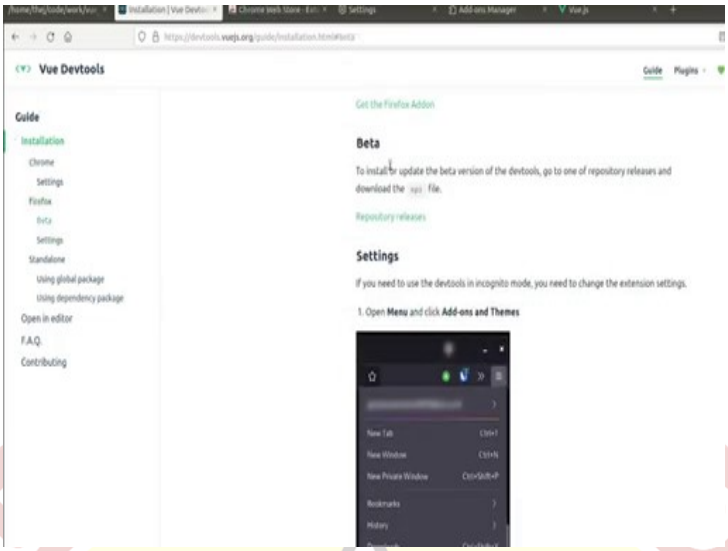
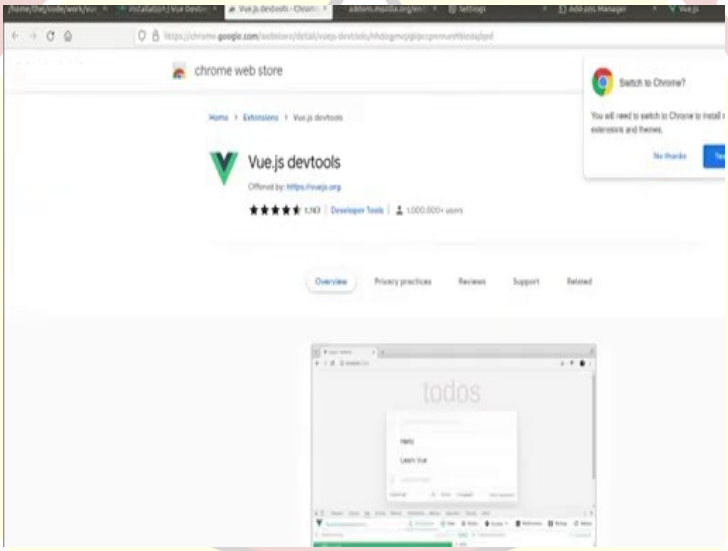
IIT Madras

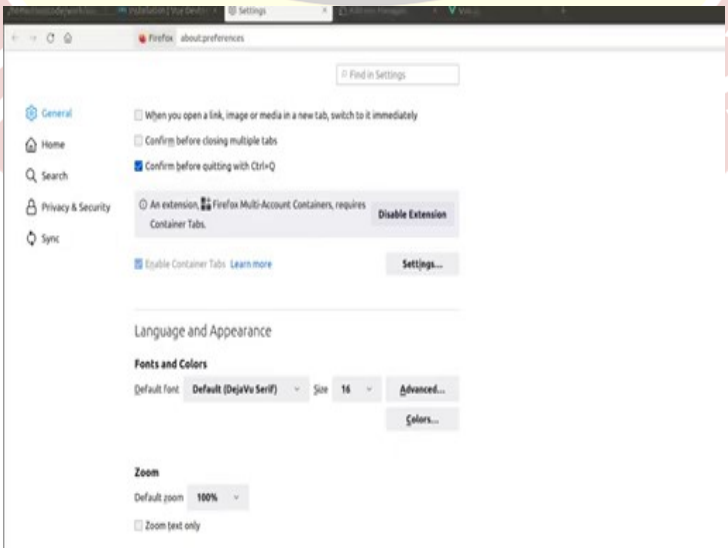
ONLINE DEGREE

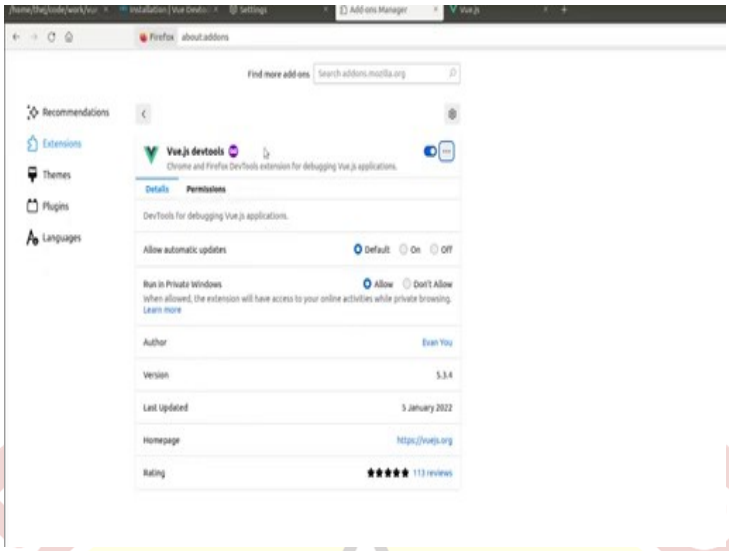
Modern Application Development - 2
Professor Thejesh G N
Software Consultant
Bachelor of Science Degree
Indian Institute of Technology, Madras
Introduction to VueJS

(Refer Slide Time: 0:15)


Vue Devtools
Guide
Plugins
Guide
Installation
Chrome
Settings
Firefox
Beta
Settings
Standalone
Using global package
Using dependency package
Open in editor
F.A.Q.
Contributing
Get the Firefox Addon
Beta
To install or update the beta version of the devtools, go to one of repository releases and download the .xpi file.
Repository releases
Settings
If you need to use the devtools in incognito mode, you need to change the extension settings.
1. Open Menu and click Add-ons and Themes
New Tab
New Window
New Private Window
Bookmarks
History


chrome web store
Switch to Chrome?
Vue.js devtools
Offered by: https://vuejs.org
Developer Tools
Overview
Privacy practices
Reviews
Support
Related
todos


Settings
Firefox about:preferences
Find in Settings
General
Home
Search
Privacy & Security
Sync
When you open a link, image or media in a new tab, switch to it immediately
Confirm before closing multiple tabs
Confirm before quitting with Ctrl+Q
An extension, Firefox Multi-Account Containers, requires Container Tabs.
Disable Extension
Enable Container Tabs Learn more
Settings...
Language and Appearance
Fonts and Colors
Default font
Default (DejaVu Serif)
Size
16
Advanced...
Colors...
Zoom
Default zoom
100%
Zoom text only

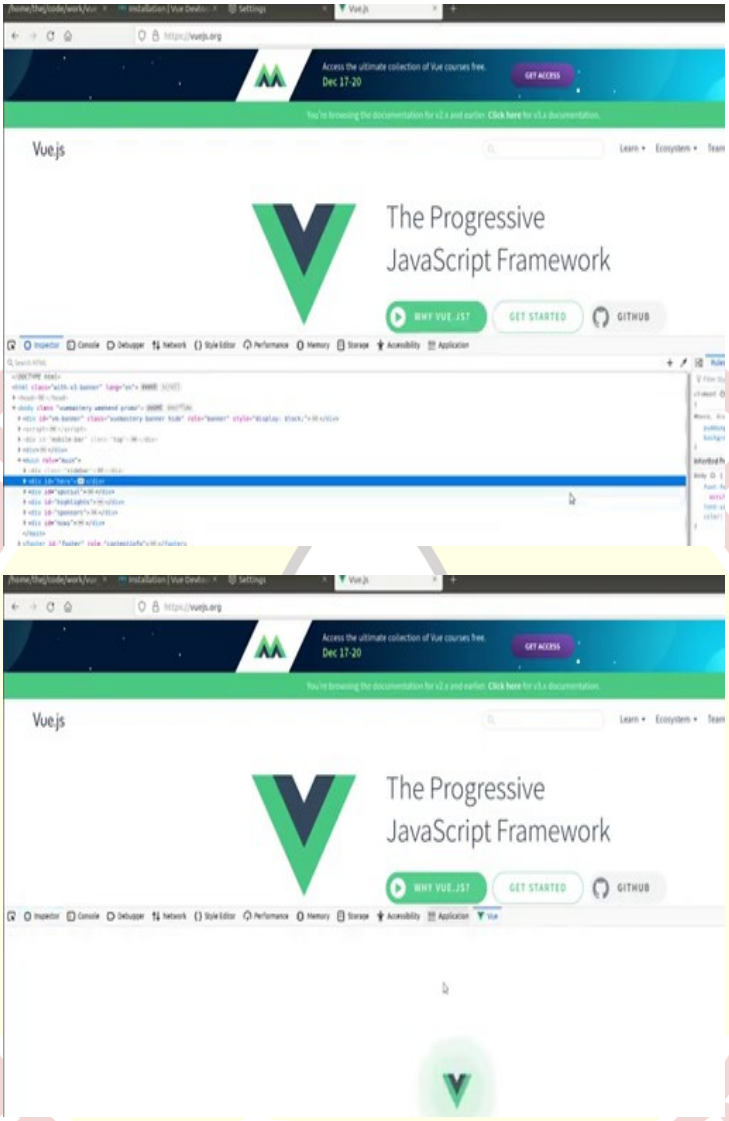


Hi! Welcome to the modern application development to screencast. In this screencast we will learn how to set up and run 'Hello World' Vue application. It is an introduction of screencast, we will learn the basics of Vue. For this you will need an editor, like VSCodium or Sublime text or Gedit, I am going to use Sublime text here, you can use whichever you like; A browser, or either Firefox or Chrome and Vue Devtools.

I will show you Vue Devtools is, and we need to install and enable it. Vue is a JavaScript framework that we are going to use. Before we begin to use we need to install Vue Devtools that we are going to use it as in when we do some debugging. It is a Chrome add-on or Firefox add-on. You can go to Devtools.vuejs.org and install it. You can click on this Google Chrome extension or if you are using Firefox you can click on Firefox extension.

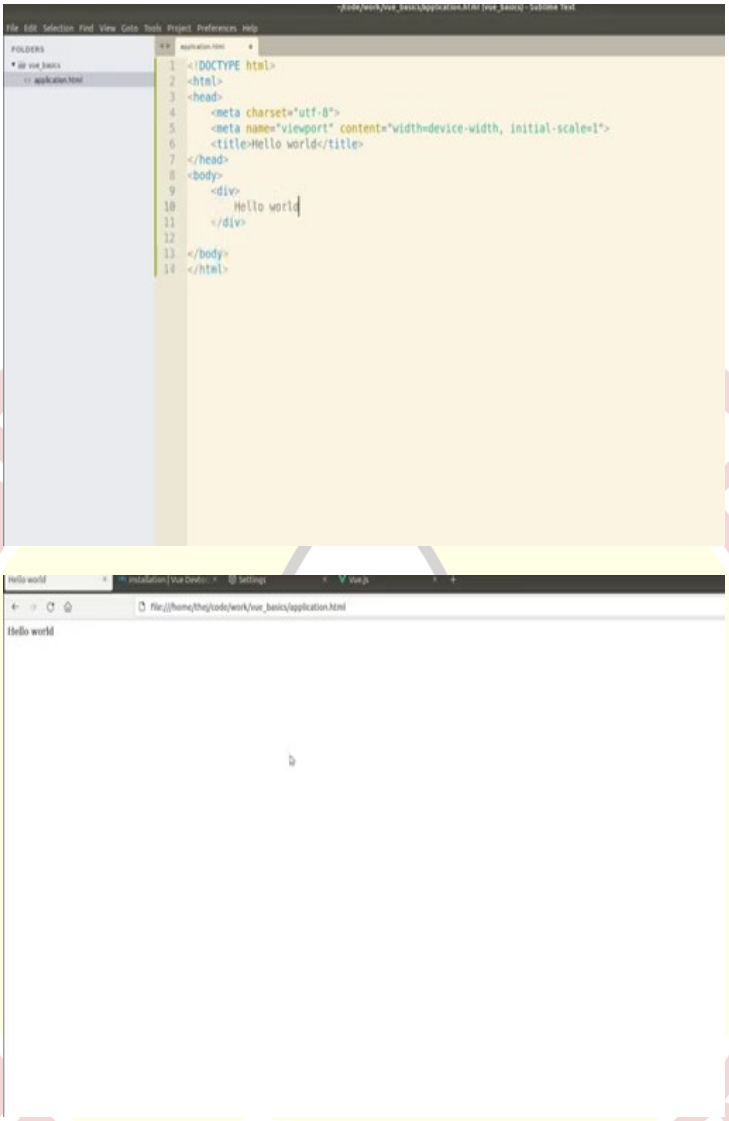
And click and install them, I am not going to install it here, since I am running Firefox, I could have installed here, but I have already installed it, so if you go to your settings, extensions and themes, you can see your Vue Devtools is installed, since I am using in private window, I am also run in private window. So, now Vue Ddevtools is installed.

(Refer Slide Time: 1:59)



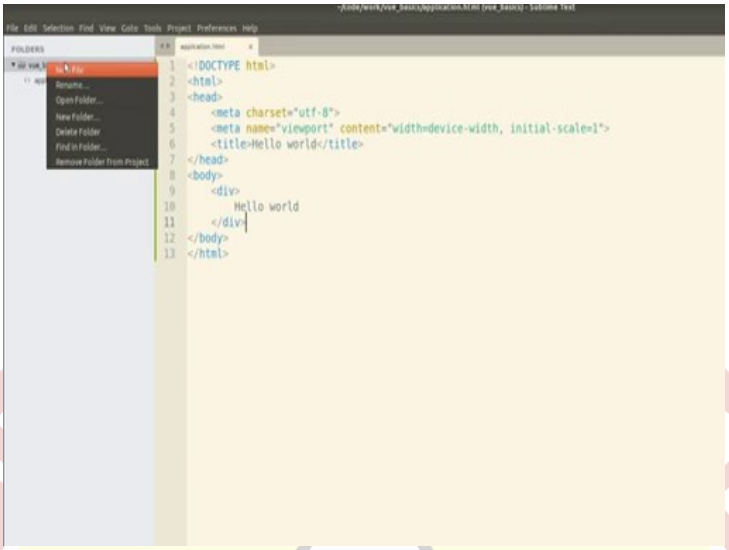
How do you check it is enabled? Just right click on any html and go to inspect; you should be able to see Vue tab in the developer tools, here, like this. So, this is the one that gets enabled after installing that add-on and this will be helpful for us to debug, and we will use it throughout. Now that we are done with the setup we will go back to our application.

