read the data from MongoDB. Let's find that out in the following section.

#### 3) Configure MongoDB In API

Let's configure the Mongo Client from our application. The first thing we need to do is to get the connection string.

Clusters	✓ Setup connection security Choose a connection method Connect		
Q Find a cluster	Choose a connection method View documentation 3 Get your pre-formatted connection string by selecting your tool below.		
e todo-cluster Version 4.2.11	Connect with the mongo shell Interact with your cluster using MongoDB's interactive Javascript interface	>	в
CONNECT METRIC CLUSTER TIER MO Sandbox (General	Connect your application Connect your application to your cluster using MongoDB's native drivers	>	
REGION AWS / N. Virginia (us TYPE Replica Set - 3 nodes	Connect using MongoDB Compass Explore, modify, and visualize your data with MongoDB's GUI	>	Enh: ated through
LINKED REALM APP None-Linked	Go Back	Close	

## **Connect your application**

Make sure you select the right language and the right MongoDB driver version.

✓ Setup connection	security V Choose a connection method Conn	ect
Select your driver	nd version	
DRIVER	VERSION	
Node.js	<ul> <li>3.0 or later</li> </ul>	
Add your connection	n string into your application code	
Include full driver		
mongodb+srv://	dmin: <password>@todo-cluster.zpikr.mongodb.net/&lt;</password>	dbnane>?retryWri db Copy
	> with the password for the <b>admin</b> user. Replace <dbnan by default. Ensure any option params are URL encoded.</dbnan 	se> with the name of the database that
Having trouble connectin	? View our troubleshooting documentation	
Go Back		Close

#### **Connection String**

Let's place the connection string and database name in the application properties file as below. You have to URL encode the password if you have any special characters in the password.

Here is the configuration file in which you connect to MongoDB with the help of the connection string. We are using Mongoose to connect with MongoDB for all the queries. Mongoose makes it easy for you to interact with MongoDB.

1	<pre>const mongoose = require('mongoose');</pre>	1	<pre>const { connect, disconnect } = require('/config</pre>
2	<pre>const logger = require('/logger/api.logger');</pre>	2	<pre>const { Task } = require('/model/task.model');</pre>
3		3	<pre>const logger = require('/logger/api.logger');</pre>
4	<pre>const connect = () =&gt; {</pre>	4	
5		5	class TaskRepository {
6	<pre>const url = "mongodb+srv://Tester123:51FWA19CFZuJe9xF@todo-cluster.dc3nz.mongodb.net/todos?r</pre>	6	
7		7	constructor() {
8	<pre>mongoose.connect(url, {</pre>	8	connect();
9	useNewUrlParser: true,	9	)
10	useFindAndModify: true,	10	
11	useUnifiedTopology: true,	11	<pre>async getTasks() {</pre>
12	useCreateIndex: true,	12	<pre>const tasks = await Task.find({});</pre>
13	})	13	<pre>console.log('tasks:::', tasks);</pre>
14		14	return tasks;
15	<pre>mongoose.connection.once("open", async () =&gt; {</pre>	15	}
16	logger.info("Connected to database");	16	
17	});	17	<pre>async createTask(task) {</pre>
18		18	<pre>let data = {};</pre>
19	<pre>mongoose.connection.on("error", (err) =&gt; {</pre>	19	try {
20	logger.error("Error connecting to database ", err);	20	<pre>data = await Task.create(task);</pre>
21	});	21	<pre>} catch(err) {</pre>
22	)	22	<pre>logger.error('Error::' + err);</pre>
23		23	}
24	<pre>const disconnect = () =&gt; {</pre>	24	return data;
25		25	}
26	if (!mongoose.connection) {	26	
27	return;		
28	}		
29		29	try {
		30	<pre>data = await Task.updateOne(task);</pre>
		31	<pre>} catch(err) {</pre>
32	<pre>mongoose.once( close , async () =&gt; {</pre>	32	logger.error('Error::' + err);
33	console.log("Diconnected to database");	33	}
34	});	34	return data;
35		35	}
36	};	36	<i>8</i>
37		37	<pre>async deleteTask(taskId) {</pre>
38	<pre>module.exports = {</pre>	38	<pre>let data = {};</pre>
39	connect,	39	try {
40	disconnect	40	<pre>data = await Task.deleteOne({_id : tas</pre>
41	}	41	<pre>} catch(err) {</pre>
4.6	• • • • • • • • • • • • • • • • • • •	42	<pre>logger.error('Error::' + err);</pre>
db.c	onfig.js hosted with 😻 by GitHub view raw	42	}
	DD Configuration File	43	<pre>return {status: `\${data.deletedCount &gt; 0 }</pre>
	DB Configuration File	-444	and a second standard and a second second

The next thing we should define is the schema for the database model as below.



Finally, we have a repository class as below using the above model for the CRUD operations.



require('../config/db.config');

Task Repository

#### 4) Externalize the Environment Variables

We have seen how to configure your MongoDB connection in the API. We need to store this kind of configuration outside of your app so that you can build once and deploy it in multiple environments with ease.

