1) Intro to Vue.JS

* Vue is a progressive framework for building user interfaces.
* Mostly used for Single Page Applications (S.P.As).

**2) Installation of Vue.JS**

* Requirements (Node.Js Latest Version ,NPM)

## npm i -g @vue/cli

* + *for install Vue Cli Globally*

## vue create app\_name

* + *Creates a New vue App*

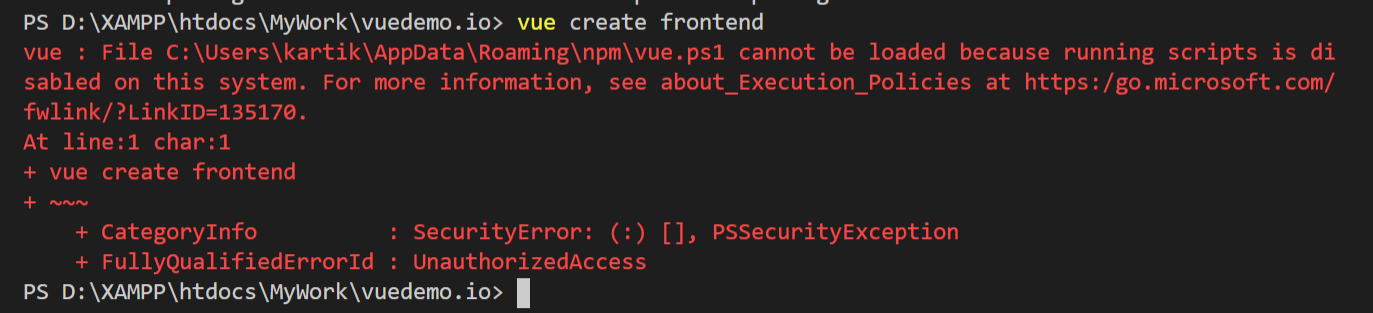
## vue ui

* + *Its Provides GUI for Create VueJs App*
  + *Its Provides Facility for Create or Import Existing VueJs App,*

*Installation of 3rd Party Node Modules Such Axios,SweetAlert,*

*Vue-router etc..*

Note - While installing VueJS App using CLI if You get UnauthorizedAccess error on CMD or Powershell



Can Resolved by Set ExecutionPolicy in Powershell as Admin Access

## Set-ExecutionPolicy RemoteSigned

* While installing Vuejs App there would be options such as
  + VueJs Version (2,3)
  + Add Router
  + Vuex, Elint, Options

Select option as per the requirements and Go Ahead..

## Run App

* + Navigating that Directory in cmd or powershell
  + and command npm run serve

## File Structure in VueJs App

* + node\_modules *(Modules or Libraries which are require for the App)*
  + public
    - img folder *(Should contains public images)*
    - index.html *(main file of App)*
    - favicon.ico
    - robots.txt
  + src
    - assets *(Should contains assets like js,css,fonts etc)*
    - components *(Vue Components (Reusable components))*
    - mixins *(if we creates, reusable functions and properties)*
    - router *(index.js, Declared Routes for Our App)*
    - store *(index.js)*
    - views *(Home.vue, Vue pages for our App contains components )*
* App.vue
* main.js

## Why VueJs ?

1. Virtual Dom Rendering (Fast)
2. Dynamic Data Binding
3. Reusable Components
4. Animation/ Transition
5. Many Css Framework Supports
6. Lightweight and Fast
7. Many Vuejs 3rd party libs available for better designing or interactivity of app
8. *Vuetify, Bootstrap Vue, Bootstrap Vue, Bootstrap Vue, UIV, VueStrap, Vue Material, Mint UI, Element UI, Vuecidity, Buefy, DeepReader, KeenUI, Quasar, AT UI, Vulma , Fish-UI*

**3) Component**

* + Every components or vue page contains
    - <template>
    - <script>
    - <style scoped>