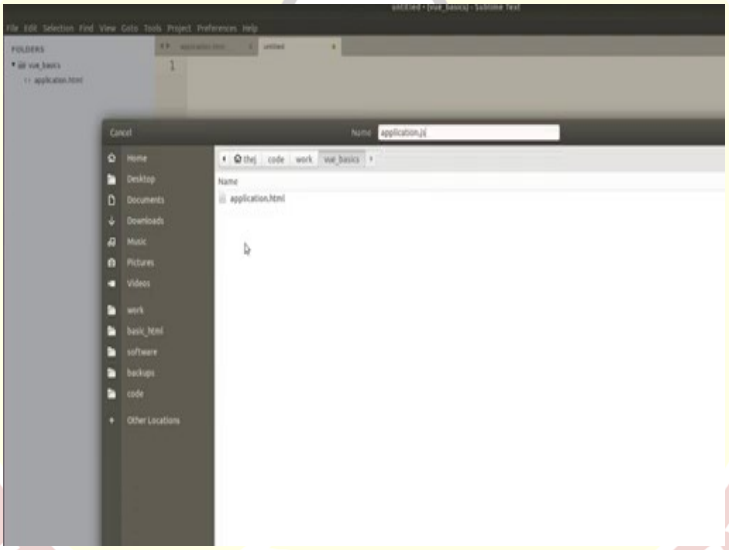
(Refer Slide Time: 2:34)

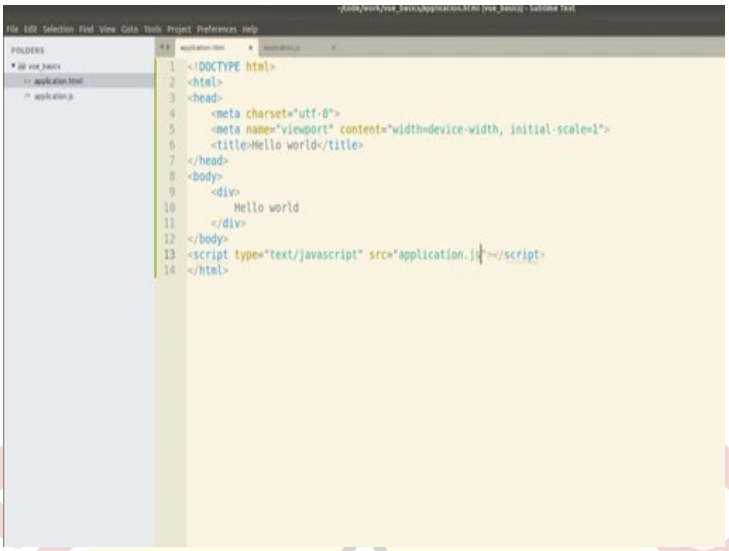


We are going to create a very-very simple application. So, I have a pre-created an html called application.html which we are going to use, let us say create an html, simple html, this is our simple html, I will just call it 'Hello world' and here let us add a div and add 'Hello world'. This is the simplest app that we could do. I am opening it locally, so you can just refresh and see my 'Hello world' here, that is the message on to display.

(Refer Slide Time: 3:14)

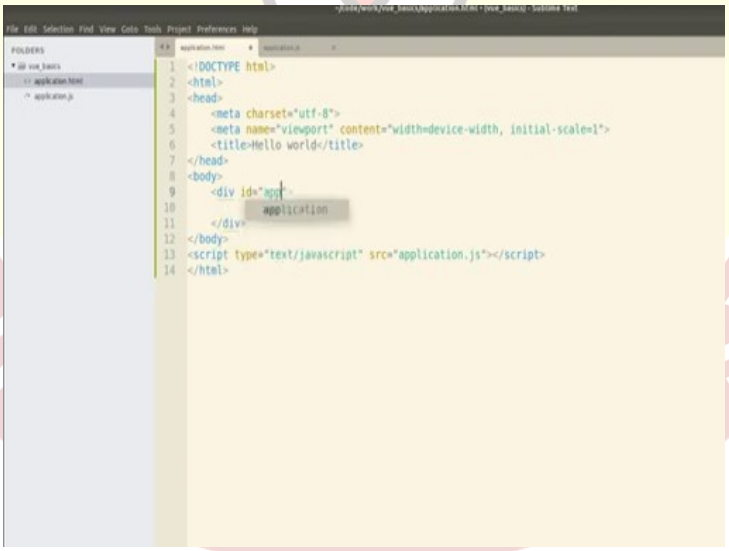


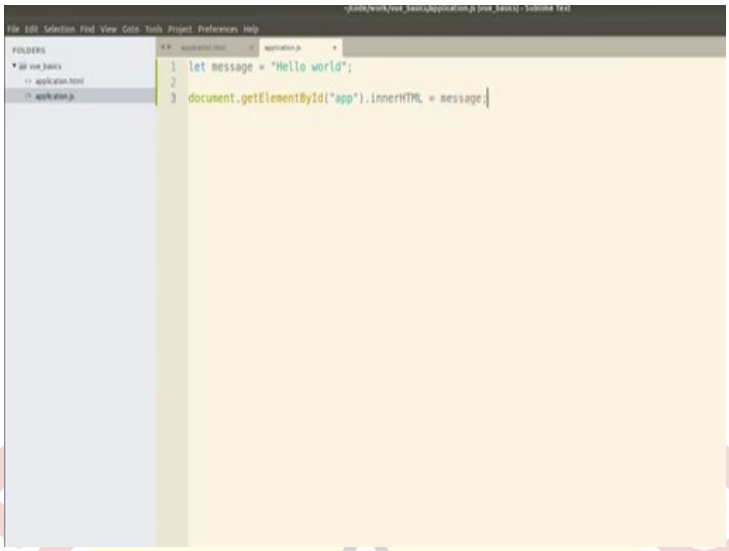




Now, I want to do some more improvement, I will just add another file called application.js where we will write all our application, so and then I am going to include that at the bottom once the whole page is loaded, I am going to include that, and src is equal to application.js, this is just simple inclusion of our JavaScript file inside our html.

(Refer Slide Time: 3:53)

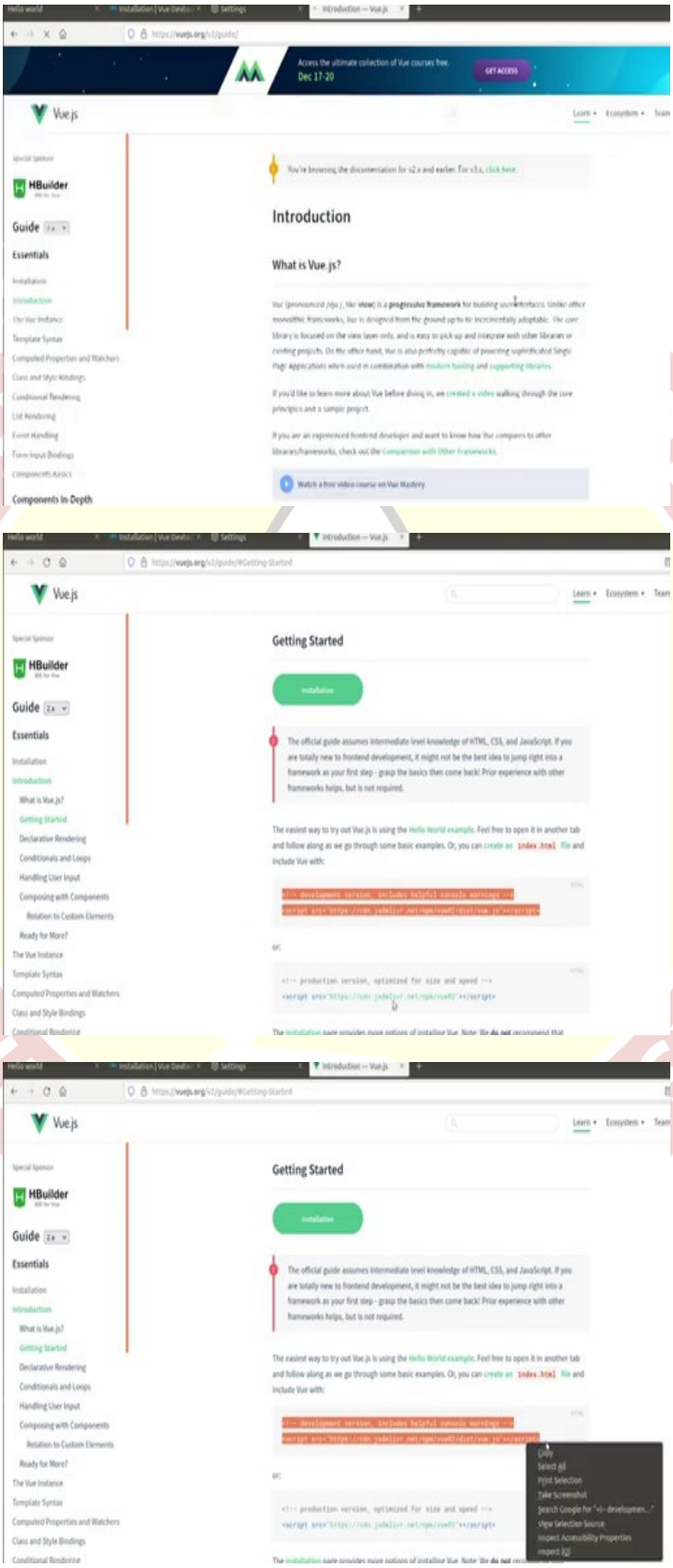


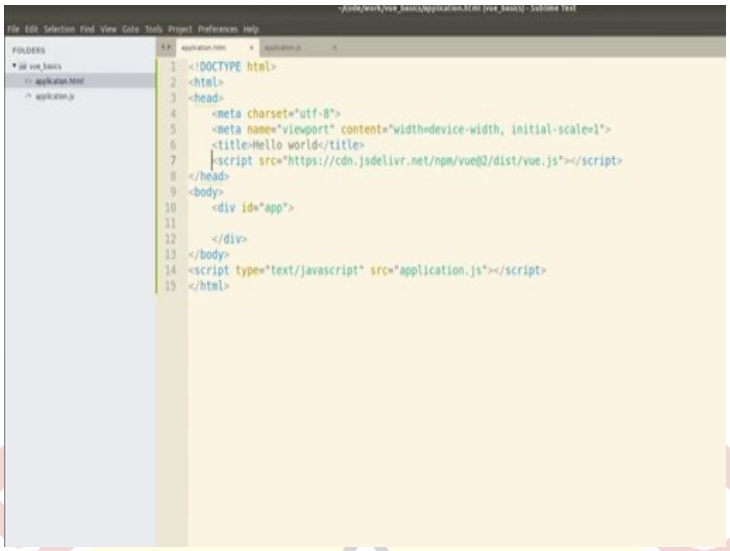


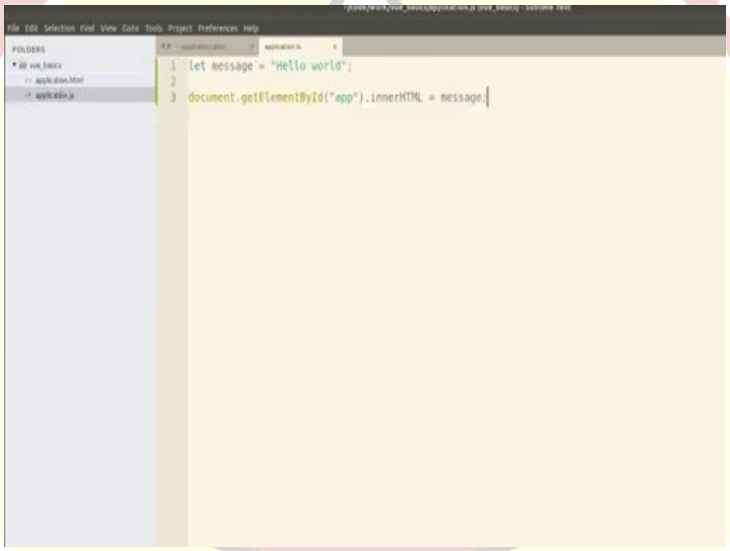
Now, let us say if we did not want to say this here, we wanted to say this here, let us say message was here. And we wanted to display here, there were two options, for example, on loading you could get the reference to the div and we can set in our html. Let us say div had an id called ‘app’ then we could do our document.get element by id, id is ‘app’ dot inner html is equal to message. I think this should work, we are setting it.

Now there are better ways to do it, this is very rudimentary JavaScript way of doing it and it is very limited to, I mean, here we are just showing a simple message and hence it is okay, it works, but if you are going to do a complete application within this, our app div, suddenly this becomes unmanageable, that is where we are going to use Vue.js. Vue.js has a way to do this, so let us start with the including Vue.js.

(Refer Slide Time: 5:23)



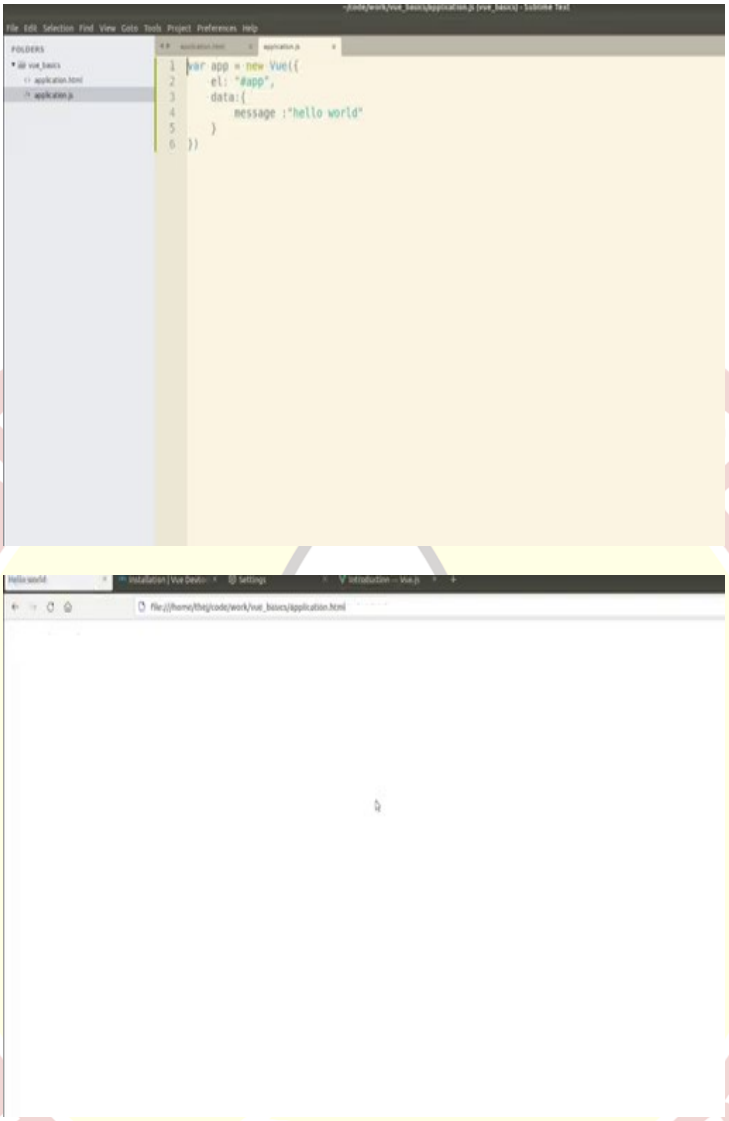


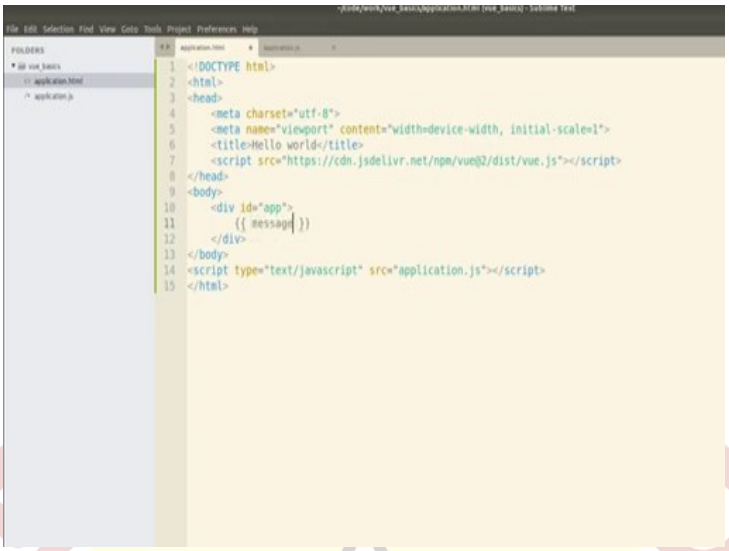


So, we are going to use Vue.js 2, so let us get started. There are two variations of Vue.js, one is Vue.js 2 and Vue.js 3 and Vue.js 2 is the most used currently, so we are going to use it, I mean, with little changes you could use Vue.js 3 as well, but in this course we are going to use Vue.js 2. We are going to click on, get started and here there is a script, which is a development version, it just has more logs and console logs, so you can debug easier.