We need to use the **dotenv** library for environment - specific things. Dotenv is a zero - dependency module that loads environment variables from a. env file into process. env. Storing configuration in the environment separate from code is based on The Twelve - Factor App methodology.

The first step is to install this library npm install dotenv and put the. env file at the root location of the project.

1 MONGO\_COMMECTION\_STRING=mongodb+srv://Tester123:51FWA19CFZU29xf@todo-cluster.dc3nz.mongodb.net/tc env hosted with ♥ by GitHub view raw .env file

We just need to put this line require ('dotenv'). config () as early as possible in the application code as in the server. js file.

Let's define the configuration class where it creates a connection with the connection string we just copied from the Atlas Dashboard. We are fetching the Mongo connection string with the dotenv library and connecting it to MongoDB with Mongoose. We are exposing two functions from this file connect and disconnect.



Configuration with dotenv

#### 5) Building UI

Once you create the separate folder for the UI code you need to start with the following command to scaffold the Vue structure with the help of the Vue CLI Service. We will not build the entire app here instead we will go through important points here. You can clone the entire GitHub Repo and check the whole app.

vue create ui

Here are the main. js, App. vue, and Home. vue files for the app as the bootstrap components which means this is the first component that loads in the browser. You can import all the CSS - related files in the Home. vue file.

