Vue.Js Setup Part 4 :

**++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++**

**links :**[**https://www.youtube.com/watch?v=MQZ4PQGw-sc HYPERLINK "https://www.youtube.com/watch?v=MQZ4PQGw-sc&list=PL8p2I9GklV45qwTH-mdzllUuFRJO-euYn&index=28"& HYPERLINK "https://www.youtube.com/watch?v=MQZ4PQGw-sc&list=PL8p2I9GklV45qwTH-mdzllUuFRJO-euYn&index=28"list=PL8p2I9GklV45qwTH-mdzllUuFRJO-euYn HYPERLINK "https://www.youtube.com/watch?v=MQZ4PQGw-sc&list=PL8p2I9GklV45qwTH-mdzllUuFRJO-euYn&index=28"& HYPERLINK "https://www.youtube.com/watch?v=MQZ4PQGw-sc&list=PL8p2I9GklV45qwTH-mdzllUuFRJO-euYn&index=28"index=28**](https://www.youtube.com/watch?v=MQZ4PQGw-sc&list=PL8p2I9GklV45qwTH-mdzllUuFRJO-euYn&index=28)

**page not found | 404 page | Vue router part 4 >>>>>**

create one pageNotfound component and set url in main.js file if not matched url then open 404 page ..

**Main.Js >>>**

// Import Vue and App component

import { createApp } from 'vue';

import App from './App.vue';

// Import Bootstrap and BootstrapVue 3 CSS files

import 'bootstrap/dist/css/bootstrap.css';

import 'bootstrap-vue-3/dist/bootstrap-vue-3.css';

// Import BootstrapVue 3

import BootstrapVue3 from 'bootstrap-vue-3';

// Import Vue Router and necessary methods from vue-router

import { createRouter, createWebHistory } from 'vue-router';

// Import the components for routing

import Home from './components/Home.vue';

import About from './components/About.vue';

import Login from './components/Login.vue';

import User from './components/User.vue';

**import PageNotFound from './components/PageNotFound.vue';**

// Define the routes

const routes = [

{ path: '/', component: Home },

{ path: '/about', component: About },

{ path: '/login', component: Login },

{ path: '/user/:id', component: User },

// { path: '\*', component: PageNotFound }, // not used its use in vue 2 not this \* is not exits

**{ path: '/:pathMatch(.\*)\*', component: PageNotFound }, // Catch-all for 404 Page Not Found**

];

// Create the router instance

const router = createRouter({

history: createWebHistory(),

routes,

});

// Create the Vue app instance

const app = createApp(App);

// Use BootstrapVue 3 globally

app.use(BootstrapVue3);

// Use Vue Router globally

app.use(router);

// Define the global "filter-like" function (for example: capitalize)

app.config.globalProperties.$filters = {

capitalize(val) {

if (!val) return '';

return val.toString().toUpperCase();

},

};

// Mount the app to the #app element

app.mount('#app');

**PageNotFound.vue >>>>**

<template>

<div>

<h2>Page Not Found</h2>

<h3>Go to home page</h3>

<router-link to="/">Home</router-link>

</div>

</template>

<script>

export default {

name:"PageNotFound",

}

</script>

**now if you test with wrong or not exist url then send 404 page ..**

**++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++**

**dynamic component >>>>>**

**SignIn.vue >>>**

<template>

<div>

<h3>SignIn Component</h3>

</div>

</template>

<script>

export default {

name:"SingIn",

}

</script>

**SignUp.vue >>>**

<template>

<div>

<h3>SignUp Component</h3>

</div>

</template>

<script>

export default {

name:"SingUp",

}

</script>

**App.vue >>>>**

**while load dynamic component then i create 2 button when you click then component is loaded.**

<template>

<div id="app">

<h2>Dynamic Component</h2>

<!-- when i am use dynamic component then not use this other way in next lines -->

<!-- <signin />

<signup /> -->

**<component v-bind:is="cmp"></component>**

**<button @click="updatecmp('signin')">Load SignIn</button>**

**<button @click="updatecmp('signup')">Load SignUp</button>**

</div>

</template>

<script>

**import SignIn from './components/SignIn.vue';**

**import SignUp from './components/SignUp.vue';**

**export default {**

**name: 'App',**

**components: {**

**'signin':SignIn,**

**'signup':SignUp,**

**},**

**data(){**

**return {cmp:null}**

**},**

**methods:{**

**updatecmp(item){**

**this.cmp = item;**

**}**

**}**

**};**

</script>

<style>

#app {

font-family: Avenir, Helvetica, Arial, sans-serif;

-webkit-font-smoothing: antialiased;

-moz-osx-font-smoothing: grayscale;

text-align: center;

color: #2c3e50;

margin-top: 60px;

}

</style>

by default no component load when i click on load signin then load signin page & same singup.

default data function need to load & after update component then its works .