So we, I am going to include this, if you are going to do production, then you can use like a minimized optimized version here, but I am going to use this one, I am just going to copy this, I am just going to include that here, just going to remove this comment. Now we have done this, I am going to remove this. We have done this.

(Refer Slide Time: 6:26)

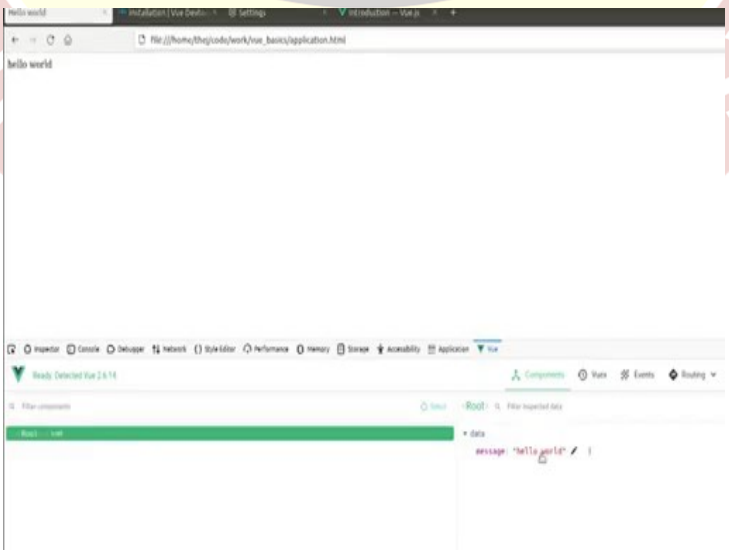



File Edit Selection Find View Goto Tools Project Preferences Help
FOLDERS
vue_basics
application.html
application.js
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<title>Hello world</title>
<script src="https://cdn.jsdelivr.net/npm/vue@2/dist/vue.js"></script>
</head>
<body>

{{ message }}

</body>
<script type="text/javascript" src="application.js"></script>
</html>


Hello world
Installation | Vue Devto
Settings
Introduction — Vue.js
file:///home/theji/code/work/vue_basics/application.html
hello world


Hello world
Installation | Vue Devto
Settings
Introduction — Vue.js
file:///home/theji/code/work/vue_basics/application.html
hello world
Inspector
Console
Debugger
Network
Style Editor
Performance
Memory
Storage
Accessibility
Application
Vue
Components
Vuex
Events
Routing
Root
data
message: "hello world"

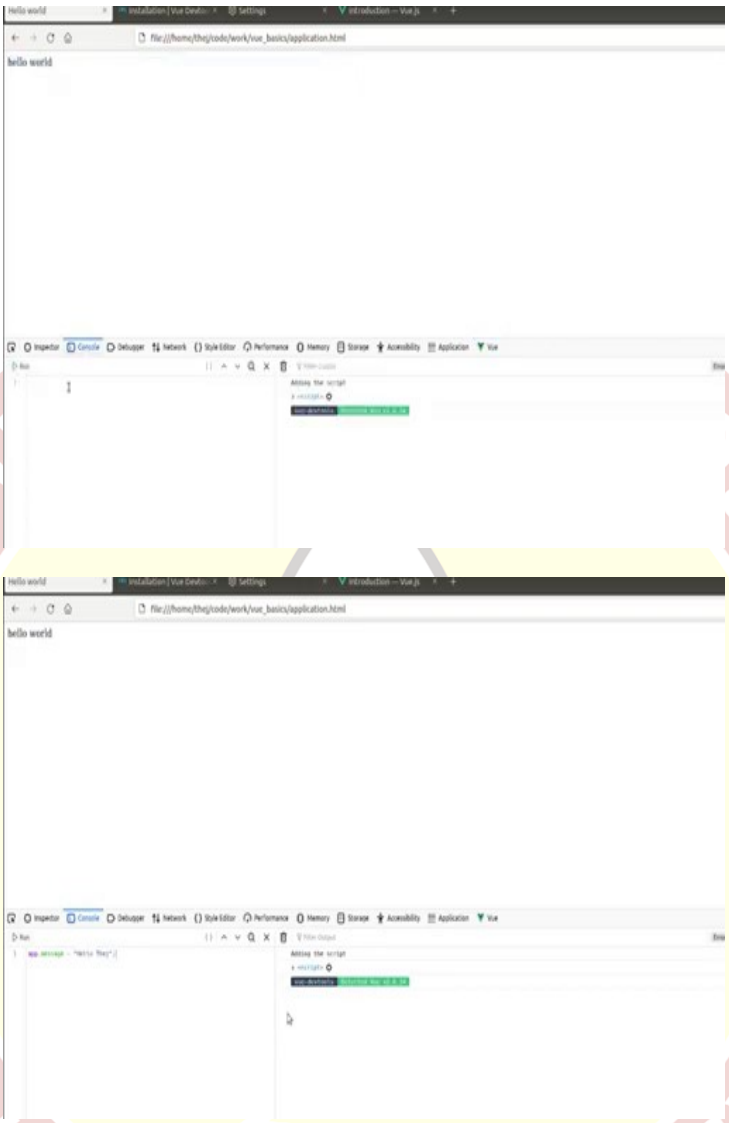
Now, we want to use Vue to set up the message here, let us do that. That is done by adding a Vue app, which is the instance of a Vue. Let us do that first, let us create an app called ‘app’ and will instantiate a new Vue class. Now Vue takes parameters and the two most important ones are called element, which is denoted by el which takes the id of our app for which we want to refer, id of our app.

That is hash app and then the next one takes data, which is the data that we are going to use in the app. And we want to use message here and our message wants to be ‘Hello world’. I am going to remove that here as a single thing, we are going to use it as a data here. Now, if I load, still not showing up, that is because even though we have associated el or element to our application html’s div app here, we are still not somehow told this html that you need to use this attribute from the Vue to show the message.

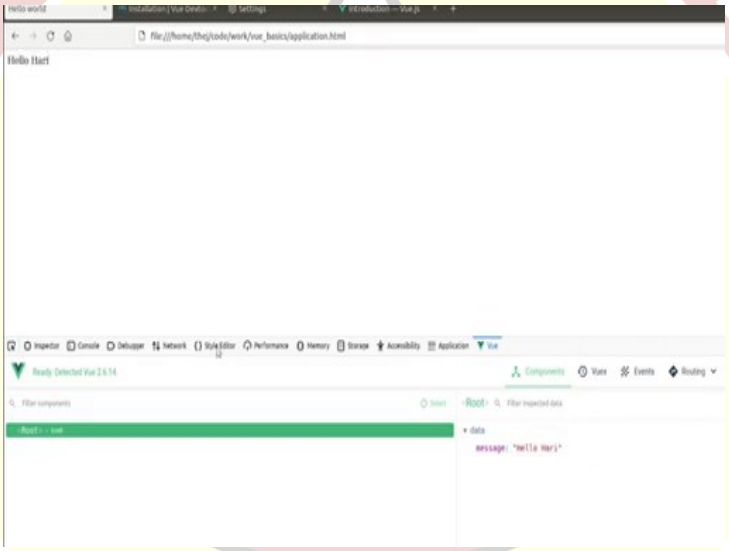
That is done by adding an expression. This is similar to a templates, you can just add message here and then, what now we are telling is this id app is attached to this Vue using this element hash app and then the message or expression comes from the data of the Vue using this message attribute inside the data. Now I am going to save this and run it and let us see what happens. Voila! You can see ‘Hello world’ here.

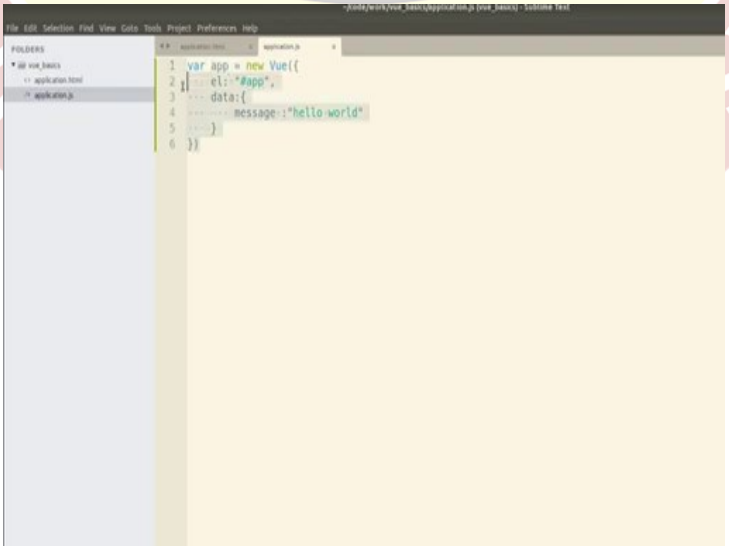
Now the most interesting part of the Vue is not this, the most interesting part of the Vue is when you actually do inspect and you go to the Vue, it shows your root element, root element is this Vue object which is what is associated with our div, this is our root. This is an instance of the Vue object, Vue class, this is the root. Now you can see here root and it has data as you can see and it has message ‘Hello world’.

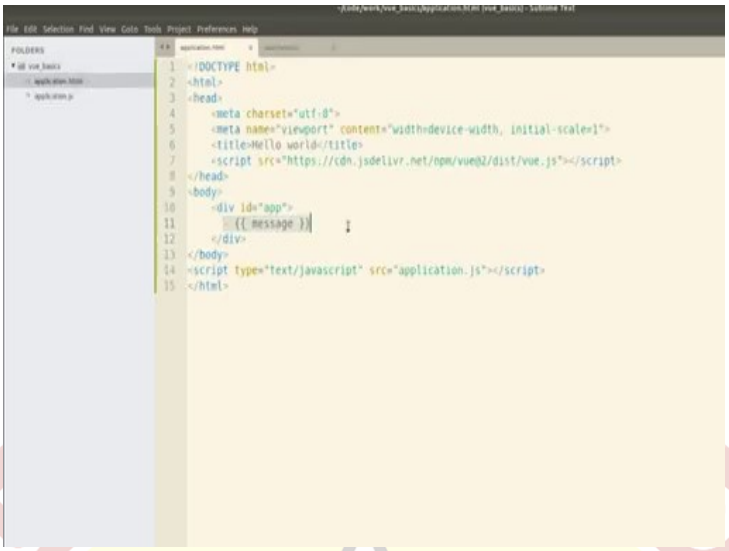
(Refer Slide Time: 9:21)



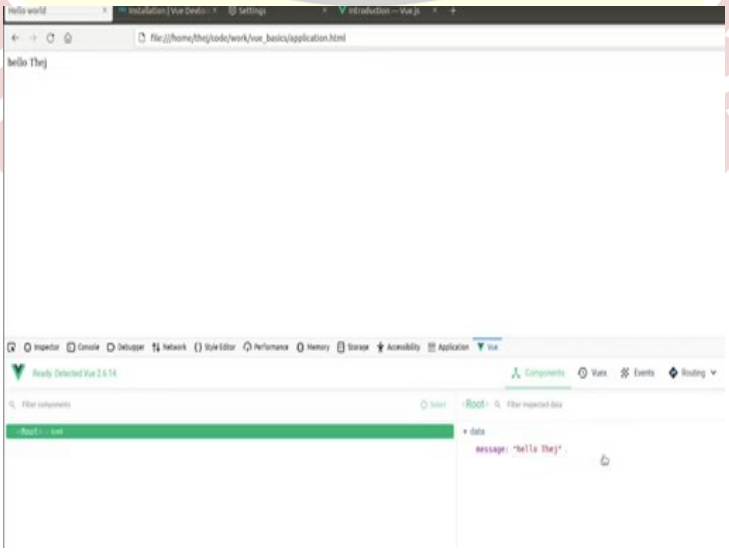

Hello world
Introduction — Vue.js
file:///home/thej/code/work/vue_basics/application.html
Hello Thej
Inspector
Console
Debugger
Network
Style Editor
Performance
Memory
Storage
Accessibility
Application
Vue
app.message = "Hello Thej";
Adding the script
vue-devtools
app.message = "Hello Thej"
"Hello Thej"


Hello world
Introduction — Vue.js
file:///home/thej/code/work/vue_basics/application.html
Hello Hari
Inspector
Console
Debugger
Network
Style Editor
Performance
Memory
Storage
Accessibility
Application
Vue
Ready. Detected Vue 2.6.14
Components
Vuex
Events
Routing
Filter components
Select
Root
Filter inspected data
<Root>
data
message: "Hello Hari"


~/code/work/vue_basics/application.js (vue_basics) - Sublime Text
File Edit Selection Find View Goto Tools Project Preferences Help
FOLDERS
vue_basics
application.html
application.js
application.html
application.js
var app = new Vue({
el:"#app",
data:{
message:"hello world"
}
})


File Edit Selection Find View Goto Tools Project Preferences Help
FOLDERS
vue_basics
application.html
application.js
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <meta charset="utf-8">
5 <meta name="viewport" content="width=device-width, initial-scale=1">
6 <title>Hello world</title>
7 <script src="https://cdn.jsdelivr.net/npm/vue@2/dist/vue.js"></script>
8 </head>
9 <body>
10
11 {{ message }}
12
13 </body>
14 <script type="text/javascript" src="application.js"></script>
15 </html>


Hello world
file:///home/theji/code/work/vue_basics/application.html
hello world
Inspector
Console
Debugger
Network
Style Editor
Performance
Memory
Storage
Accessibility
Application
Vue
Ready. Detected Vue 2.6.14
Components
Vuex
Events
Routing
Root

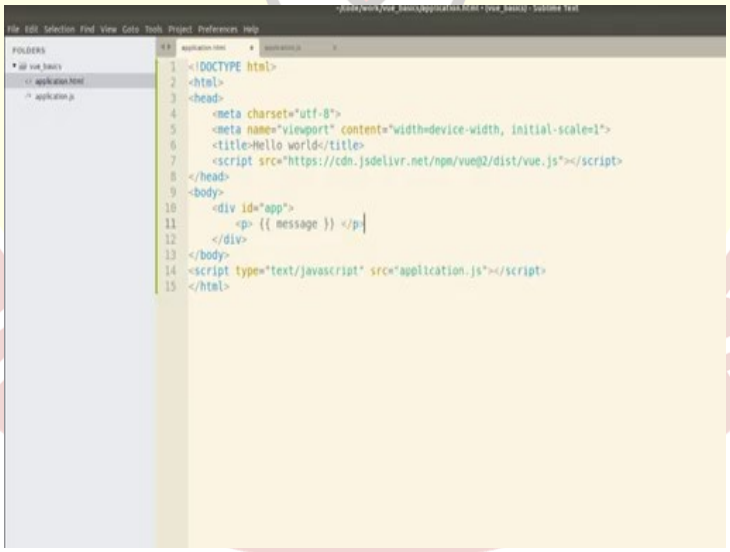

Hello world
file:///home/theji/code/work/vue_basics/application.html
hello Thej
Inspector
Console
Debugger
Network
Style Editor
Performance
Memory
Storage
Accessibility
Application
Vue
Ready. Detected Vue 2.6.14
Components
Vuex
Events
Routing
Root
data
message: "hello Thej"