з	<template> </template>
4	
6	<script> export default {</td></tr><tr><td>7</td><td>name: 'App'</td></tr><tr><td>8</td><td>}</td></tr><tr><td>9</td><td></script>
Арр	.vue hosted with 🤎 by GitHub view raw
1	<template></template>
2	<div class="App" id="app"> <header></header></div>
4	<pre><div class="container mrgnbtm"></div></pre>
5	<div class="row"></div>
6	<div class="col-md-12"></div>
7	<createtask @createtask="taskCreate(\$event)"></createtask>
9	
10	
11	<pre><div class="row mrgnbtm"></div></pre>
12	<tasks :tasks="tasks" <="" @deletetask="taskDelete(\$event)" @edittask="" div="" v-if="tasks.length &gt; 0"></tasks>
13	
15	
16	
17	<script></td></tr><tr><td>18 19</td><td><pre>import 'bootstrap/dist/css/bootstrap.css' import 'bootstrap-vue/dist/bootstrap-vue.css'</pre></td></tr><tr><td>20</td><td></td></tr><tr><td>21</td><td><pre>import Header from './Header.vue'</pre></td></tr><tr><td>22</td><td><pre>import CreateTask from './CreateTask.vue'</pre></td></tr><tr><td>23</td><td><pre>import Tasks from './Tasks.vue'</pre></td></tr><tr><td>25</td><td><pre>import { getAllTasks, createTask, deleteTask, editTask } from '/services/TodoService'</pre></td></tr><tr><td>26</td><td></td></tr><tr><td>27</td><td>console.log('Home')</td></tr><tr><td>28 29</td><td>export default {</td></tr><tr><td>30</td><td>name: 'App',</td></tr><tr><td>31</td><td>components: {</td></tr><tr><td>32</td><td>Header,</td></tr><tr><td>33 34</td><td>CreateTask, Tasks</td></tr><tr><td>34</td><td>145K5 },</td></tr><tr><td>36</td><td>data() {</td></tr><tr><td>37</td><td>return (</td></tr><tr><td>38 39</td><td>tasks: [], settings: false</td></tr><tr><td>40</td><td><pre>settings: taise }</pre></td></tr><tr><td>41</td><td>Ъ</td></tr><tr><td>42</td><td>methods: (</td></tr><tr><td>43</td><td><pre>taskCreate(data) {     console.log('data:::', data)</pre></td></tr><tr><td>45</td><td><pre>createTask(data).then(response => {</pre></td></tr><tr><td>46</td><td>console.log(response)</td></tr><tr><td>47</td><td><pre>this.getAllTasks();</pre></td></tr><tr><td>48 49</td><td>}); },</td></tr><tr><td>50</td><td>}, getAllTasks() {</td></tr><tr><td>51</td><td><pre>getAllTasks().then(response => {</pre></td></tr><tr><td>52</td><td>console.log(response)</td></tr><tr><td>53</td><td>this.tasks = response</td></tr><tr><td>54</td><td>)) },</td></tr><tr><td>56</td><td>}, taskDelete(taskId) {</td></tr><tr><td>57</td><td><pre>deleteTask(taskId).then(response => {</pre></td></tr><tr><td>58</td><td>console.log(response)</td></tr><tr><td>59 60</td><td><pre>this.getAllTasks(); });</pre></td></tr><tr><td>60</td><td>}); });</td></tr><tr><td>62</td><td>taskEdit(task) {</td></tr><tr><td>63</td><td><pre>editTask(task).then(res => {</pre></td></tr><tr><td></td><td>console.log(res);</td></tr><tr><td>64</td><td></td></tr><tr><td>64 65</td><td><pre>this.getAllTasks();</pre></td></tr><tr><td>64</td><td></td></tr><tr><td>64 65 67 68</td><td><pre>this.getAllTasks(); }) },</pre></td></tr><tr><td>64 65 67 68 69</td><td><pre>this.getAllTasks(); )) ), mounted () (</pre></td></tr><tr><td>64 65 67 68 69 70</td><td><pre>this.getAllTasks(); )) }, mounted () { this.getAllTasks(); </pre></td></tr><tr><td>64 65 67 68 69 70 71</td><td><pre>this.getAllTasks(); )) } , mounted () { this.getAllTasks(); } </pre></td></tr><tr><td>64 65 67 68 69 70 71</td><td><pre>this.getAllTasks(); )) , mounted () {    this.getAllTasks(); }</pre></td></tr><tr><td>64 65 67 68 69 70 71 72</td><td><pre>this.getAllTasks(); )) ), mounted () ( this.getAllTasks(); ) ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ; ;</td></tr><tr><td>64 65 67 68 69 70 71 72 72</td><td><pre>this.getAllTasks(); )) , mounted () (    this.getAllTasks(); ) , //contents</pre></td></tr><tr><td>64 65 66 67 68 69 70 71 72 72 72 72 72</td><td><pre>this.getAllTasks(); )) , mounted () (     this.getAllTasks();     } } //reflats @lmport */assets/styles/global.css'; </style></pre></td></tr><tr><td>64 65 66 67 68 69 70 71 72 72 72 72 72 74</td><td><pre>this.getAllTasks(); } } mounted () { this.getAllTasks(); } } //contint> //contint=/contin</td></tr><tr><td>64 65 66 67 68 69 70 71 72 73 72 73 76 77 ◀ ◀</td><td><pre>this.getAllTasks(); } , mounted () { this.getAllTasks(); } //creint*  @import '/assets/styles/global.css'; c/style> nevue hosted wilh  by GitHub view raw </pre></td></tr><tr><td>64 65 66 70 71 72 73 76 77 ₹ 4 € Hon 1</td><td><pre>this.getAllTasks(); )) ), mounted () { this.getAllTasks(); } );  #unort '/assets/styles/global.css'; #unort '/assets/styles/global</td></tr><tr><td>64 65 66 67 68 69 70 71 72 73 73 74 4 Hom 1 2</td><td><pre>this.getAllTasks(); }) }, mounted () { this.getAllTasks(); } } //creates @import '/assets/styles/global.css'; </tyle> nevue hosted with  by GitHub view raw import Yue from '.vue' </pre></td></tr><tr><td>64 65 66 70 71 72 73 76 77 ₹ 4 € Hon 1</td><td><pre>this.getAllTasks(); )) ), mounted () { this.getAllTasks(); } );  #disport '/assets/styles/global.css'; e/styles newue hosted with  by GitHub view raw isport Yue from 'uue'</pre></td></tr><tr><td>64 65 66 67 68 69 70 71 72 73 73 74 Hon 1 2 3</td><td><pre>this.getAllTasks(); }) }, mounted () { this.getAllTasks(); } } //creates @import '/assets/styles/global.css'; </tyle> nevue hosted with  by GitHub view raw import Yue from '.vue' </pre></td></tr><tr><td>64 65 66 69 70 71 72 73 73 74 40 Hon 1 2 3 4 5 6</td><td><pre>this.getAllTasks(); }) }, mounted () { this.getAllTasks(); } }  //creines  //creines //cre</td></tr><tr><td>64 65 66 67 68 69 70 71 72 73 76 76 77 77 74 4 Hor 1 2 3 4 5 6 7</td><td><pre>this.getAllTasks(); )) ), mounted () ( this.getAllTasks(); ) //creints //creints //creints //ereints //ereints</td></tr><tr><td>64 65 66 67 70 71 72 73 70 71 72 73 74 4 Hor 1 2 3 4 5 6 77 8</td><td><pre>this.getAllTasks(); }) }, mounted () { this.getAllTasks(); } } //creates @lipport '/assets/styles/global.css'; </style>  @lipport '/assets/styles/global.css'; </style>  @lipport voue from './App.vue' lisport App from './App.vue' lisport pouter from './App.vue' lisport router from './App.vue' lisport in -> h(App), </pre></td></tr><tr><td>64 65 66 67 68 69 70 71 72 73 76 76 77 77 74 4 Hor 1 2 3 4 5 6 7</td><td><pre>this.getAllTasks(); )) ), mounted () ( this.getAllTasks(); ) //creints //creints //creints //ereints //ereints</td></tr></tbody></table></script>

App and Home Compo

#### International Journal of Science and Research (IJSR) ISSN: 2319-7064 SJIF (2022): 7.942

The App. vue component is the first component that loads since it is defined in the main. js file. We have a router defined in the App component that loads the Home component. This is a simple application where you add, update, delete tasks. You can go through the GitHub repo to check the rest of the files.

We have another two important components here one is for the createTask Form component and another is for the Tasks table.