**+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++**

**make custom directive >>>>**

**App.vue >>>**

<template>

<div id="app">

<h2>Directive Component</h2>

<!-- Corrected component usage -->

<DirectivePage />

</div>

</template>

<script>

// Correct import for renamed component

import DirectivePage from './components/DirectivePage.vue';

export default {

name: 'App',

components: {

DirectivePage // Registered as DirectivePage

}

};

</script>

<style>

#app {

font-family: Avenir, Helvetica, Arial, sans-serif;

-webkit-font-smoothing: antialiased;

-moz-osx-font-smoothing: grayscale;

text-align: center;

color: #2c3e50;

margin-top: 60px;

}

</style>

**DirectivePage.vue >>>**

<template>

<div>

<h3>Custom Directive Example</h3>

**<!-- Add content here related to the directive -->**

**<h1 v-color="'red'">This text will be red!</h1>**

**<p v-color="'blue'">This text will be blue!</p>**

**<!-- focus directive input text -->**

**<input v-focus.log type="text" placeholder="This input will be focused" />**

</div>

</template>

<script>

export default {

name: 'DirectivePage' // Multi-word name

};

</script>

**Main.js >>>**

// Import Vue and App component

import { createApp } from 'vue';

import App from './App.vue';

// Import Bootstrap and BootstrapVue 3 CSS files

import 'bootstrap/dist/css/bootstrap.css';

import 'bootstrap-vue-3/dist/bootstrap-vue-3.css';

// Import BootstrapVue 3

import BootstrapVue3 from 'bootstrap-vue-3';

// Import Vue Router and necessary methods from vue-router

import { createRouter, createWebHistory } from 'vue-router';

// Import the components for routing

import Home from './components/Home.vue';

import About from './components/About.vue';

import Login from './components/Login.vue';

import User from './components/User.vue';

import PageNotFound from './components/PageNotFound.vue';

// Define the routes

const routes = [

{ path: '/', component: Home },

{ path: '/about', component: About },

{ path: '/login', component: Login },

{ path: '/user/:id', component: User },

// { path: '\*', component: PageNotFound }, // not used its use in vue 2 not this \* is not exits

{ path: '/:pathMatch(.\*)\*', component: PageNotFound }, // Catch-all for 404 Page Not Found

];

// Create the router instance

const router = createRouter({

history: createWebHistory(),

routes,

});

// Create the Vue app instance

const app = createApp(App);

// Use BootstrapVue 3 globally

app.use(BootstrapVue3);

// Use Vue Router globally

app.use(router);

**// Custom directive 'v-color'**

**app.directive('color', {**

**// This will be called when the directive is mounted on the element**

**mounted(el, binding) {**

**el.style.color = binding.value; // Set the element's color to the directive's value**

**}**

**});**

**// Custom directive 'focus'**

**app.directive('focus', {**

**mounted(el, binding) {**

**// Focus the element**

**if (binding.modifiers.log) {**

**console.log("Element is being focused!");**

**}**

**el.focus();**

**}**

**});**

// Define the global "filter-like" function (for example: capitalize)

app.config.globalProperties.$filters = {

capitalize(val) {

if (!val) return '';

return val.toString().toUpperCase();

},

};

// Mount the app to the #app element

app.mount('#app');

**I create custom directive function in main.js file and create component using v-color or v-focus like this is custom directive in vue .**

**+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++**

**Non-Props Data >>>>**

**App.vue >>>>**

<template>

<div id="app">

<h2>Non Props Data</h2>

<!-- Corrected component usage -->

<SignIn />

</div>

</template>

<script>

// Correct import for renamed component

import SignIn from './components/SignIn.vue';

export default {

name: 'App',

components: {

SignIn // Registered as SignIn

}

};

</script>

<style>

#app {

font-family: Avenir, Helvetica, Arial, sans-serif;

-webkit-font-smoothing: antialiased;

-moz-osx-font-smoothing: grayscale;

text-align: center;

color: #2c3e50;

margin-top: 60px;