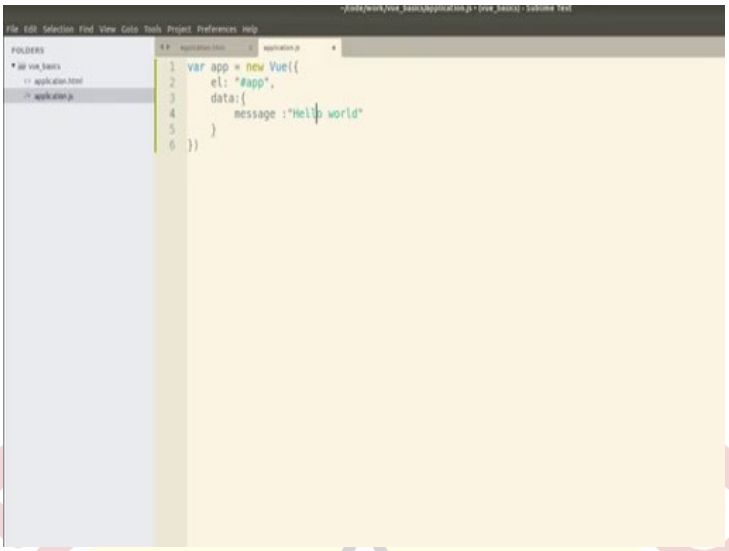
Now let me show you something very interesting, I can go to console, I can see we have an app object, app.message is equal to Hello Thej. Now let me run this JavaScript, now you can see quickly the value of this changed and you can also see the value here change, if you go to Vue you can also… This value here changed or you can click here, I mean, this is the advantage of having a Vue add-on that will give you access to Vue related things including component, attributes, events, etc., direct access.

Now here you can do Hari and if I save it changes. Now this is what we call reactivity. Here the Vue is being reactive, it looks for the values in this and as soon as this value of the message changes here it goes, it reacts to that and goes and updates this. You can see that instantly happening. Now, if I refresh it, it goes back to the original status of the message which is this, but as soon as I update it click on root, click on this, click on Thej.

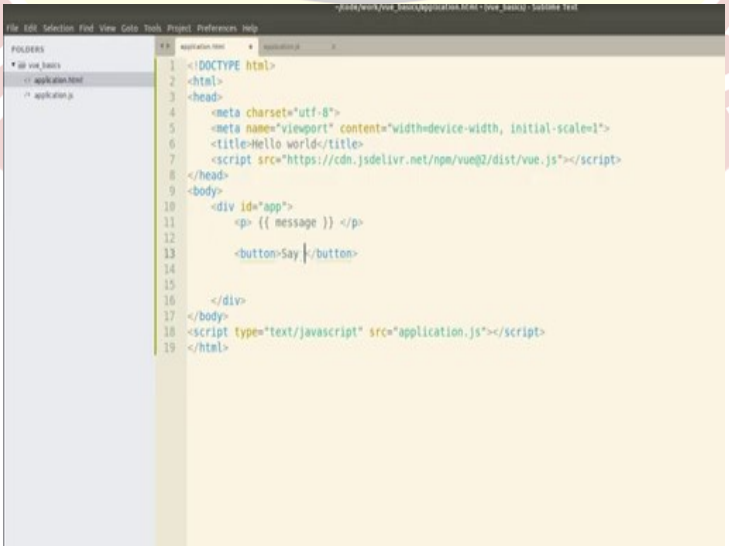
I just have to update the value of the status or data and the UI gets reflected automatically, I do not have to do any changes to the UI. This is the most important part of using Vue ,which is being reactive.

(Refer Slide Time: 11:21)




File Edit Selection Find View Goto Tools Project Preferences Help
FOLDERS
vue_basics
application.html
application.js
var app = new Vue({
el: "#app",
data:{
message :"Hello world"
}
})


file:///home/thej/code/work/vue_basics/application.html
hello world
Inspector
Console
Debugger
Network
Style Editor
Performance
Memory
Storage
Accessibility
Application
Vue
Ready. Detected Vue 2.6.14
Components
Vuex
Events
Routing
Root
Select a component instance to


File Edit Selection Find View Goto Tools Project Preferences Help
FOLDERS
vue_basics
application.html
application.js
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<title>Hello world</title>
<script src="https://cdn.jsdelivr.net/npm/vue@2/dist/vue.js"></script>
</head>
<body>

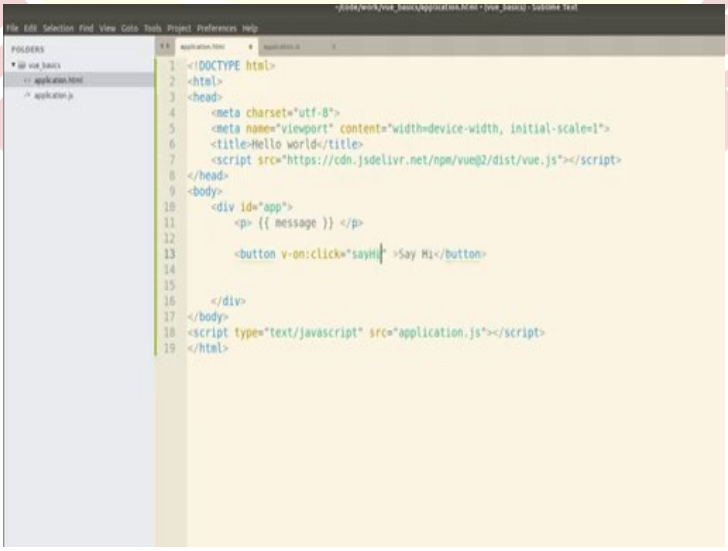
 {{ message }}
<button>Say</button>

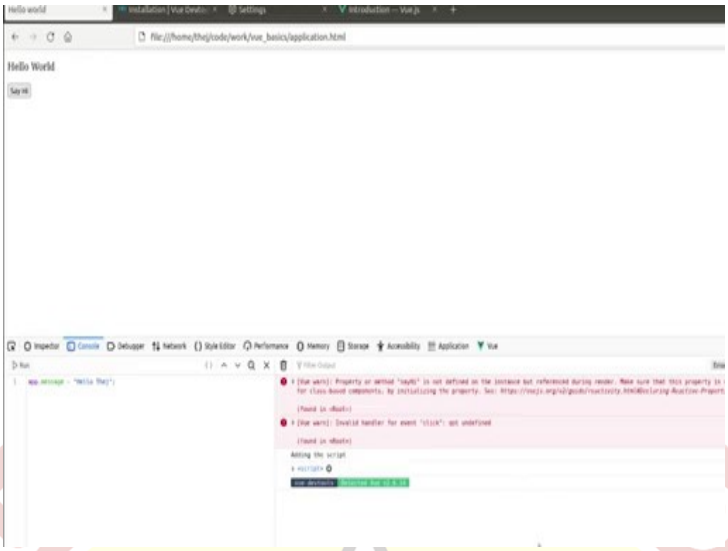
</body>
<script type="text/javascript" src="application.js"></script>
</html>

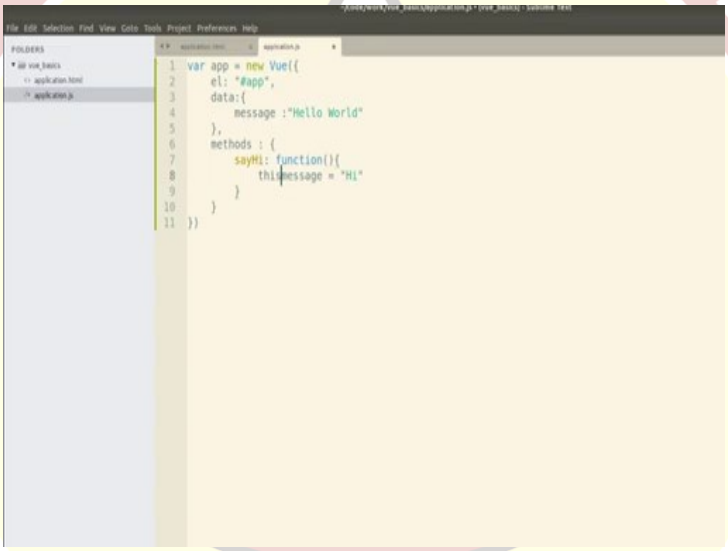
Now let us do some more improvements to it. Let us say when we do, when someone comes here and looks at Hello Thej or some other message Hello World, they want to say ‘hello’ back, so let us add some button for them to say ‘hi’ to us. It is a simple thing. We will just add go back here, let us put this inside a p, it should not matter and let us slash p, it will still work, so you can still works.

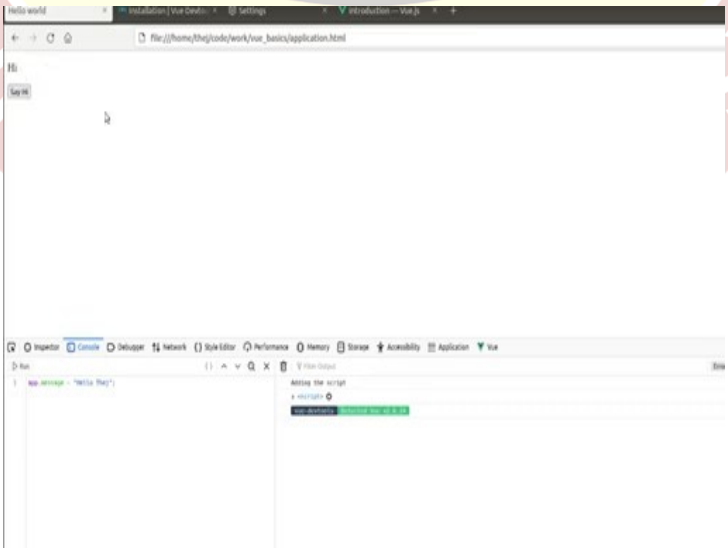
I am just going to make this as capital letters ‘Hello World’ to make it cleaner. And now I am going to add, within our app I am going to add a button, button is our standard button. Now let us say ‘hi’, so if I go back here and refresh, you can see there is a button here and you are clicking, ‘Say hi’ and it is not doing anything.

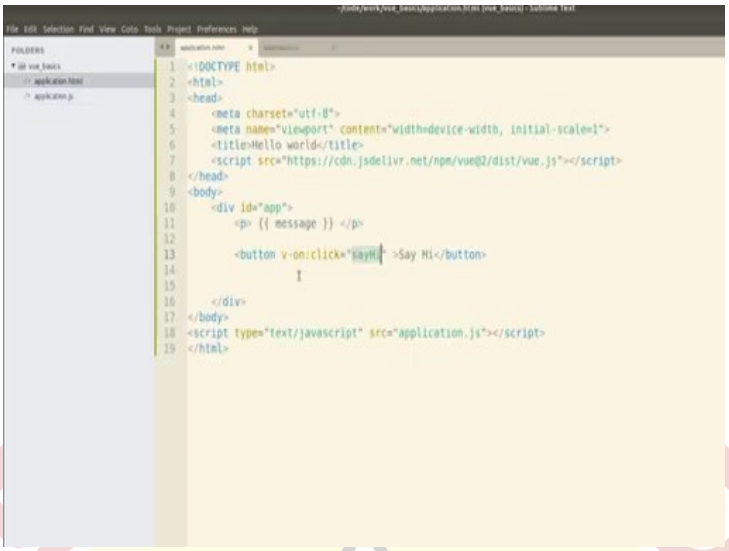
(Refer Slide Time: 12:38)




Hello world
file:///home/theji/code/work/vue_basics/application.html
Hello World
Say Hi


File Edit Selection Find View Goto Tools Project Preferences Help
FOLDERS
vue_basics
application.html
application.js
var app = new Vue({
el: "#app",
data:{
message :"Hello World"
},
methods : {
sayHi: function(){
this.message = "Hi"
}
}
})


Hello world
file:///home/theji/code/work/vue_basics/application.html
Hi
Say Hi



Now for it to act we need to introduce event handler, so on certain event on this button we need to be able to do something, usually you will do it by doing on click, that is what we learnt in our previous session, we do associate events directly. Now as you know we need to add an event handler, that is done by adding v on in Vue and this is, after this v on you actually give the event name which we want to handle which is click.

And what do we want to do? What is a event handler? Let us say we want to do say ‘hi’, and if I go refresh here and say hi, it throws an error because there is no ‘say hi’ defined, so let us define this ‘say hi’, like data the Vue object here can also have a methods attribute. We can define methods and then we can define our method called ‘say hi’. Let us do methods and we will define our first method called ‘say hi’.

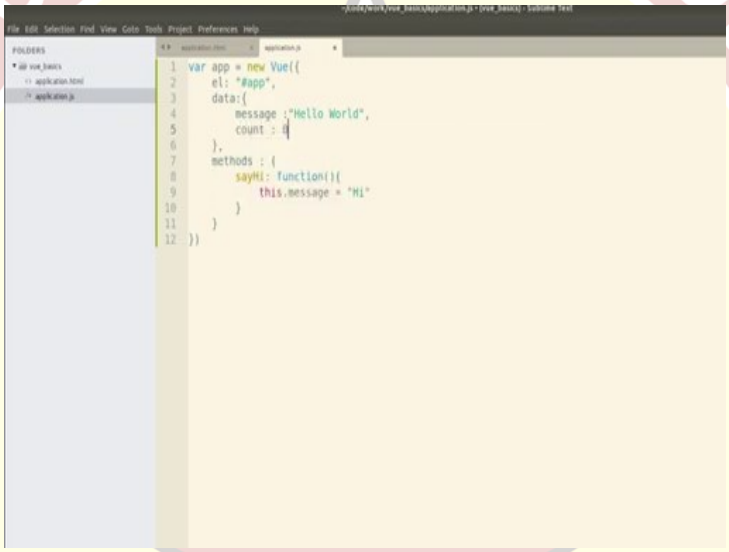
‘Say hi’ is equal to function; I am going to create an anonymous function. Now, I am just going to update this same message to say ‘said hello’ back. So, for example, we can do message is equal to ‘hi’, when say hi, I am just doing ‘say hi’. Now we still do not know where this message is, how do we refer this message. Since the message is from the same object we are going to do this dot message and then say hi.