## demo.vue

<template>

<div class="header"> Hello World </div>

</template>

<script>

export default {

name: 'Demo',

data() {

return{

name : '',

address : '',

hobby: []

}

}

}

</script>

<style scoped>

.header {

font-size: 28px;

margin: auto;

display: flex;

}

</style>

## Some errors occurred while running app listed below

* **Newline required** at end of file but not found

Solved by Just Adding newline character at the end of the file

* **Parsing Error** when incomplete html tag found

<script , <template, <style

complete incomplete tags

* **npm run serve** stopped or Freezed at nearly 40 %

any components should have errors in template

(little hard to identify)

* **no-multiple-empty-lines** Too many blank lines at the end of file. Max of 0 allowed

Resolved nu Deleting extra space or new lines from end of the file

**4) Routes**

Import components or view into route file (router/index.js)

Define route : -

{

path: '/about',

name: About,

component: Home

},

Note : Where Home is View inside views folder

**5) Binding**

* VueJs Provides various extended attributes

## Text Binding

<template>

<section>

<div> Hello {{name}} </div>

<div v-text="'My Name is '+name"> </div>

</section>

</template>

<script>

export default {

data() {

return{

name : 'Jemish',

}

}

}

</script>

## Looping

<template>

<section>

<div>

<li v-for="x in items" :key="x">

{{x}}

</li>

</div>

</section>

</template>

<script>

export default {

data() {

return{

items : ['Dosa','Manchurian','Paw Bhaji','Panipuri']

}

}

}

</script>

Note - Where Key is Mandatory to Bind in v-for

Note - Please Double Check that you’re giving Primitive Data Type to

key otherwise it will gives your bundle of errors at runtime

Best is give index as key

## Event Listener

<template>

<div>

<button @click="submit()">Submit</button>

</div>

</template>

<script>

export default {

methods: {

submit(){

console.log("Clicked")

}

}

}

</script>

Note - @keyup & @mouseover event listener worked same and @ is used for shorthand syntax we also can use v-on for binding like v-on:click

## Class Binding

<template>

<div :class="{green : success,red : !success}">

Hello

</div>

</template>

<script>

export default {

data() {

return {

success : false

}

}

}

</script>

<style scoped>

.green {

background : green;

}

.red {

background : red;

}

</style>

6) Add Bootstrap Jquery

* Installing bootstrap jquery etc via npm

npm i bootstrap jquery popper.js

* Importing installed libraries in main.js

import 'bootstrap/dist/css/bootstrap.min.css'

import 'jquery/src/jquery.js'

import 'bootstrap/dist/js/bootstrap.min.js'

var $ = require('jquery')

add this into component to use Jquery

7) Child Component

Send Data from Child Component to Parent Component

methods: {

myClick(par) {

this.$emit('clicked', par,this.btncal)

},

},

Parent Component

<CalcKeypad @clicked="setstring" />

setstring(value) {

console.log(value)

