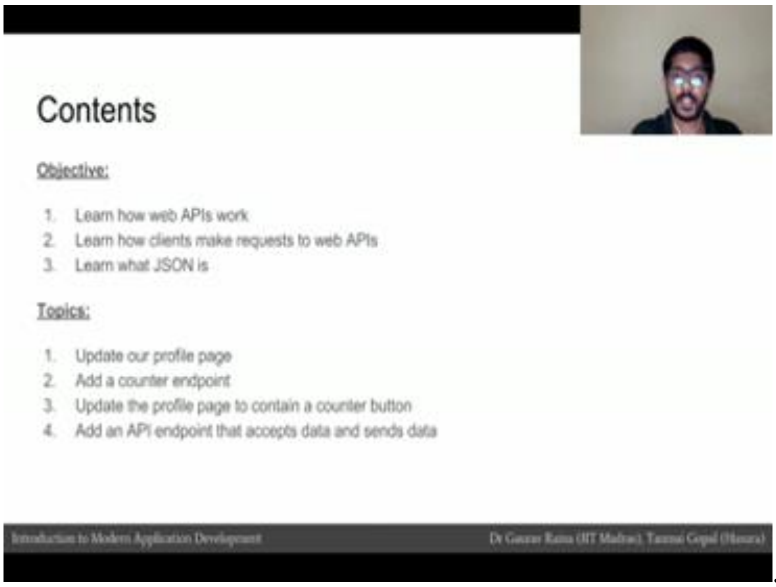
Introduction to Modern Application Development

Prof. Tanmai Gopal
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Module - P6
Lecture - 12
Practical: Introduction to APIs

Hi guys, welcome to Module - P6 in this module we will be taking an introduction to APIs from both the server, and the client side.

(Refer Slide Time: 00:00)



Contents

Objective:

1. Learn how web APIs work
2. Learn how clients make requests to web APIs
3. Learn what JSON is

Topics:

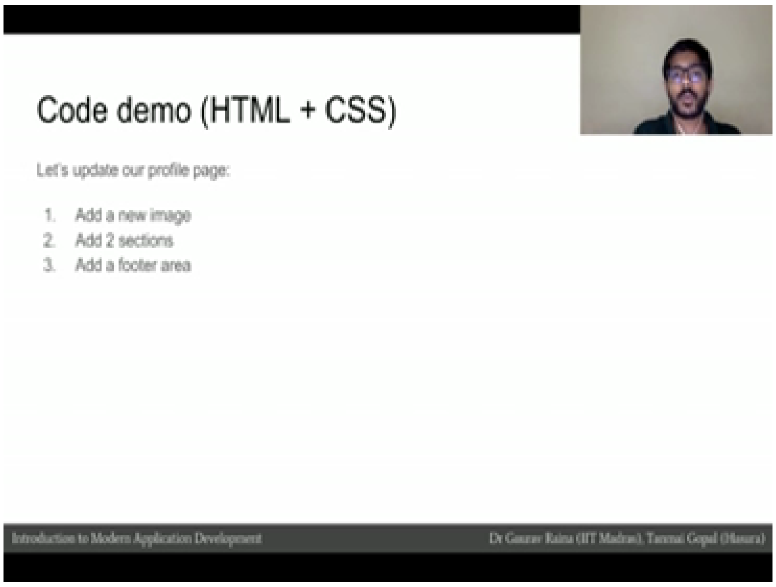
1. Update our profile page
2. Add a counter endpoint
3. Update the profile page to contain a counter button
4. Add an API endpoint that accepts data and sends data

Introduction to Modern Application Development   Dr Gaurav Raina (IIT Madras), Tanmai Gopal (Hasura)

.

Our main objective in this module is to learn how, web APIs work how clients make request to web APIs and also what JSON is which is a very popular format and almost another modern default standard for exchanging information, while making API request and receiving API responses we will do this by solving several small tasks, over this module and the very first thing that we will do is update our profile page.