}

</style>

**SignIn.vue >>>>**

<template>

<div>

<ChildProps data="Some Props data" id="non-props-data" />

**<!-- id="non-props-data" this is non props data because i am not define in child as props object like data or something so its is non props data ( but one thing if i use non props then child have one element tag inside all tags are present then its works )-->**

**<!-- data="Some Props data" this is props data because i am define in child as props object like data as String then its works as a props -->**

</div>

</template>

<script>

import ChildProps from './ChildProps.vue';

export default {

name:"SingIn",

components:{

ChildProps

}

}

</script>

**ChildProps.vue >>>>**

<template>

<!-- <div>

<p>John</p>

<p>Mohan</p>

<p>Vivek</p>

<p>{{ data }}</p>

</div> -->

<!-- if i remove div here then we can you attar form single tag value like wise -->

<p **v-bind="$attrs">**John</p> <!-- for access non props data then we can use -->

<!-- if access non props data if parent div not i have then we can use attributes -->

<p>Mohan</p>

<p>Vivek</p>

<p>{{ data }}</p>

</template>

<script>

export default {

name:"ChildProps",

**inheritAttrs:false, //** its use one call one time not use inside same id for non props data

**props:{**

**data:String // for access props data**

**}**

}

</script>

**props data use then child component pass**

**props:{**

**data:String // for access props data**

**}**

**non props data use then chid component pass**

**if parent div or tag present then no need to call any directive or function automatically called .**

**if not parent then you need to call single on single tag then we can use**

<p **v-bind="$attrs">**John</p>

this function ..

**+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++**

**27 Computed Property >>>>**

**Computed Property using like props but when use if calculation is tuf then we can create.**

**big method use then create computed property.**

**Ex .**

<div id="example">

**{{ message.split(' ').reserve().join(' ') }}**

</div>

**like wise i am create and use this .**

**computed property perfomance is fast , method is slow if you want to create logic then its works.**

**simple logic then use method & property , if logic has complex then use computed property because its perfomance is to fast rather than method & property .**

**App.vue >>>>**

<template>

<div id="app">

<h2>Computed Property</h2>

<!-- Corrected component usage -->

**<ComputedComponent />**

</div>

</template>

<script>

// Correct import for renamed component

**import ComputedComponent from './components/computedComponent.vue';**

export default {

name: 'App',

components: {

**ComputedComponent // Registered** as ComputedComponent

}

};

</script>

<style>

#app {

font-family: Avenir, Helvetica, Arial, sans-serif;

-webkit-font-smoothing: antialiased;

-moz-osx-font-smoothing: grayscale;

text-align: center;

color: #2c3e50;

margin-top: 60px;

}

</style>

**ComputedComponent .vue >>>**

<template>

<div>

<h5>Computed Component</h5>

<!-- Using direct logic in the template (this can affect performance for complex calculations) -->

<p>Logic is Dollar to rupee conversation less discount : {{ (Dollar \* rupeeVal) - discount }}</p>

<!-- Using computed property for better performance -->

<p> Computed Property Result not need to small Brackets: {{ getResult }}</p>

<!-- Using method property but computed property is better performance to method -->

<p> Method Property Result need to small Brackets: {{ getResultMethod() }}</p>

</div>

</template>

<script>

export default {

name: "computedComponent",

data() {

return {

Dollar: 100,

rupeeVal: 83.90,

discount: 30,

};

},

methods: {

// method property to calculate the result

getResultMethod() {

return (this.Dollar \* this.rupeeVal) - this.discount;

}

},

computed: {

// Computed property to calculate the result

getResult() {

return (this.Dollar \* this.rupeeVal) - this.discount;

}

}

}

</script>

computed property have better performance to method .

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

**Watchers with example >>>>**

**all property observed via watchers .**

**both function name should be same in property inside function & watchers inside function.**

**if you want to observe the data or value inside fucntion then i can use watch property .**

**Ex >>>>**

**App.vue >>>>**

<template>

<div id="app">

<h2>Watchers Components</h2>

<!-- Corrected component usage -->

<WatchersComponents />

</div>

</template>

<script>

// Correct import for renamed component

import WatchersComponents from './components/watchersComponents.vue';

export default {

name: 'App',

components: {

WatchersComponents // Registered as ComputedComponent

}

};

</script>

<style>

#app {

font-family: Avenir, Helvetica, Arial, sans-serif;

-webkit-font-smoothing: antialiased;

-moz-osx-font-smoothing: grayscale;

text-align: center;

color: #2c3e50;

margin-top: 60px;

}

</style>

**watcherComponents.vue >>>>**

<template>

<div>

<h2>Watchers</h2>

<h4>{{ count }}</h4>

<button @click="count=count+1">+</button>

<button @click="count=count-1">-</button>

</div>

</template>

<script>

export default {

name:"watchersComponents",

data(){

return {

count:0

}

},

watch:{

count(val,prev){ // have 2 value first value & old value

if(val>5 && val>prev){

// when this value goes up to 5 then its reset value 0

this.count=0;

alert("stop counting");

}

}

}

}

</script>

watch is observe value like higher 5 then its reset count value is 0.