1 2			
~	<pre><tomplato> <div class="container"></div></tomplato></pre>		
3	<div class="row"></div>		
4	<div class="col-md-12 mrgnbtm"> <h2>TODo List</h2></div>		
6 7	<form> <div class="row"></div></form>		
8	<pre><div class="form-group col-md-6"></div></pre>		
10	<label htmlfor="exampleInputEmail1">Task</label> <input <br="" class="form-control" id="task" name="task" type="text" v-model="task"/>		
11 12	<div class="form-group col-md-6"></div>		
13	<label htmlfor="exampleInputPassword1">Assignee</label>		
14 15	<pre><input class="form-control" ic<br="" name="assignee" type="text" v-model="assignee"/></pre>		
16 17	<div class="row"></div>		
18	<pre><div class="form-group col-md-12"></div></pre>		
19 20	<label htmlfor="exampleInputEmail1">Status:</label> <select class="form-control" id="sel1" name="status" v-model="status"></select>		
21	<pre><option>To Be Done</option> <option>In Progress</option></pre>		
23	<option>Completed</option>		
24 25			
26 27	 <button @click="createTask()" class="btn btn-danger" type="button">Create</button>		
28			
29 30			
31 32			
33			
34 35	<script> export default (</td></tr><tr><td>36 37</td><td>name: 'CreateTask', data() (</td></tr><tr><td>38</td><td>aata() ( neturn ( task: '',</td></tr><tr><td>39 40</td><td>assignee: '',</td></tr><tr><td>41 42</td><td>status: ''</td></tr><tr><td>43</td><td>3.</td></tr><tr><td>44 45</td><td><pre>methods: {     createTask() {</pre></td></tr><tr><td>46</td><td><pre>const payload = {     task: this.task,</pre></td></tr><tr><td>48</td><td>assignee: this.assignee,</td></tr><tr><td>49 50</td><td>status: this.status )</td></tr><tr><td>51</td><td>this.\$emit('createTask', payload)</td></tr><tr><td>52 53</td><td><pre>this.clearForm(); };</pre></td></tr><tr><td>54 55</td><td><pre>clearForm() {    this.task = "";</pre></td></tr><tr><td>56</td><td>this.status = ""; this.status = "";</td></tr><tr><td>57 58</td><td>)</td></tr><tr><td>59 60</td><td>}</td></tr><tr><th>61</th><th></script>		
Crea	teTask.vue hosted with 🎔 by GitHub view raw		
1	<template></template>		
1 2 3	<template> <div class="container"> <d>27akks&gt;</d></div></template>		
1 2	<template> <div class="container"></div></template>		
1 2 3 4 5 6	<templato> class="container"&gt; class="container" class="container" class="container" class="container" class="container"</templato>		
1 2 3 4 5	<pre>ctemplates cdip class="container"&gt; cd2FTacksc/hb&gt; cd2btacksc/hb&gt; cd2btacksc-td2bt table-bordered"&gt; cd2btacksc-td2btable-bordered"&gt; cd2btackst-td2btable-bordered"&gt; cd2btackst-td2btable-bordered"&gt; cd2btackst-td2btable-bordered"&gt; cd2btackst-td2btable-bordered"&gt; cd2btackst-td2btable-bordered"&gt; cd2btackst-td2btable-bordered"&gt; cd2btackst-td2btable-bordered"&gt; cd2btable-bordered cd2btable-bor</pre>		
1 2 3 4 5 6 7 8 9	<template> <div class="container"> </div> <ii class="container"> <i class="container"> </i> </ii></template>		
1 2 4 5 6 7 8 9	<template> <idiv class="container"> <idiv class="container"> <idiv class="container"> <idiv class="table table-bordered"> <idiv class="table table-bordered"> <idiv class="table table-bordered">   cdsvclass="container"&gt; cdsvclass="container" cdsvc</idiv></idiv></idiv></idiv></idiv></idiv></template>		
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19	<templates <pre>stars="container"&gt;</pre></templates 		
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 9 20	<teplates <type:lass="container"></type:lass="container"></teplates 		
1 2 3 4 5 6 7 8 9 10 111 12 13 14 15 16 17 18 19 20 21 22	<tepplates <tepplates< td=""></tepplates<></tepplates 		
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 19 20 22 23 24	<pre>ctepjoto</pre>		
1 2 3 4 5 6 7 8 9 101 1112 131 1112 1112 1112 1200 2212 222 222 222 222 222 222 225	<templates stass=container&gt;&gt; ctable class="container"&gt; ctable class="container</templates 		
1 3 4 5 6 7 8 9 10 11 12 13 14 15 6 7 8 9 10 11 12 22	<pre>ctapjoto cdapicas="container"&gt; cdapicas="container"&gt; cdapicas="container"&gt; cdapicas="container"&gt; cdapicas="container"&gt; cdapicas="container"&gt; cdapicas="container"&gt; cdapicade container"&gt; cdapicad</pre>		
1 3 4 5 6 7 8 9 10 11 12 13 14 15 6 7 8 9 10 11 12 13 14 15 10 22	<teplots stars="container"&gt; stars="container</teplots 		
1 3 4 5 6 7 8 9 101 111 12 131 141 15 161 17 120 221 222 223 225 222 225 222 225 222 225 222 225 222 225 222 225 235 335	<teplates <typelas="container"></typelas="container"></teplates 		
$\begin{smallmatrix} 1 \\ 2 \\ 3 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 10 \\ 111 \\ 12 \\ 131 \\ 15 \\ 10 \\ 111 \\ 120 \\ 222 \\ 24 \\ 222 \\ 220 \\ 222 \\ 220 \\ 301 \\ 322 \\ 332 \\$	<pre>steplats divclass=containe"&gt; divclass=containe"&gt; divclass=containe"&gt; divclass=containe"&gt; divclass=containe"&gt; divclass=contered divcla</pre>		
$     \begin{array}{c}       1 \\       2 \\       3 \\       4 \\       5 \\       6 \\       7 \\       8 \\       9 \\       0 \\       10 \\       11 \\       12 \\       13 \\       14 \\       15 \\       16 \\       17 \\       18 \\       12 \\       22 \\       23 \\       22 \\       22 \\       22 \\       22 \\       22 \\       22 \\       22 \\       22 \\       23 \\       22 \\       22 \\       23 \\       22 \\       23 \\       22 \\       23 \\       22 \\       23 \\       22 \\       23 \\       22 \\       23 \\       23 \\       22 \\       23 \\       22 \\       23 \\       23 \\       22 \\       23 \\       23 \\       23 \\       23 \\       23 \\       23 \\       23 \\       23 \\       23 \\       23 \\       23 \\       23 \\       23 \\       23 \\       23 \\       31 $	<tepplates stass=container"&gt; cd2/rasks/hb2 cd2/rasks/hb</tepplates 		