(Refer Slide Time: 00:31)



Code demo (HTML + CSS)

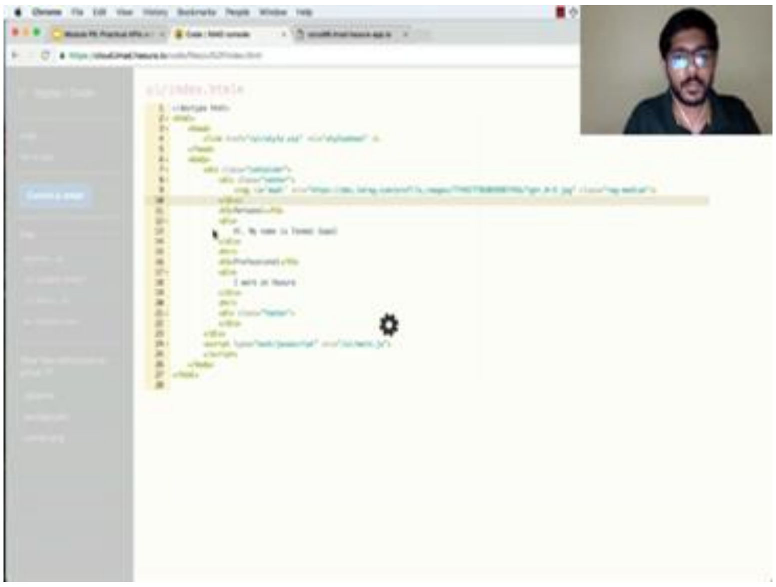
Let's update our profile page:

1. Add a new image
2. Add 2 sections
3. Add a footer area

Introduction to Modern Application Development   Dr Gaurav Raina (IIT Madras), Tanmai Gopal (Hasura)

We will add a new image we will add 2 sections, and we will add a footer area on this modified profile page, we will start making changes to be able to make API requests and receive responses to API requests alright. Let us head to our coding console.

(Refer Slide Time: 00:46)



src/index.html

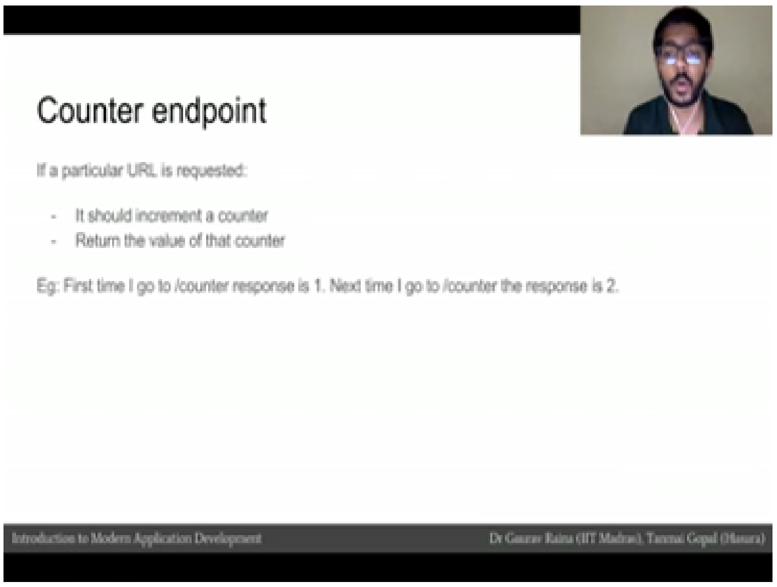
```
<!doctype html>
<html>
    <head>
        <link href="src/style.css" rel="stylesheet" />
    </head>
    <body>
        <div class="container">
            <div class="center">
                <img id="dp" src="[illegible]" class="[illegible]">
            </div>
            <h3>Personal</h3>
            <div>
                Hi, My name is Tanmai Gopal
            </div>
            <br/>
            <h3>Professional</h3>
            <div>
                I work at Hasura
            </div>
            <br/>
            <div class="footer">
            </div>
        </div>
        <script type="text/javascript" src="src/main.js">
        </script>
    </body>
</html>
```

The first thing that we want to do is change the image and I will go to my twitter profile click on this image, right click open image in new tab grab this image and replace that here. Now, I will have a different image that comes up I also want to use the same styling that we were, using for article 1, 2, 3 which was that we had a container, and where all the contact was centered.