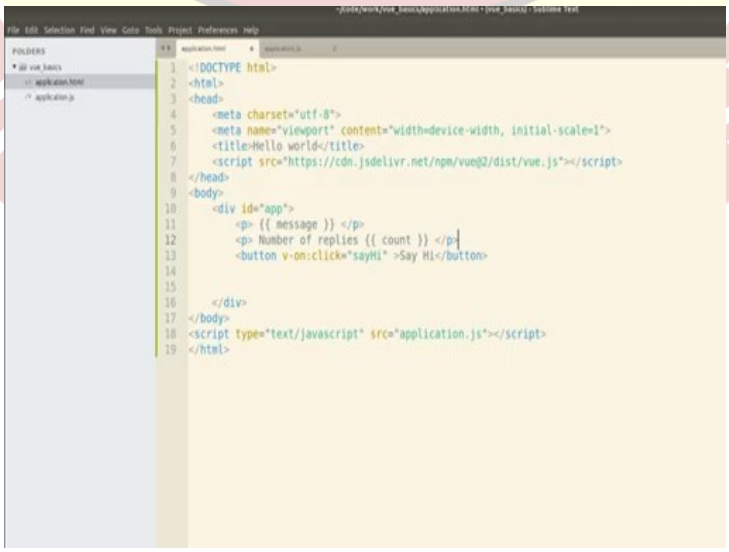
Let us go refresh it and say ‘hi’, it said ‘hi’. Now, what did we learn? We added a button, we added an event handler using v on and click is the event and we are calling an event handler called ‘say hi’ and then we implemented that ‘say hi’ as part of our Vue object, root object by defining methods attribute within that defining the function ‘say hi’.

Now, you can have multiple methods within the same Vue object and you can call them at various places. Now this is one way of handling event. Now, you can have many other events like say mouse over and other things. For example, if you want to do mouse over, we could add a mouse over event and then handle it. Before we go there let us do what we call attribute handling, attribute binding rather. That will make us, our messaging much more easier.

Let us say if you want to change something as soon as you click 'hi' as part of the message, let us say you want to make, not only make it 'hi' but you also want to count how many 'hi's' have been told to you. Let us add that.

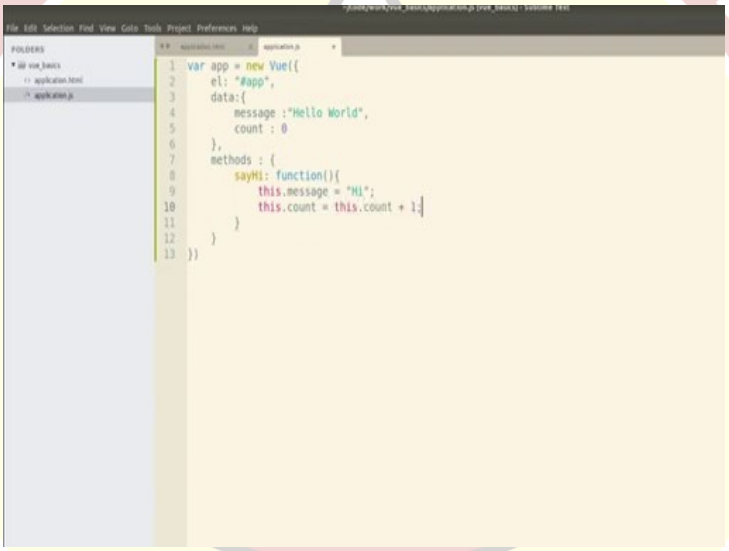
(Refer Slide Time: 16:17)

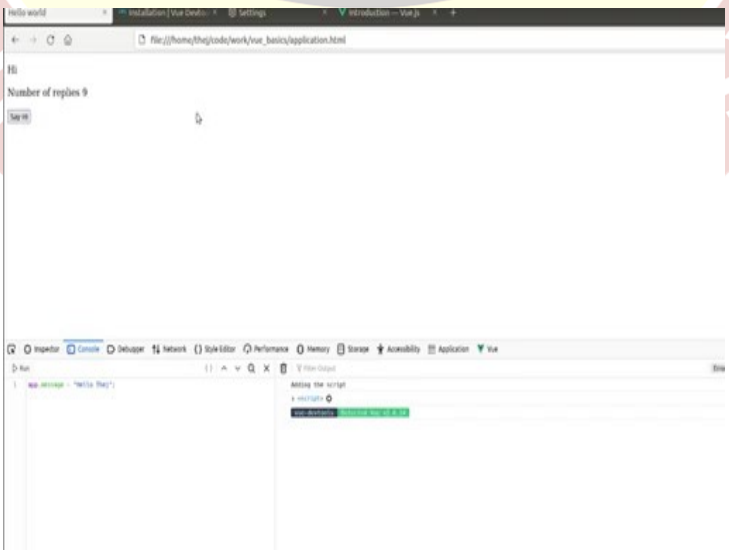





Hello world
Introduction — Vue.js
file:///home/theji/code/work/vue_basics/application.html
Hi
Number of replies 0
Say Hi

Inspector
Console
Debugger
Network
Style Editor
Performance
Memory
Storage
Accessibility
Application
Vue


File Edit Selection Find View Goto Tools Project Preferences Help
FOLDERS
vue_basics
application.html
application.js
var app = new Vue({
el: "#app",
data:{
message :"Hello World",
count : 0
},
methods : {
sayHi: function(){
this.message = "Hi";
this.count = this.count + 1;
}
}
})

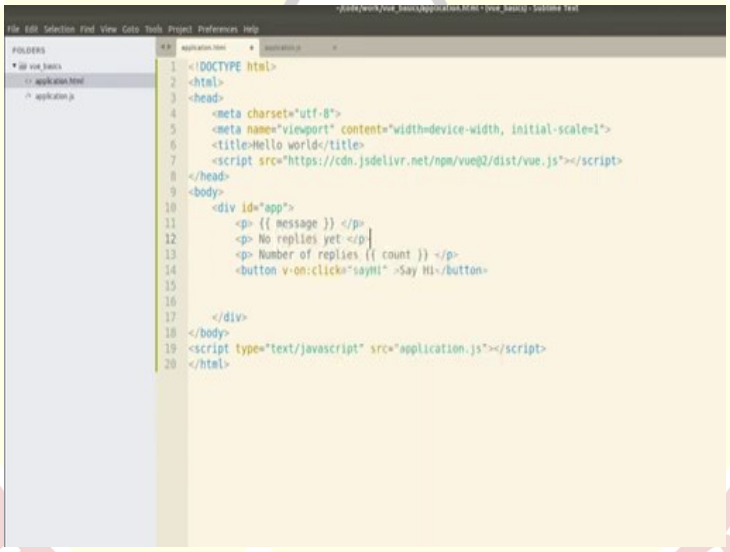

Hello world
Introduction — Vue.js
file:///home/theji/code/work/vue_basics/application.html
Hi
Number of replies 9
Say Hi

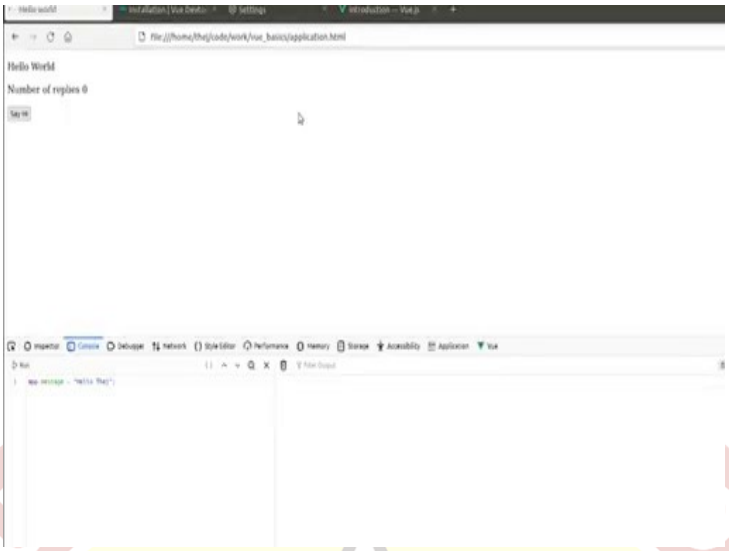
Inspector
Console
Debugger
Network
Style Editor
Performance
Memory
Storage
Accessibility
Application
Vue

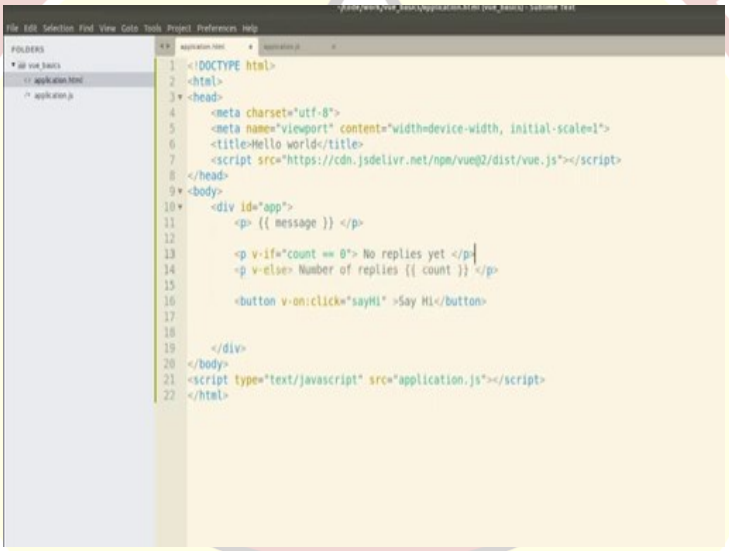
Let us take, define another data point called count and it starts with 0, and we can say here number of replies is count. Now, if I refresh it and click 'say hi' its became 'hi' but this is not increasing, this zero that is because on click of this we have not incremented the count, let us do that also. I will just put that, I will just this dot count is equal to this dot, or you can just increment it. Now refresh it, say hi, say it became, it is increasing.

It is the same thing that we are doing, it is just, we are adding another attribute called count and then incrementing and that referencing that here. Now, let us say we want to show some message based on this count. For example, we do not want to show, for example, number of replies is equal to zero, we want to show 'no replies' when number of reply is equal to zero and if there is more than one, then we will show number of replies.

(Refer Slide Time: 17:59)




Hello World
Number of replies 0
Say Hi


File Edit Selection Find View Goto Tools Project Preferences Help
FOLDERS
vue_basics
application.html
application.js
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<title>Hello world</title>
<script src="https://cdn.jsdelivr.net/npm/vue@2/dist/vue.js"></script>
</head>
<body>

 {{ message }}
 No replies yet
<p v-else> Number of replies {{ count }}
<button v-on:click="sayHi" >Say Hi</button>

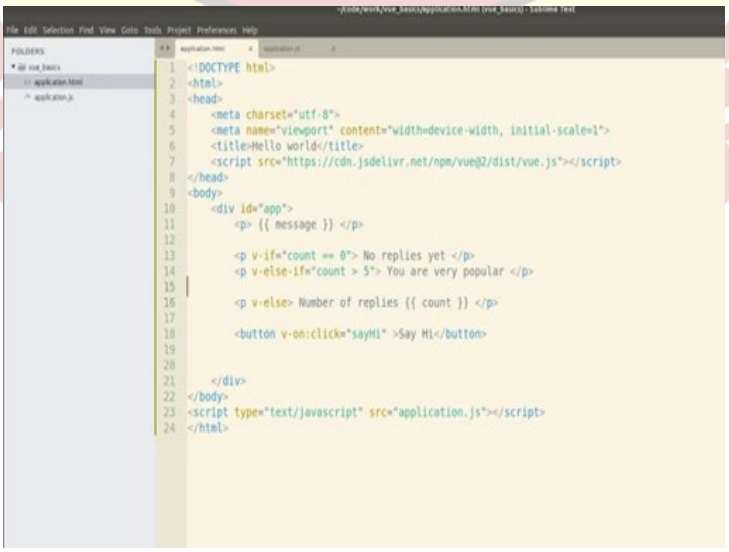
</body>
<script type="text/javascript" src="application.js"></script>
</html>

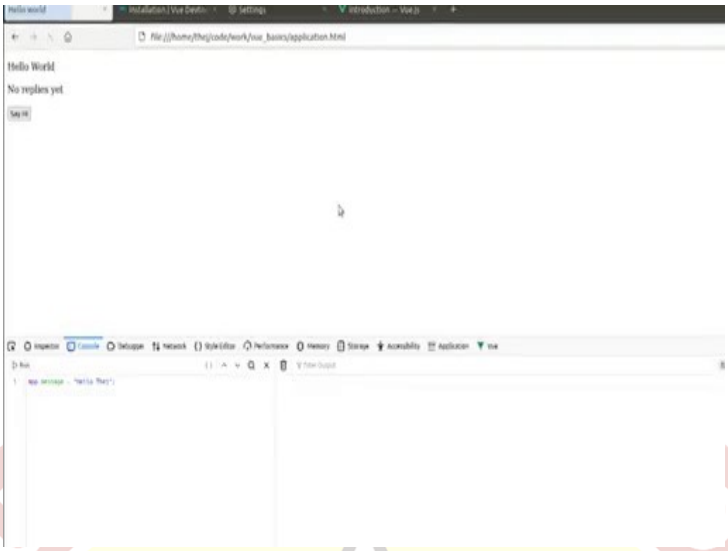

Hello World
No replies yet
Say Hi

Now that is like how we want to show, we want to show p no replies yet, and if I refresh this it will show both, but now I want to add like ‘if else’ based on the count. Now in Vue you can have conditionals based on the attribute and that is done by v if. So, let us add that. Let us add v if, let us a v if and our condition is when count is equal to is equal to zero.

We are going to show this and else we are going to show this, v.else, e-l-s-e, and so now let us see. It showed ‘no replies yet’, ‘say hi’ it just went and we have the new update. So, what is happening, when the count was 0 this was getting used, when the count became more than 0, then it got used.