$\begin{smallmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 10 \\ 111 \\ 123 \\ 14 \\ 15 \\ 16 \\ 17 \\ 18 \\ 10 \\ 222 \\ 23 \\ 4 \\ 25 \\ 6 \\ 7 \\ 8 \\ 9 \\ 10 \\ 111 \\ 123 \\ 33 \\ 33 \\ 35 \\ 135 \\ 10 \\ 10 \\ 10 \\ 10 \\ 10 \\ 10 \\ 10 \\ 1$	<tepplates templates tables="container"&gt; tables="container"&gt; tables="container"&gt; table</tepplates 		
$\begin{smallmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 0 \\ 10 \\ 111$	<pre>ctmplots divclass="container"&gt; divclass</pre>		
1 2 3 4 5 6 7 8 9 0 1112 1314 1112 1113 1112 1113 1112 1113 1112 1113 1112 1113 1112 112 112 112 112 112 112 112 112 112 112	<pre>stepiots stass=container&gt;&gt; did&gt;fass=container&gt;&gt; did&gt;fass=container&gt;&gt; did&gt;fass=container&gt;&gt; did&gt;fass=container&gt;&gt; did&gt;fass=container&gt;&gt; did&gt;fass=container&gt;&gt; did&gt;fass=container&gt;&gt; did&gt;fass=fast=fass=fast=fass=fast=fast=fast=</pre>		
$\begin{smallmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 0 \\ 1112 \\ 13 \\ 115 \\ 16 \\ 7 \\ 100 \\ 122 \\ 222 \\ 222 \\ 222 \\ 222 \\ 220 \\ 101 \\ 222 \\ 220 \\ 101 \\ 222 \\ 220 \\ 101 \\ 222 \\ 220 \\ 101 \\ 2$	<pre>staplats disclass="container"&gt; disclass="container" disclass="container"&gt; disclass="container" disclass="container"&gt; disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" discl</pre>		
$\begin{smallmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 0 \\ 0 \\ 11 \\ 12 \\ 13 \\ 14 \\ 15 \\ 6 \\ 7 \\ 10 \\ 0 \\ 12 \\ 22 \\ 22 \\ 22 \\ 22 \\ 22 $	<pre>ctmplots divclass="container"&gt; divclass</pre>		
1 2 3 4 5 6 7 8 9 10 1112 131 115 1112 112 112 112 112 112 112 11	<pre>steplates states=Container&gt;&gt; dd33Tanks(Ab2 dd33Tanks(</pre>		
$\begin{smallmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 0 \\ 0 \\ 11123 \\ 1456 \\ 7 \\ 180 \\ 0 \\ 222 \\ 223 \\ 4 \\ 566 \\ 7 \\ 8 \\ 0 \\ 0 \\ 1123 \\ 145 \\ 6 \\ 7 \\ 100 \\ 122 \\ 223 \\ 225 \\ 222 \\ 220 \\ 133 \\ 333 \\ 333 \\ 333 \\ 333 \\ 333 \\ 401 \\ 223 \\ 444 \\ 444 \\ 445 \\ 100 \\ $	<pre>steplats div class="container"&gt; div class="container" div class="container"&gt; div class="container"</pre>		
1 2 3 4 5 6 7 8 9 10 111 123 141 15 167 221 222 223 225 227 220 331 332 333	<pre>ctmplots divclass="container"&gt; divclass</pre>		
$\begin{smallmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1$	<pre>ctmplots div=class="container"&gt; div=clas</pre>		
$\begin{smallmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 0 \\ 0 \\ 10112 \\ 1131 \\ 15 \\ 16 \\ 7 \\ 100 \\ 1222 \\ 222 \\ 222 \\ 222 \\ 222 \\ 222 \\ 222 \\ 222 \\ 222 \\ 222 \\ 223 \\ 333 $	<pre>steplats (daylass=container"&gt;     diviss="container"&gt;     diviss="contain</pre>		
$\begin{smallmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 0 \\ 0 \\ 11123 \\ 14 \\ 15 \\ 6 \\ 7 \\ 8 \\ 0 \\ 0 \\ 11123 \\ 14 \\ 15 \\ 6 \\ 7 \\ 10 \\ 12 \\ 22 \\ 23 \\ 4 \\ 22 \\ 22 \\ 22 \\ 22 \\ 22$	<pre>steplats (dayse=Container"&gt;     divelens="Container"&gt;     divelens="Container"&gt;     divelens="Container"&gt;     divelens="Container"&gt;     divelens="Container"&gt;     divelens="Container"&gt;     divelens="Container"&gt;     divelens="Container"&gt;         divelens="Container"&gt;</pre>		
$\begin{smallmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 0 \\ 0 \\ 11 \\ 12 \\ 13 \\ 14 \\ 15 \\ 6 \\ 7 \\ 8 \\ 0 \\ 0 \\ 11 \\ 12 \\ 13 \\ 14 \\ 15 \\ 6 \\ 7 \\ 18 \\ 0 \\ 0 \\ 12 \\ 22 \\ 23 \\ 4 \\ 22 \\ 22 \\ 22 \\ 22 \\ 22$	<pre>steplets divclass="container"&gt; divclass</pre>		
$\begin{smallmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 0 \\ 0 \\ 11 \\ 12 \\ 11 \\ 12 \\ 11 \\ 11 \\$	<pre>stupplets divclass="container"&gt; divclass="container" divclass="container" divclass="container" divclass="container" divclass="container" divclass="container" divclass="container" divclass="container" divclass="container" divclass="container" divclass="container" divclass="container" divclass="container" divclass="container" divclass="container" divclass="container" divclass="container" divclass="container" div</pre>		
$\begin{smallmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 0 \\ 0 \\ 11113 \\ 11111 \\ 11111 \\ 11111 \\ 11111 \\ 11111 \\ 11111 \\ 11111 \\ 11111 \\ 11111 \\ 1111$	<pre>stup:lts; disclass="container"&gt; disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disc</pre>		
$\begin{smallmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1$	<pre>templates     dive inser*container*&gt;     dive inser*container*:     dive inser*container*:     dive inser*container*:     import fourter from */fourtes*     import fourter*(instrainer*, inskid)     j,     insertificaine(inskid) (         insertificainet*:         insertifica</pre>		
$\begin{smallmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 0 \\ 0 \\ 1112 \\ 111$	<pre>stup:lts; disclass="container"&gt; disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disclass="container" disc</pre>		