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

**Slots with example >>>>**

Slots component uses for the all type of data pass then we can use this .

you can pass multiple data type using slot tag and send to parent ton child.

**App.vue >>>>**

<template>

<div id="app">

<h2>Slots Components Property in Vue Js</h2>

<!-- Corrected component usage -->

<SlotsComponent />

</div>

</template>

<script>

// Correct import for renamed component

import SlotsComponent from './components/SlotsComponent.vue';

export default {

name: 'App',

components: {

SlotsComponent // Registered as ComputedComponent

}

};

</script>

<style>

#app {

font-family: Avenir, Helvetica, Arial, sans-serif;

-webkit-font-smoothing: antialiased;

-moz-osx-font-smoothing: grayscale;

text-align: center;

color: #2c3e50;

margin-top: 60px;

}

</style>

**SlotsComponents.vue >>>>>**

<template>

<div>

<h3>Slots Components</h3>

**<!-- default slot pass -->**

**<SlotsChild></SlotsChild>**

**<SlotsChild><h4>Pass H4 tag in slot value using slot tag</h4></SlotsChild>**

**<!-- href pass slot pass -->**

**<SlotsChild><a href="#google">Google</a></SlotsChild>**

**<!-- image pass slot pass -->**

**<SlotsChild><img src="https://via.placeholder.com/600/92c952" /></SlotsChild>**

**<!-- el ( html code ) pass slot pass -->**

**<SlotsChild><p>Pass paragraph</p></SlotsChild>**

</div>

</template>

<script>

import SlotsChild from './SlotsChild.vue';

export default {

name:"SlotsComponents",

components:{

SlotsChild,

}

}

</script>

**SlotsChild.vue >>>>**

<template>

<div>

<!-- how to pass default value in slots -->

**<slot>Default Slots</slot>**

<!-- how to pass all value pass from data components using child components -->

**<slot></slot>**

</div>

</template>

<script>

export default {

name:"SlotsChild",

}

</script>

**You can use multiple data type you pass in child using slot tag...**

**++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++**

**Multiple slot with Name Slots >>>>>**

**SlotsComponents.vue >>>>**

<template>

<div class="card\_1">

**<SlotsChild>**

**<template v-slot:header>**

**<h4>Saurabh Shukla</h4>**

**</template>**

**<template v-slot:main>**

**<p>**

**This is main content for you !!**

**</p>**

**</template>**

**<template v-slot:footer>**

**<h6>**

**@copyright for 2024**

**</h6>**

**</template>**

**</SlotsChild>**

</div>

<div class="card\_2">

**<SlotsChild>**

**<template v-slot:header>**

**<h4>**

**Vijay Shukla**

**</h4>**

**</template>**

**<template v-slot:main>**

**<p>**

**This is main content for vijay !!**

**</p>**

**</template>**

**<template v-slot:footer>**

**<h6>**

**@copyright for 2024**

**</h6>**

**</template>**

**</SlotsChild>**

</div>

</template>

<script>

import SlotsChild from './SlotsChild.vue';

export default {

name: "SlotsComponents",

components: {

SlotsChild,

}

}

</script>

<style scoped>

.card\_1{

padding: 10px;

background: blue;

color: white;

width: 300px;

height: 150px;

align-content: center;

}

.card\_2{

padding: 10px;

background: skyblue;

color: blue;

width: 300px;

height: 150px;

align-content: center;

}

</style>

**SlotsChild.vue >>>>**

<template>

<div>

**<header>**

**<slot name="header"></slot>**

**</header>**

**<content>**

**<slot name="main"></slot>**

**</content>**

**<footer>**

**<slot name="footer"></slot>**

**</footer>**

</div>

</template>

<script>

export default {

name:"SlotsChild",

}

</script>

In this type i can use slot inside child based on her name , because multiple slot access via name inside slot point.

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++