(Refer Slide Time: 19:10)

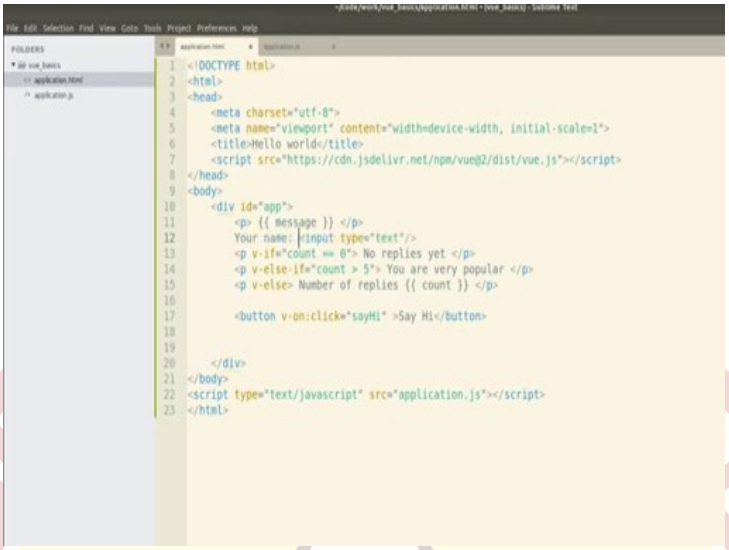


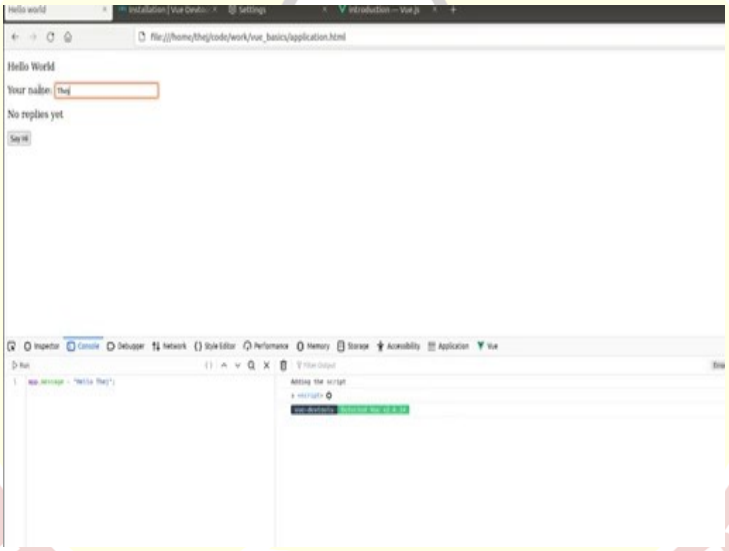


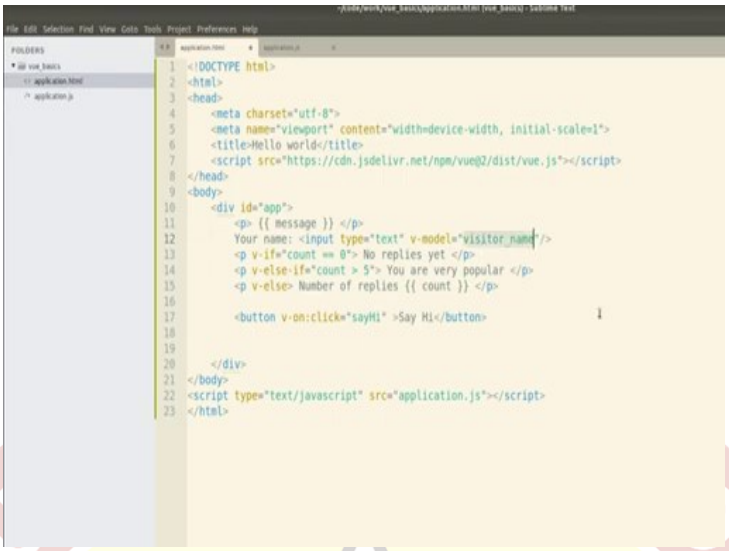
Now, let us add one more condition just for fun, if you are very popular, you got many replies, let us say five replies, then we will say 'you are very popular', I am just saying number of replies. Let us, we could add that many other conditions like v if instead of that v else if, we can add v else if if count is greater than let us say 5.

Let us say 10, or let us say 5 just to test and we can just put 'you are very popular' but from until 5 we just want to show number of replies. So, let us see what happens. Now, no replies yet, hi, hi, hi, you are very popular. So, once it reaches things, so this how you can use conditional statements in the Vue.

(Refer Slide Time: 20:49)



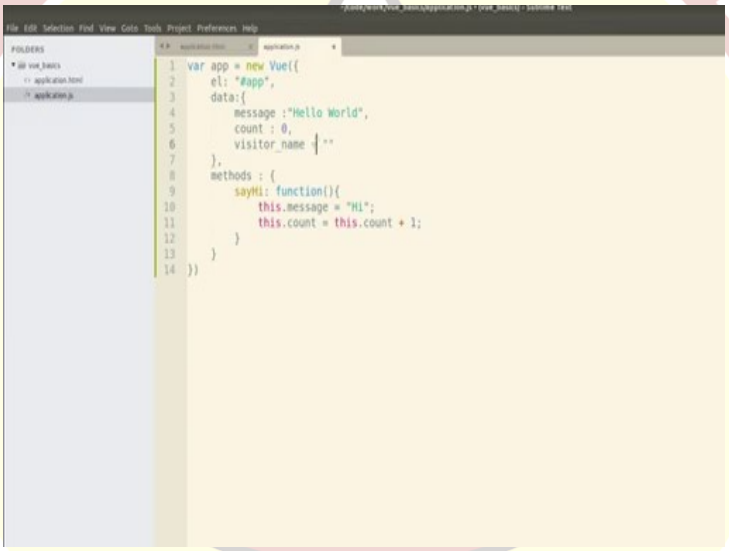




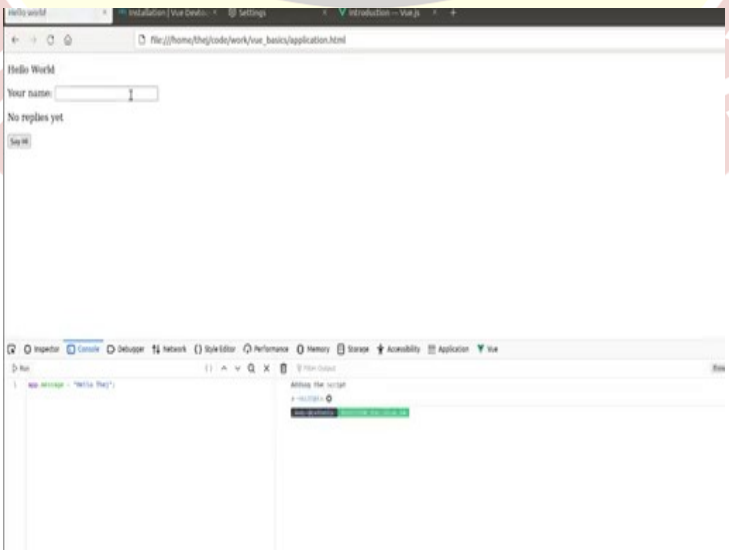
```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Hello world</title>
    <script src="https://cdn.jsdelivr.net/npm/vue@2/dist/vue.js"></script>
</head>
<body>
    <div id="app">
        <p> {{ message }} </p>
        Your name: <input type="text" v-model="visitor_name" />
        <p v-if="count == 0"> No replies yet </p>
        <p v-else-if="count > 5"> You are very popular </p>
        <p v-else> Number of replies {{ count }} </p>

        <button v-on:click="sayHi" >Say Hi</button>


    </div>
</body>
<script type="text/javascript" src="application.js"></script>
</html>
```

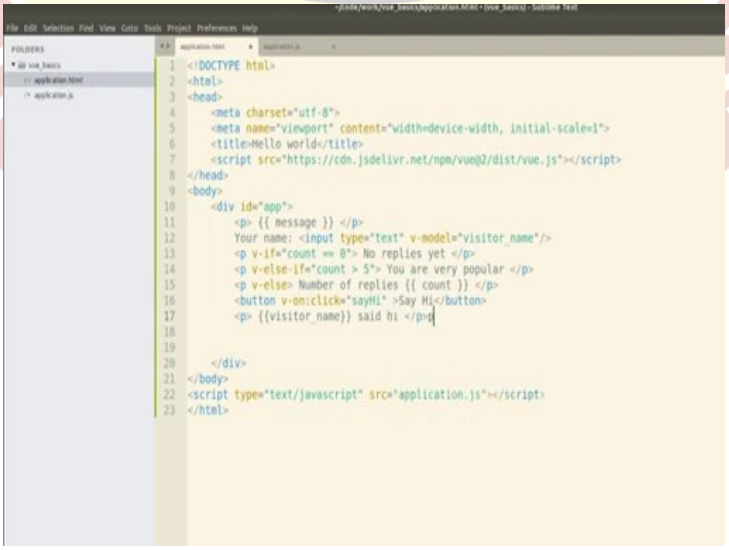


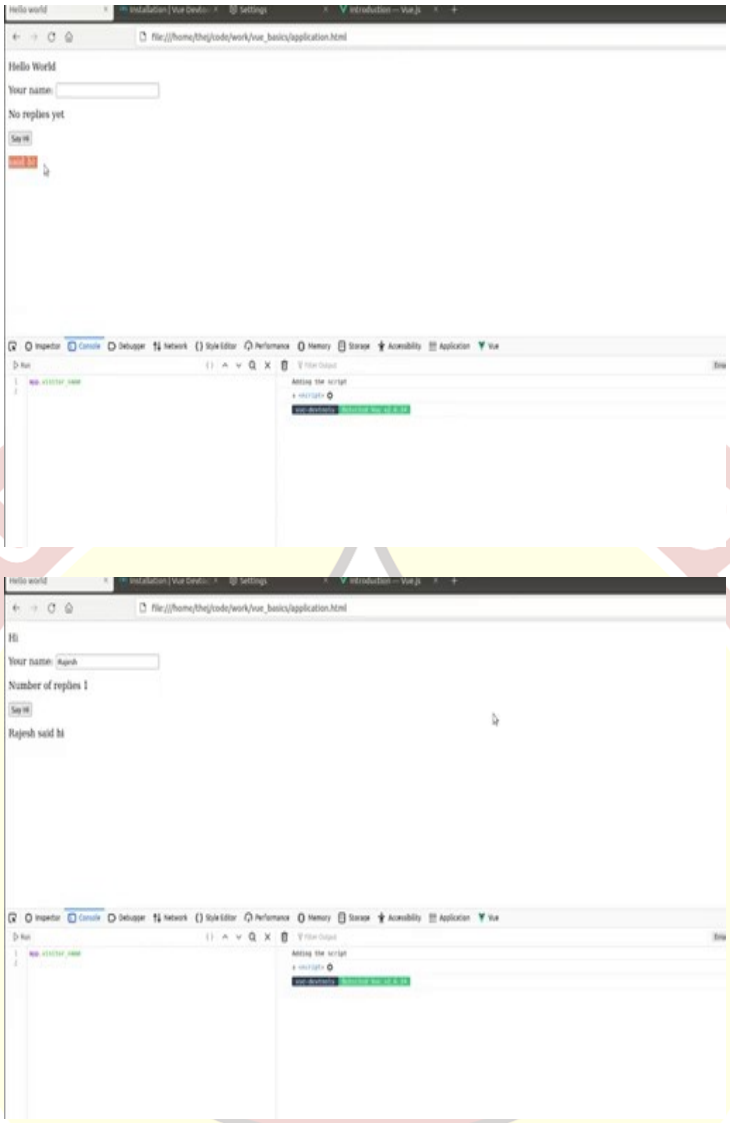
```
var app = new Vue({
    el: "#app",
    data:{
        message :"Hello World",
        count : 0,
        visitor_name : ""
    },
    methods : {
        sayHi: function(){
            this.message = "Hi";
            this.count = this.count + 1;
        }
    }
})
```




Hello world
file:///home/thej/code/work/vue_basics/application.html
Hi
Your name:
Number of replies 1
Say Hi


Hello world
file:///home/thej/code/work/vue_basics/application.html
Hi
Your name:
Number of replies 2
Say Hi


File Edit Selection Find View Goto Tools Project Preferences Help
FOLDERS
vue_basics
application.html
application.js
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <meta charset="utf-8">
5 <meta name="viewport" content="width=device-width, initial-scale=1">
6 <title>Hello world</title>
7 <script src="https://cdn.jsdelivr.net/npm/vue@2/dist/vue.js"></script>
8 </head>
9 <body>
10
11 {{ message }}
12 Your name: <input type="text" v-model="visitor_name"/>
13 No replies yet
14 <p v-else-if="count > 5"> You are very popular
15 <p v-else> Number of replies {{ count }}
16 <button v-on:click="sayHi" >Say Hi</button>
17 {{visitor_name}} said hi
18
19
20
21 </body>
22 <script type="text/javascript" src="application.js"></script>
23 </html>



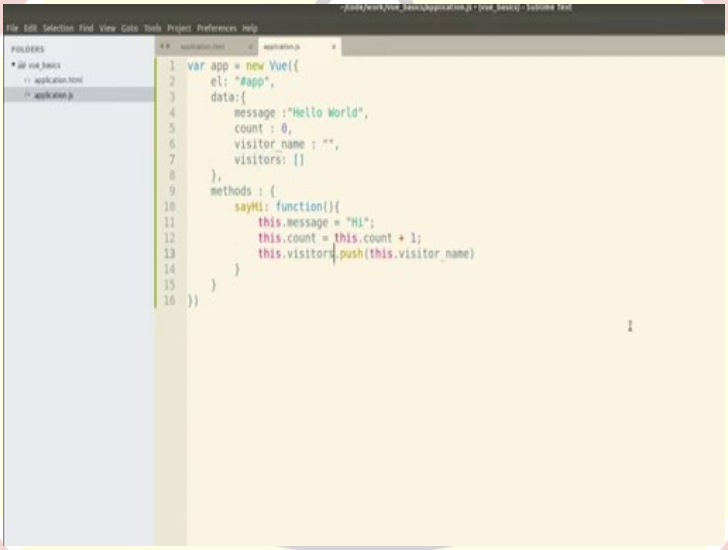
Now, we might also want to do some… taking some input from the user. Let us say we just do not want to say ‘hi,’ we want to know who is saying ‘hi’ twice, that can be done by adding inputs. Input type is equal to text, I am just going to say your name and if we look at it, say ‘hi’, not hi, Thej says hi, but nothing is happening, we are not doing anything to this.

Now, until now we saw one way of binding, we had data and whenever the change happened in the data it was going back to the html. But there was no other way to bind, like binding from actual html into data. Now, that can be done by something called v model. Let us say we add a v model, this is the one, we use to bind from the other way or two-way binding to be precise, then I will call it visitor name.

Now since visitor name is we are binding, we are going to add visitor name, by default will be empty. And then um we can just show visitor name also here. Let us say we want to show visitor name, we can just show here, let us see first of all whether this works. No, there is some error. Now, 'Hello World', you say 'hi', oh, sorry, let us do like Rajesh and then say 'hi'.

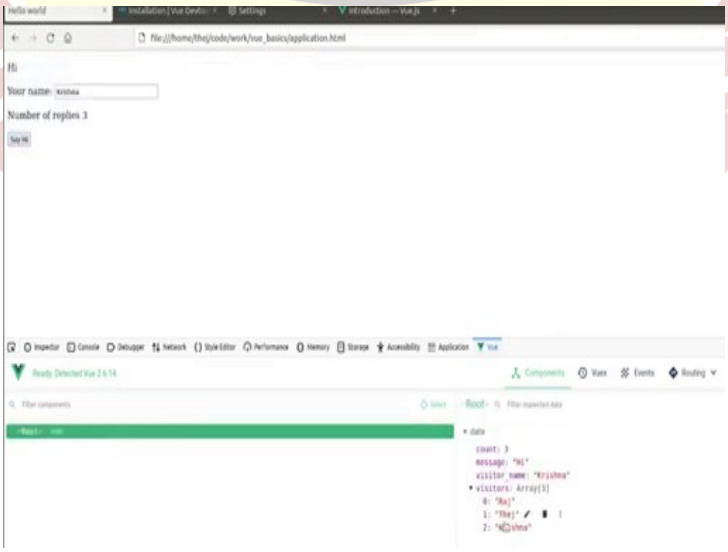
Now, let us check in the Vue, what is the value of this attribute, app dot that let us run it, you can see it has been set to Rajesh. Now you could also add a thing here called 'who said hi to you'. Let us say we can add p here. Now, it is saying, it is already saying hi, we will remove it later, but let us go with it. Now Rajesh, you can see Rajesh said hi.