Run the Vue code in local with the following command which runs on port **8080** on localhost. Make sure you are in the root folder of Vue code which is todo - app here.

cd ui		
npm run serve		
← → ♂ ④ localhost:8080	û 🖈 🏶 🕲 Upda	• •
MEVN	Stack Example	
ToDo List Task	Assignee	
Create a Task	Assignee	
Status:		
	~	
Create		

Vue Code running on port 8080

#### 6) Make API Calls From UI

Here is the service file which calls the API, in this case. We have four API operations to get, add, edit, and delete tasks with root path /api.

1	<pre>export async function getAllTasks() {</pre>	
2		
з	<pre>const response = await fetch('/api/tasks');</pre>	
4	<pre>return await response.json();</pre>	
5	}	
6		
7	<pre>export async function createTask(data) {</pre>	
8	<pre>const response = await fetch(`/api/task`, {</pre>	
9	method: 'POST',	
10	<pre>headers: {'Content-Type': 'application/json'},</pre>	
11	<pre>body: JSON.stringify({task: data})</pre>	
12	})	
13	<pre>return await response.json();</pre>	
14	}	
15		
16	<pre>export async function deleteTask(taskId) {</pre>	
17	<pre>const response = await fetch('/api/task/\${taskId}', {method: 'DELETE'})</pre>	
18	<pre>return await response.json();</pre>	
19	}	
20		
23	method: 'PUI',	
24	<pre>headers: {'Content-Type': 'application/json'},</pre>	
25	<pre>body: JSON.stringify({task: data})</pre>	
26	3)	
27	<pre>return await response.json();</pre>	
28	}	
Tod	oService.js hosted with 🤎 by GitHub	view raw
	TodoService.is	
	1000001100.10	

From the Vue components, you can call this service to get the data as below. Here is an example.