I will do def class equal to container and I will move all of these 1 step inside I will disable this javascript because, I do not want an alert box to come up and wasting the time I load my page and That is my that is my image let us put a horizontal line here let us, add a small heading that says personal and I am def and we can write here hi my name is and now, let us add a block let us create another horizontal line and this is the professional information about me.

Let us say professional and I will say hi I work at, let us have another horizontal line and now here we will add let us say a small footer area we are going to call it of its not really a footer, and we will not put any content here for now, let us just see what this looks like cool. I have a little profile page. Now and what we will do is I can see that there is a lot of extra space here which I do not want. I am just going to remove both the horizontal line and the brick and Then this space will vanish and that is what is an IP look like.

(Refer Slide Time: 02:46)



Counter endpoint

If a particular URL is requested:

- It should increment a counter
- Return the value of that counter

Eg: First time I go to /counter response is 1. Next time I go to /counter the response is 2.

Introduction to Modern Application Development — Dr Gaurav Raina (IIT Madras), Tanmai Gopal (Hasura)

Now this is the staff that we have let us save this. So; now, what we want to do is we going to add a counter end point, and what this counter end point will do is it will be a new URL on our server just like say slash or slash article 1, or article 2. We are going to have a new end point, and what this end point will do is that it will increment a counter, the first time I go to this end point it will give me a 1, the next time I go it will give me a 2, if somebody else goes to this end point it will be 3 then, if I go to the end point it will be - 4. Let us let us see what this is what that would look like.

(Refer Slide Time: 03:36)

Code demo (server-side JS)

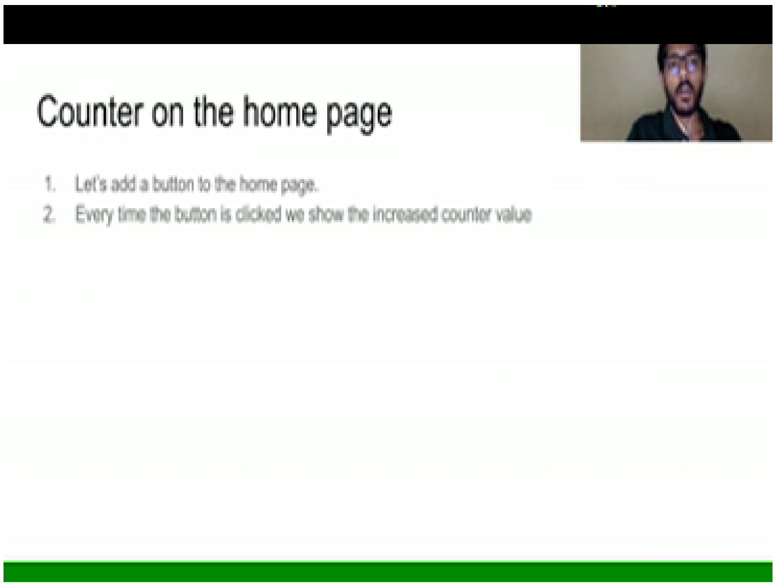
Let's write this simple counter URL

Introduction to Modern Application Development   Dr Gaurav Raina (IIT Madras), Tanmai Gopal (Hasura)

I go to server dot j s and I am going to add a line of quote that says slash counter, and if the slash counter request is made use this function and this function takes the value of counter and increments it by 1 it responds with that value of counter, but counter is a number. Let us convert that to a string using 2 strings, we need to initialize the value of counters. Let us say where counter equal to 0 so; that means, counter is initialized to 0 when the server has starts when the sever started and every time he go to a slash counter the counter variable will be incremented, and then the responsibly sent which is that counter.