}

8) Axios Requests

There are 2 Ways to send post data from request

**npm install --save axios**

**mounted() {**

**let Json = {**

**name: "Jemish",**

**roll\_no: "37",**

**};**

**const params = new URLSearchParams();**

**params.append("data1", "jemish");**

**params.append("data2", "fdfdf");**

**var qs = require("qs");**

**// Method1**

**axios**

**.post(**

**"http://127.0.0.1/MyWork/VueApi/api/getTwitterUsers",**

**qs.stringify(Json)**

**)**

**.then(function(response) {**

**console.log(response.data);**

**});**

**// Method2**

**var formData = new FormData();**

**formData.append("avatar", this.file);**

**formData.append("data", JSON.stringify(this.add));**

**this.axios**

**.post(process.env.VUE\_APP\_API + "add-avenger", formData, {**

**headers: {**

**"Content-Type": "multipart/form-data",**

**},**

**})**

**.then((response) => {**

**})**

**.catch((e) => {**

**this.errors.push(e);**

**});**

9) Vue Lifecycle methods

**mounted(){**

**console.log("Before Mounted");**

**},**

**beforeCreate(){**

**console.log("Before Create")**

**},**

**created(){**

**console.log("Created");**

**},**

**beforeMount(){**

**console.log("Before Mounted");**

**}**

These methods called on Sequence like

*beforeCreate => created => beforeMount => mount*

**10) Other Attributes**

**v-once => when we want not changes in vue variables (Constant)**

**<span v-once>This will never change: {{ msg }}</span>**

**v-html=> for rendering html content from the variables**

**<p>Using v-html directive: <span v-html="rawHtml"></span></p>**

**<!-- full syntax -->**

**<a v-bind:href="url"> ... </a>**

**<!-- shorthand →**

**<a :href="url"> ... </a>**

**<!-- shorthand with dynamic argument -->**

**<a :[key]="url"> ... </a>**

**<!-- full syntax -->**

**<a v-on:click="doSomething"> ... </a>**

**<!-- shorthand -->**

**<a @click="doSomething"> ... </a>**

**<!-- shorthand with dynamic argument -->**

**<a @[event]="doSomething"> ... </a>**

11) Form Binding, v-model

When Code Consists <form> </form> tags

Submit button Requires preventDefault otherwise Page Will Reloading

**<template>**

**<div>**

**<form>**

**<input type="text" v-model="name" />**

**<button @click.stop.prevent="submit()"> Submit</button>**

**</form>**

**</div>**

**</template>**

**<script>**

**export default {**

**data() {**

**return {**

**name: "",**

**};**

**},**

**methods : {**

**submit(){**

**console.log(this.name)**

**}**

**}**

**};**

**</script>**

12) How to use SweetAlert2 in VueJS

Installing SweetAlert via npm

npm i vue-sweetalert2

Importing Plugin and Css in main.js

**import VueSweetalert2 from 'vue-sweetalert2';**

**import 'sweetalert2/dist/sweetalert2.min.css';**

**createApp(Index).use(router).use(VueSweetalert2).mount('#app')**

**this.$swal({**

**title: msg,**

**timer: 3000,**

**showCancelButton: false,**

**showConfirmButton: false,**

**icon: type**

**}).then(**

**function() {},**

**// handling the promise rejection**

**function(dismiss) {**

**if (dismiss === "timer") {**

**//console.log('I was closed by the timer')**

**}**

**}**

**);**

this.$swal is used for swal notify

**13) Mixins in VueJS**

Mixin can be used for use common functions and Properties in any Component or Vue by Just Importing and Registering it

<script>

export default {

name: "My-mixins",

methods: {

notify(msg, type) {

this.$swal({

title: msg,

timer: 3000,

showCancelButton: false,

showConfirmButton: false,

icon: type

}).then(

function() {},

// handling the promise rejection

function(dismiss) {

if (dismiss === "timer") {

//console.log('I was closed by the timer')

}

}

);

},

},

};

</script>

How to Import Mixins

import Mixins from '@/Mixins/My-mixins.vue'

Registering Mixins

mixins : [Mixins],

Use Mixins Function

this.notify();

**14**) MEVN App (MongoDb, Express, VueJS, NodeJs)

* Here I’m Using Express Js for Rest Apis, Mongoose Lib. (Object Relational Model) for Database Model in Express
* Setup Express MVC for Apis using my Documentation of Express JS
  + <https://docs.google.com/document/d/1JXiipJSoIX-HQaqlyhZgbie26rleSogAMsHgLZeXUdQ/edit?usp=sharing>
* Create vue.config.js file in your root directory of Vue JS App
* Setup proxy in vue.config.js to use same domain for Rest Apis also

module.exports = {

devServer: {

proxy: {

'/api': {

target: 'http://localhost:3000/',

ws: true,

changeOrigin: true,

secure: false

},

}

}

Here VueAppDomain.com/api will Redirects to localhost:3000/

* Configurations Edited so Restart Vue App and Enjoy Connectivity with Backend Express JS