(Refer Slide Time: 23:55)




Hello world
file:///home/thej/code/work/vue_basics/application.html
Hi
Your name:
Number of replies 2
Say Hi

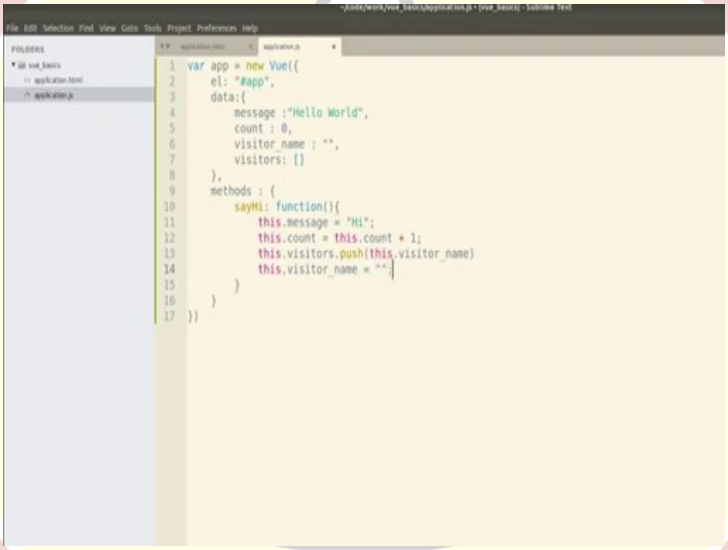

Hello world
file:///home/thej/code/work/vue_basics/application.html
Hi
Your name: Krishna
Number of replies 2
Say Hi

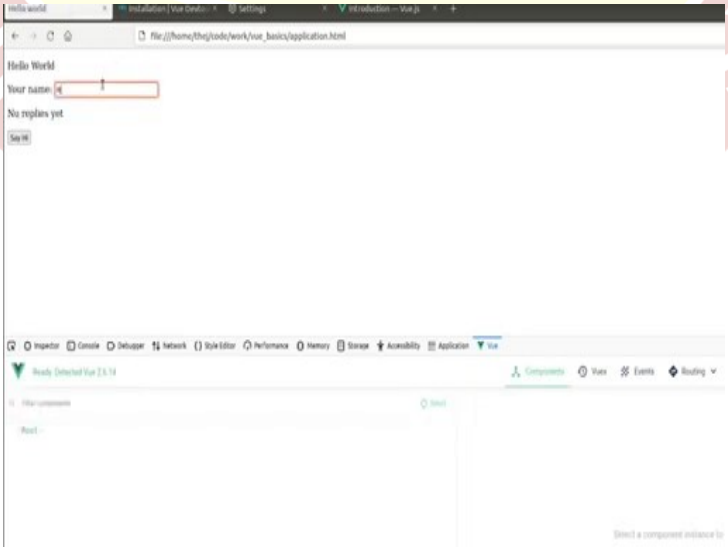

Hello world
file:///home/thej/code/work/vue_basics/application.html
Hi
Your name: Krishna
Number of replies 3
Say Hi

Now we do not want to do it, as soon as we change it here we do not want it to reflect directly, because we want to happen when we click on 'hi' and we wanted to add how many people. So, let us not do it here, instead let us add another variable called visitors and define an array. And then when somebody clicks 'say hi', what I will do is this dot, visitors dot push, which I am adding to the array, this dot visitor name.

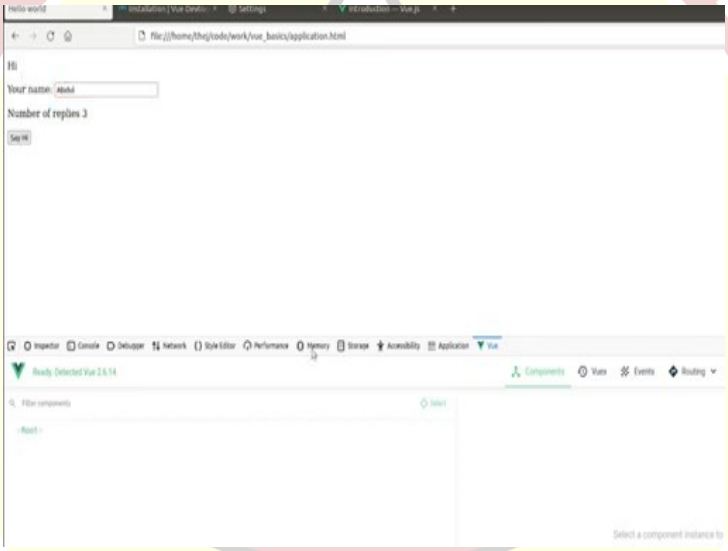
So, I am just going to do that and show how it looks and then we will do something with this visitors, spelling mistake, so let us fix it. Refresh, Raj 'say hi', Thej 'say hi', Krishna 'say hi', now if I go to my Vue and click on root you can see that the visitor's array has all the names, Raj, Thej and Krishna and the latest one is visitor named Krishna.

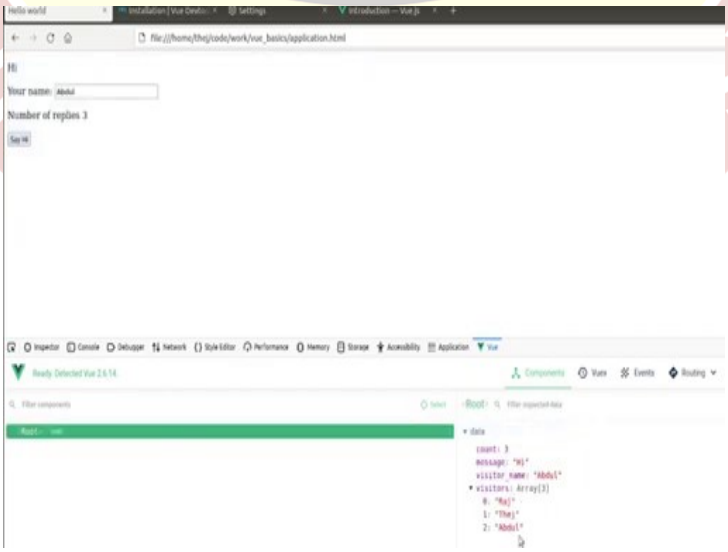
(Refer Slide Time: 25:25)

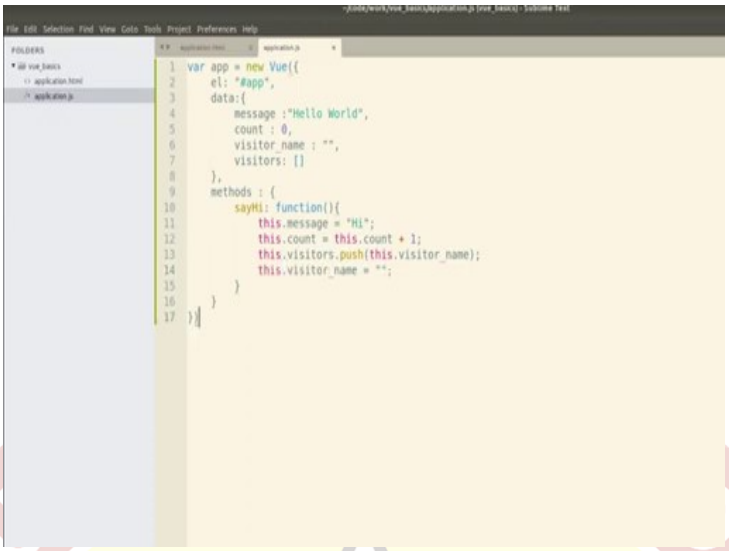




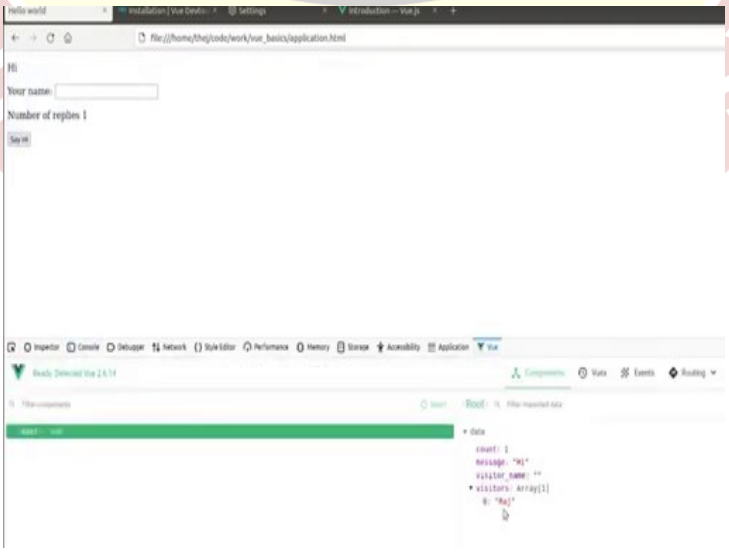

Hello world
file:///home/theji/code/work/vue_basics/application.html
Hi
Your name: Theji
Number of replies 1
Say Hi
Inspector Console Debugger Network Style Editor Performance Memory Storage Accessibility Application Vue
Ready. Detected Vue 2.6.14
Components Vuex Events Routing
Filter components
Select
<Root>
Select a component instance to

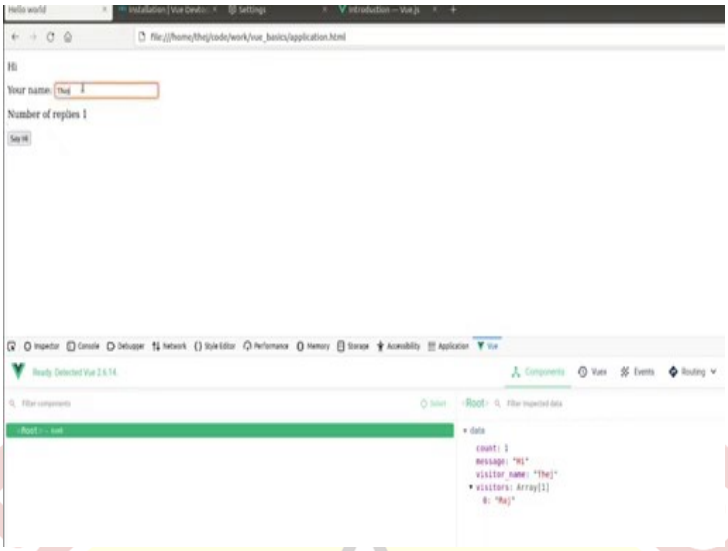

Hello world
file:///home/theji/code/work/vue_basics/application.html
Hi
Your name: Abdul
Number of replies 3
Say Hi
Inspector Console Debugger Network Style Editor Performance Memory Storage Accessibility Application Vue
Ready. Detected Vue 2.6.14
Components Vuex Events Routing
Filter components
Select
<Root>
Select a component instance to

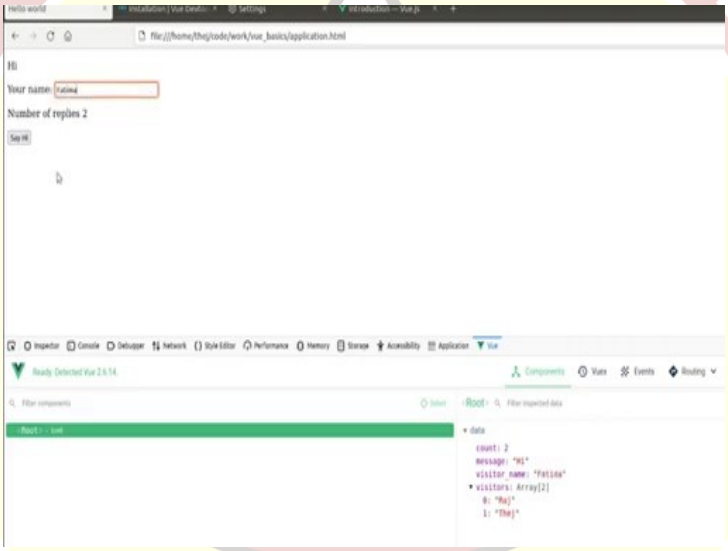

Hello world
file:///home/theji/code/work/vue_basics/application.html
Hi
Your name: Abdul
Number of replies 3
Say Hi
Inspector Console Debugger Network Style Editor Performance Memory Storage Accessibility Application Vue
Ready. Detected Vue 2.6.14
Components Vuex Events Routing
Filter components
Select
<Root>
<Root> vue
data
count: 3
message: "Hi"
visitor_name: "Abdul"
visitors: Array(3)
0: "Raj"
1: "Theji"
2: "Abdul"

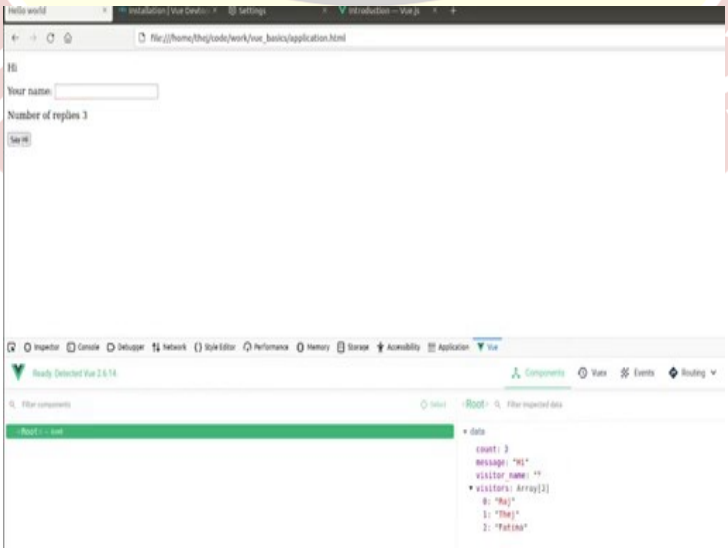


```
var app = new Vue({
    el: "#app",
    data:{
        message :"Hello World",
        count : 0,
        visitor_name : "",
        visitors: []
    },
    methods : {
        sayHi: function(){
            this.message = "Hi";
            this.count = this.count + 1;
            this.visitors.push(this.visitor_name);
            this.visitor_name = "";
        }
    }
})
```




Hello world
file:///home/theji/code/work/vue_basics/application.html
Hi
Your name: Theji
Number of replies 1
Say Hi
Ready. Detected Vue 2.6.14.
Components
Vuex
Events
Routing
<Root> - sub
data
count: 1
message: "Hi"
visitor_name: "Theji"
visitors: Array[1]
0: "Raj"

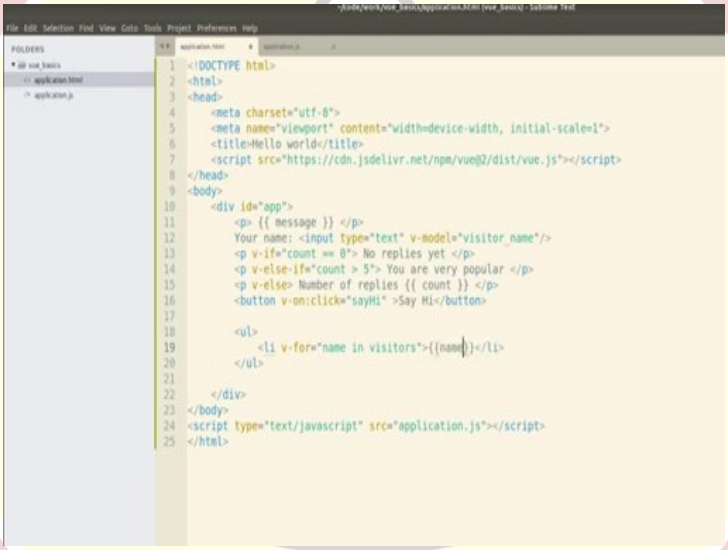

Hello world
file:///home/theji/code/work/vue_basics/application.html
Hi
Your name: Fatima
Number of replies 2
Say Hi
Ready. Detected Vue 2.6.14.
Components
Vuex
Events
Routing
<Root> - sub
data
count: 2
message: "Hi"
visitor_name: "Fatima"
visitors: Array[2]
0: "Raj"
1: "Theji"