1	<template></template>
2	<pre><div class="App" id="app"></div></pre>
З	<header></header>
4	<div class="container mrgnbtm"></div>
5	<div class="row"> <div class="col-md-12"></div></div>
7	<pre><createtask @createtask="taskCreate(\$event)"></createtask></pre>
8	
9	
10	
11 12	<div class="row mrgnbtm"> <tasks :tasks="tasks" @deletetask="taskDelete(\$event)" @edittask="&lt;/td" v-if="tasks.length &gt; 0"></tasks></div>
13	
14	
15	
16	
17 18	<script> import 'bootstrap/dist/css/bootstrap.css'</td></tr><tr><td>19</td><td><pre>import 'bootstrap/ust/css/bootstrap.vss import 'bootstrap-vue/dist/bootstrap-vue.css'</pre></td></tr><tr><td>20</td><td></td></tr><tr><td>21</td><td>import Header from './Header.vue'</td></tr><tr><td>22</td><td><pre>import CreateTask from './CreateTask.vue'</pre></td></tr><tr><td>23</td><td><pre>import Tasks from './Tasks.vue'</pre></td></tr><tr><td>24 25</td><td><pre>import { getAllTasks, createTask, deleteTask, editTask } from '/services/TodoService'</pre></td></tr><tr><td>25</td><td>import { getAillasks, createlask, deletelask, editlask } from/services/iodoservice</td></tr><tr><td>27</td><td>console.log('Home')</td></tr><tr><td>28</td><td></td></tr><tr><td>29</td><td>export default {</td></tr><tr><td>30</td><td>name: 'App',</td></tr><tr><td>31 32</td><td>components: { Header,</td></tr><tr><td>33</td><td>reauer, CreateTask,</td></tr><tr><td>34</td><td>Tasks</td></tr><tr><td>35</td><td>},</td></tr><tr><td>36</td><td>data() {</td></tr><tr><td>37</td><td>return {</td></tr><tr><td>38 39</td><td>tasks: [],</td></tr><tr><td>39 40</td><td><pre>settings: false }</pre></td></tr><tr><td>41</td><td>b.</td></tr><tr><td>42</td><td>methods: {</td></tr><tr><td>43</td><td><pre>taskCreate(data) {</pre></td></tr><tr><td>44</td><td><pre>console.log('data:::', data)</pre></td></tr><tr><td>45 46</td><td><pre>createTask(data).then(response => {     console.log(response)</pre></td></tr><tr><td>40</td><td>this.getAllTasks();</td></tr><tr><td>48</td><td>});</td></tr><tr><td>49</td><td>},</td></tr><tr><td>50</td><td>getAllTasks() {</td></tr><tr><td>51</td><td><pre>getAllTasks().then(response => {</pre></td></tr><tr><td>52 53</td><td><pre>console.log(response) this.tasks = response</pre></td></tr><tr><td>54</td><td><pre>})</pre></td></tr><tr><td></td><td>)</td></tr><tr><td></td><td>Z v</td></tr><tr><td></td><td></td></tr><tr><td>58</td><td>console.log(response)</td></tr><tr><td>59 60</td><td><pre>this.getAllTasks(); });</pre></td></tr><tr><td>61</td><td>};</td></tr><tr><td>62</td><td>taskEdit(task) {</td></tr><tr><td>63</td><td><pre>editTask(task).then(res => {</pre></td></tr><tr><td>64</td><td><pre>console.log(res);</pre></td></tr><tr><td>65</td><td><pre>this.getAllTasks();</pre></td></tr><tr><td>66 67</td><td>)) }</td></tr><tr><td>68</td><td>},</td></tr><tr><td>69</td><td>mounted () {</td></tr><tr><td>70</td><td><pre>this.getAllTasks();</pre></td></tr><tr><td>71</td><td>}</td></tr><tr><td>72 73</td><td>} </script>
73	A act they
75	<style></td></tr><tr><td>76</td><td><pre>@import '/assets/styles/global.css';</pre></td></tr><tr><td>77</td><td></style>
< •	e.vue hosted with ♥ by GitHub view raw
nom	e.vue hosted with 💙 by GitHub view raw
	Home Component

You can look at the below article for a detailed post. *How To Make API calls in Vue. JS Applications* 

#### 7) Development Environment Setup

Usually, the way you develop and the way you build and run in production are completely different.

In the development phase, we run the nodejs server and the Vue app on completely **different ports.** It's easier and faster to develop that way. If you look at the following diagram the Vue app is running on port **8080** with the help of a webpack dev server and the nodejs server is running on port **3080**.



#### **Development Environment**

There should be some interaction between these two. We can proxy all the API calls to nodejs API. Vue - cli - service provides some inbuilt functionality and tells the development server to proxy any unknown requests to your API server in development, you need to add a **vue. proxy. js** file at the root of the location where package. json resides. We need to add the following file.



Now you can run both Vue UI and NodeJS API on different ports and the Vue Code interacts with the API.

<pre>// Vue Code cd ui npm install npm run serve // API code cd api npm install npm run dev</pre>		
← → C ① localhost.8080		0 x 🖈 🇐 (Update :)
N	IEVN Stack Example	(k)         )         Dements         Concie         Sources         Network         >>         (k)
TODO List Tak Create a Tak I Beter Create	Assignee	Beodrig information activity      Beodrig information activity      Beodrig information activity      Beodrig information activity      Recording informatio

#### Network Calls

#### 8) Running on Docker Compose

Docker Compose is really useful when we don't have the development environment setup on our local machine to run all parts of the application to test or we want to run all parts of the application with one command. For example, if you want to run NodeJS REST API and MongoDB database on different ports and need a single command to set up and run the whole thing. You can accomplish that with Docker Compose.

In the below post, we will see what is Docker Compose and how we can do the local development of MEVN Stack with Docker Compose, and its advantages as well. Coming Soon!! You can check out other stacks here *How To Run MERN Stack on Docker Compose* 

How To Run MEAN Stack on Docker Compose

## 9) Dockerize MEVN Stack

Docker is an enterprise - ready container platform that enables organizations to seamlessly build, share, and run any application, anywhere. Almost every company is containerizing its applications for faster production workloads so that they can deploy anytime and sometimes several times a day. There are so many ways we can build a MEVN Stack. One way is to dockerize it and create a docker image so that we can deploy that image any time or sometimes several times a day.

In the below post, we look at the example project and see the step - by - step guide on how we can dockerize the MEVN Stack.

#### Coming Soon !!

You can check out other stacks here How To Dockerize MEAN Stack How To Dockerize MERN Stack

#### a) Linting

We need to lint our project in that way it's easier to follow some standards in your project. We will see this in a separate article.

Coming Soon!!

#### b) Unit Testing API

There are so many tools out there to unit test your application such as Mocha, Chai, etc. We need a separate article for that to cover different libraries. Coming Soon!!

#### c) Unit Testing UI

We will see how to unit test with UI with jest library. Coming Soon!

#### d) Integration Tests

We will use cypress for the integration tests. Coming Soon!

#### e) Build for production

We have to build the project for production in a different way. We can't use the proxy object. Here is the detailed article on how to package your app for production.

Packaging Your Vue. js App With NodeJS Backend For Production

#### How to Build MEVN Stack for Production

## f) Demo

Here is an example of a simple tasks application that creates, retrieves, edits, and deletes tasks. We actually run the API on the NodeJS server and you can use MongoDB to save all these tasks.

localhost:8080		I				0 ¥	* 🕲 😡
		MEVN Sta	ack Example	•			
ToDo List							
Task			Assignee				
Create a Task			Assignee				
Status:							
						~	
Create							
Create Tasks							
	Tasi	< Name	Assignee	Status			
Tasks		< Name df131232133123	Assignee sdfsdf55	Status To Be Done	Edit	Delete	

#### **Example Project**

As you add users we are making an API call to the nodejs server to store them and get the same data from the server when we retrieve them. You can see network calls in the following video.

← → C (① localhost:8080					0 🛠 🕯	· 🕲 💽	ipdate		
MEVN Stack Example			Q, Console	ng   🗌 Disable	cache No throttling		0 :	×	
ToDo List		Filter   Filter Filter  Filter Filter  Filter  Filter  Filter  Filter  Filter  Filter  Filter  Filter  Filter  Filter  Filter  Filter  Filter  Filter							
Task	Assignee	20 m	s 40 ms	60 m	80 ms		100 ma		
Create a Task I	Assignee								
Status:									
	~ ~								
Create									
			Pac	ording network	activity				
				est or hit # R	to record the rela	ad.			
				Learn mor	•				

#### **Network Calls**

# 5. Summary

- There are so many ways we can build Vue apps and ship them for production.
- One way is to build the Vue app with NodeJS and MongoDB as a database. There are four things that make this stack popular and you can write everything in Javascript.
- The four things are MongoDB, VueJS, Express, and NodeJS. This stack can be used for a lot of uses cases in web development.
- We will have two package. json: one for the **VueJS** and another for **nodejs API**. It's always best practice to have completely different node\_modules for each one.
- The core of MongoDB Cloud is MongoDB Atlas, a fully managed cloud database for modern applications. Atlas is the best way to run MongoDB, the leading modern database.
- We need to use the **dotenv** library for environment specific things. Dotenv is a zero dependency module that loads environment variables from a. env file into process. env. Storing configuration in the environment separate from code is based on The Twelve Factor App methodology.
- In the development phase, we run the nodejs server and the **Vue** app on completely **different ports.** It's easier and faster to develop that way.
- We need to lint our project in that way it's easier to follow some standards in your project.

# Volume 13 Issue 9, September 2024

# Fully Refereed | Open Access | Double Blind Peer Reviewed Journal

www.ijsr.net

- There are so many tools out there to unit test the API such as Mocha, Chai, etc.
- We can unit test with UI with jest library.
- We will use cypress for the integration tests.
- We have to build the project for production in a different way. We can't use the proxy object.

# 6. Conclusion

The MEVN stack provides a powerful framework for building modern web applications, offering flexibility and efficiency through JavaScript. This article has demonstrated the setup, development, and deployment of a MEVN stack application, highlighting various automation techniques. Future studies should explore further optimizations and deployment strategies across different cloud platforms to enhance scalability and performance.

# 7. References

- JavaScript Documentation https://developer.mozilla.org/en-US/docs/Learn/Getting\_started\_with\_the\_web/JavaScript\_ basics
- [2] VueJS
- Documentation https://vuejs.org/guide/introduction.html
- [3] Bachina, Bachina. (2024). \*Ultimate Full Stack Web Development with MEVN\*. OrangeAVA. Available at https://orangeava.com/collections/allebooks/products/ultimate-full-stack-web-developmentwith-mevn?variant=45552578560218
- [4] Bhargav Bachina. (2021). Optimizing Deployment: React with NodeJS Backend on Azure App Services. Journal of Scientific and Engineering Research, 8(4), 218– 227. https://doi.org/10.5281/zenodo.10902911