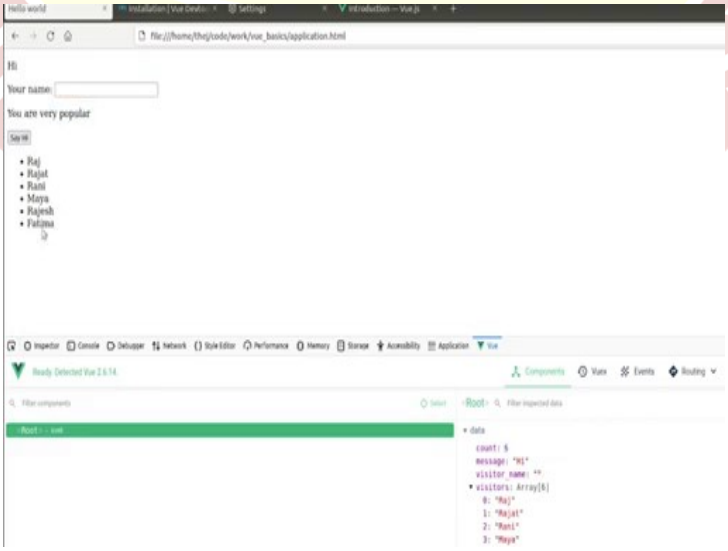

Hello world
file:///home/theji/code/work/vue_basics/application.html
Hi
Your name:
Number of replies 3
Say Hi
Ready. Detected Vue 2.6.14.
Components
Vuex
Events
Routing
<Root> - sub
data
count: 3
message: "Hi"
visitor_name: ""
visitors: Array[3]
0: "Raj"
1: "Theji"
2: "Fatima"

I just want to make one more change, as soon as you enter and click ‘say hi’ I want to clear this so that people know that it has been used, so that you can do by resetting back to empty, this dot visitor name is equal to empty. So, let us try once more, say Raj ‘hi’, Thej ‘hi’, Abdul says ‘hi’ and now if I click on root you can see Raj, Thej and Abdul have said ‘hi’.

It is not clearing, so here is the problem, I put comma instead of dot. Now, if you refresh it again Raj, say hi, it goes away, but you can go back to your Vue and click on root, you can check, it is still there as part of array. Now, if I do Thej ‘say hi’ it works, Fatima, works again. Now, we know that we got three replies, we have also captured their name, let us show their name below, that can be just done by iterating over visitors and showing it.

(Refer Slide Time: 26:59)



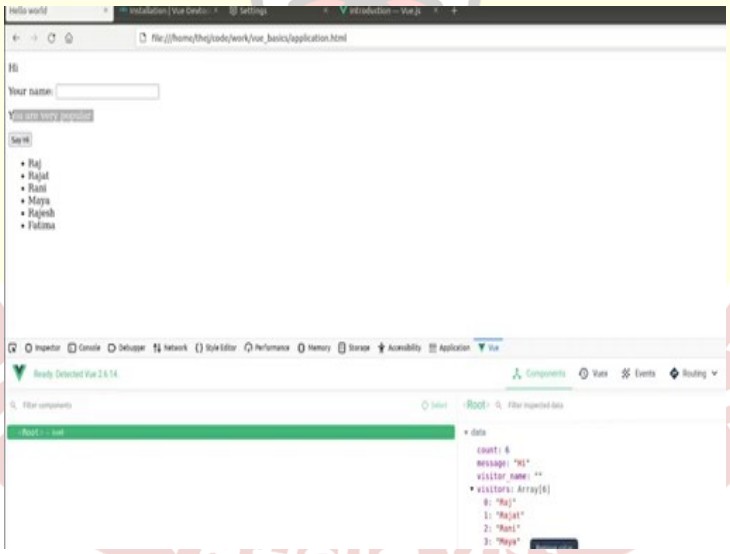


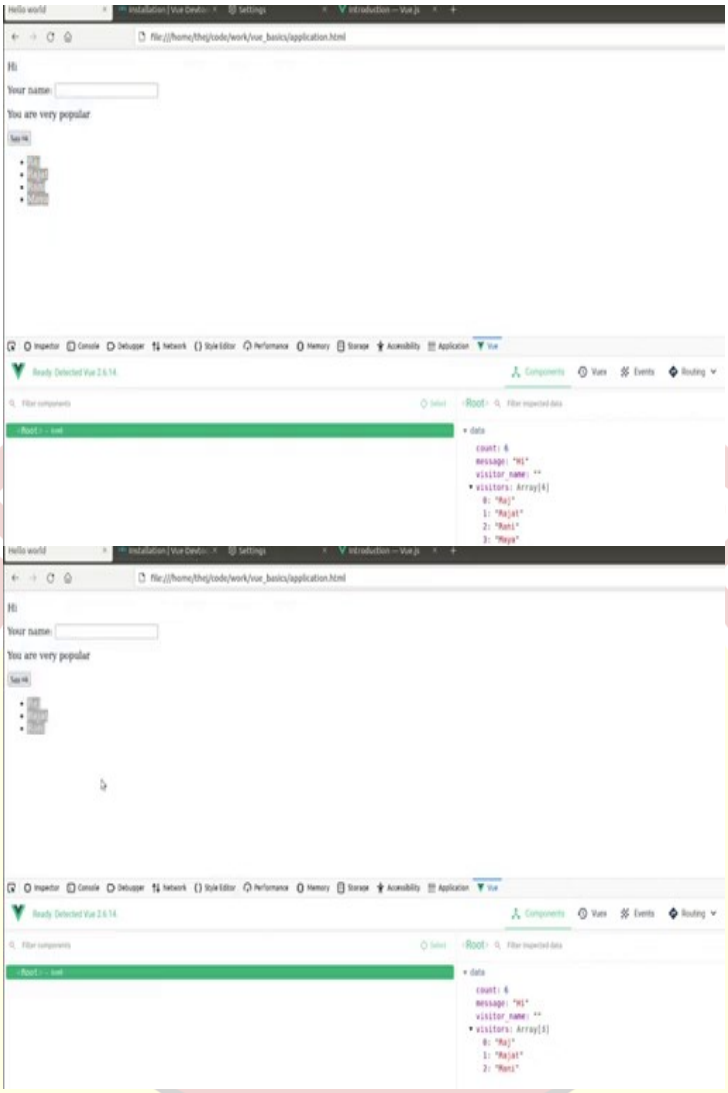
I mean, let us do that using what we call ‘looping in Vue’, that can be done using looping. So, let us create an unordered list, starts within ul. Now, you want to create as many li's as there are names in the visitors. So, let us do that by li, and then I will just close the li. Within the li I will just do v for, this loops over something, what does it loop over, it should loop over visitors, it should loop over visitors, that is done using name in visitors.

So, it will loop over visitors and each visitors name will appear in the name, one by one, and then that we want to print here.So, if you put name here, this should actually work. Let us see. I will just click on root here just to make sure that we are seeing everything. Raj ‘say hi’, so it came down, as you can see, Rajat, Rani, Maya.

You can see it is getting added to the below to the visitors and as soon as it gets added to the visitors here it gets added to the thing here and let us say Rajesh and as soon as Rajesh gets added it gets added. Now, let us add one more to see the other change as well. We will make it Fatima, say hi, so it added Fatima, it also became ‘you are very popular’.

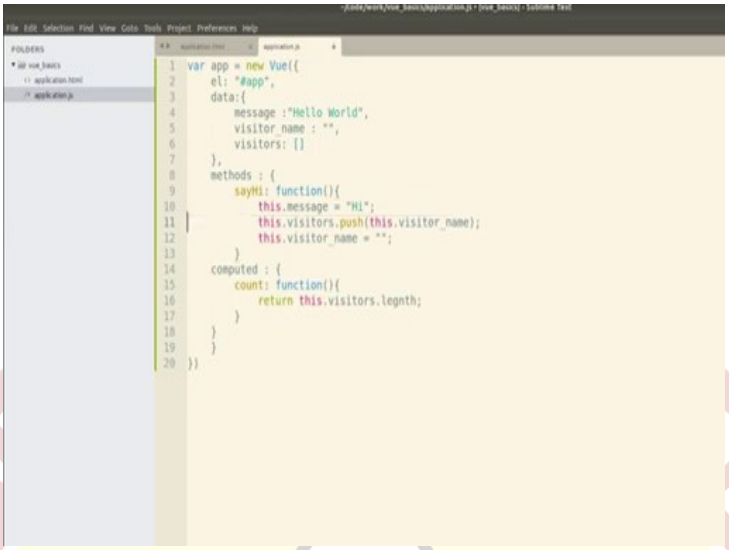
(Refer Slide Time: 29:02)

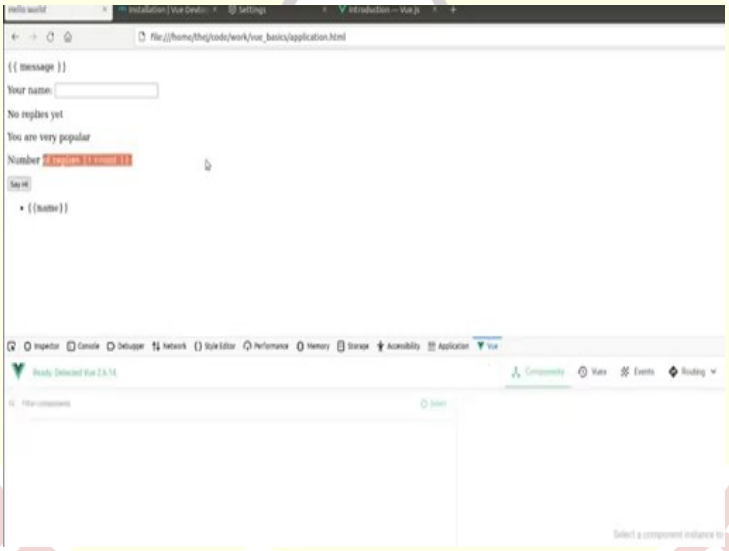


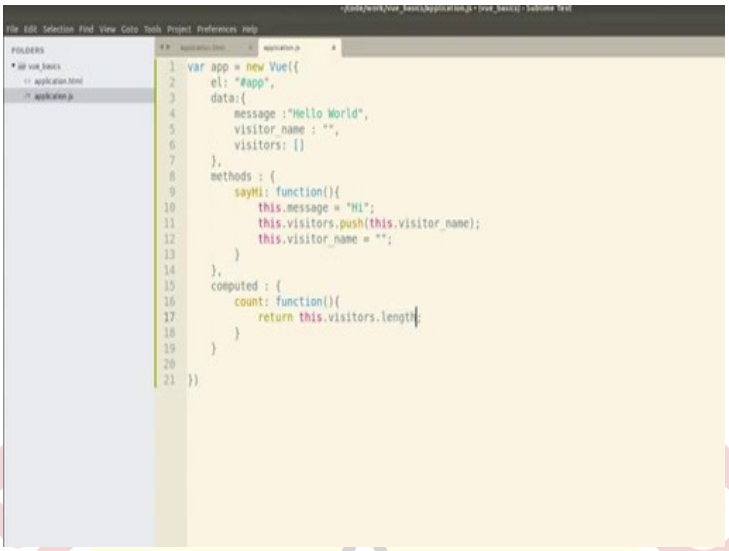


Now, let us do one thing, we can just delete this from our data through some other process, let us say, you can see it got away from here too. This will not run, so that is how it is two-way.

(Refer Slide Time: 29:22)

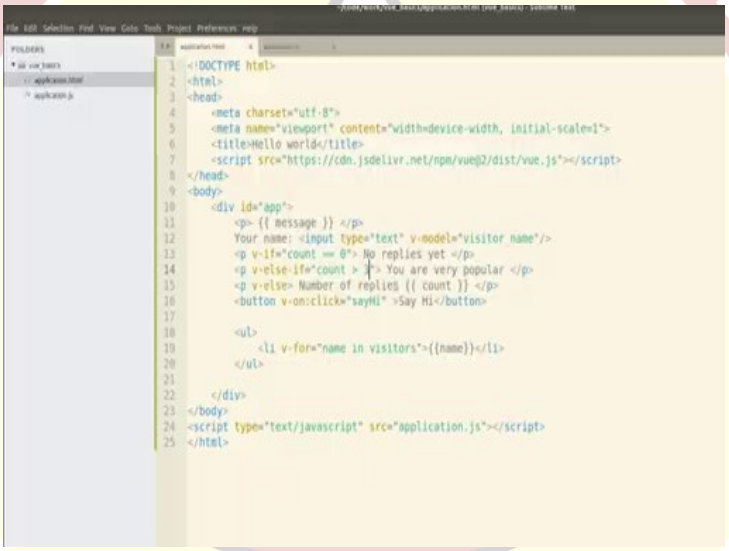






```
var app = new Vue({
    el: "#app",
    data:{
        message :"Hello World",
        visitor_name : "",
        visitors: []
    },
    methods : {
        sayHi: function(){
            this.message = "Hi";
            this.visitors.push(this.visitor_name);
            this.visitor_name = "";
        }
    },
    computed : {
        count: function(){
            return this.visitors.length;
        }
    }

})
```



```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Hello world</title>
    <script src="https://cdn.jsdelivr.net/npm/vue@2/dist/vue.js"></script>
</head>
<body>
    <div id="app">
        <p> {{ message }} </p>
        Your name: <input type="text" v-model="visitor name"/>
        <p v-if="count == 0"> No replies yet </p>
        <p v-else-if="count > 2"> You are very popular </p>
        <p v-else> Number of replies {{ count }} </p>
        <button v-on:click="sayHi" >Say Hi</button>

        <ul>
            <li v-for="name in visitors">{{name}}</li>
        </ul>

    </div>
</body>
<script type="text/javascript" src="application.js"></script>
</html>
```

Now, if you want this also to change according to the count in this, then we will have to do not on the count plus, then we will have to do on the visitor name plus, so instead of doing count, we could actually take a count of visitors and then do that as a thing, so that you do not have to depend on one and not on the other.

So, for example, that you can do by removing the count here and you can do something called computed. Now, we can say count, and make it a function and what will it return, it will return the length of a array. Return, this dot, visitors dot, let us see, now v in, instead of actually attribute we are computing it, so that both are in sync, and I do not need this because we are just indirectly counting it using the names.

Let us see this, number of replies; it is coming as count, because I think count initially is zero, that happened because I think there was a bug here, so there is a spelling mistake in the length, so I fixed it. Now, let us try again, like you can add, so I have also made it greater than two, let us just make it three, let us start Raj, Thej, Rahim, Shweta, so I can see that now it changed based on the computed count instead of actually we keeping the count.

So, that is how you can create the computed attribute and this is going to be very useful if the value of attribute depends on multiple other attributes, you could use many other calculations here. So, until now we saw how to create Vue object, how to set values, then how to bind using v on, bind the events using v on and then how to bind the two way data by using v model, then we used if else conditionals, then we used for loops.

This completes the basics of Vue.js, in the next session we will look how to use some of the advanced features of Vue.js. Thank you so much for watching.