Now, remember that you can only send a string as a response you cannot really send a number as a response. The number has to be converted to a string format to be sent as a response. Let us save this and let us go to our end point. Let us go to slash counter, wait for our server to restart and you can see that a 1, has appeared I refresh second 2, 3, 4, 5. In fact, I can go to some other page make a ton of these requests, and it becomes twelve and. If I refresh this it should be 13 and. Now, you can see 13 here so; that means, anybody go into this counter will increment this counter value. So, we added a counter end point by adding some server side javascript.

(Refer Slide Time: 05:09)

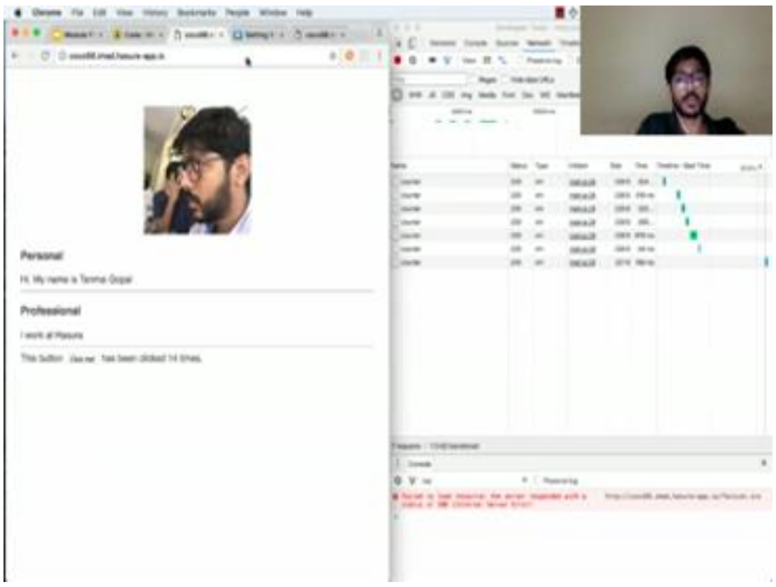


Counter on the home page

1. Let's add a button to the home page.
2. Every time the button is clicked we show the increased counter value

Let us now, add a counter button on the homepage and every time the button is clicked, we show an increased counter value. Let us see what this code looks like.

(Refer Slide Time: 05:15)



Personal

Hi. My name is Tanmai Gopal

Professional

I work at Hasura

This button Click me has been clicked 14 times.

First thing is that we will have to go to our homepage, which is index dot HTML and in our footer let us add some quote who says this button and then, let us go to button here. Let us say button let us give the button name. There is a button called counter its call the button. This button has been clicked 0 times, and what I have done is I have given this button a name and I have given this area which is called as span, I have given this span a name.

So, by default when I look at the index or a (Refer Slide Time: 05:56) this button has been clicked 0 times. Let us save this let us see what our server responds, with to the homepage. So, you can see that I have a little photo that says, this button has been clicked 0 times. Let us let us actually add the javascript to this to make the request, and then make and then see the latest value of the counter here. Let us go to our main dot j s which is where our javascript code is. Let us remove the old code.

The first thing that I want to do is select a button and. Let us get our button by doing a document dot get element by I d counter and then when this button is clicked let us attach a function to it and what should this function, do well ideally this function should first make a request to the counter end point and capture the response and store it in a variable. So, render the variable in the correct span.

Let us try to do just this last bit first which is rendering the variable in the correct span because you already, know how to do this. So, we will say that if we have a counter variable let us assume that a counter variable came and for now, let us just add that code here and I will say that my span, let us select the span in the document. I say document dot get element by I d and I will select count and then I will set the inner HTML to the counter value, let us initialize the counter so; that means, every time I click on this the counter variable will go up by 1 and then it will take shown here.

But this is this is not what we want to do because we have not done the request part yet. Let us just check if this is working save this the restart our server and. So, our servers loaded, now if I click on it you can see the value is increasing right and this has nothing to do the with the counter and points the counter and point is still probably at 1 and if we try that out you can see the counter end point is just at 1 right.

Let us now see how we make a request. The first thing that we do is create a request variable and use something called a XML http request, this is the documentation for this you can find a developer dot. So, you see XML http request, let us see how a request is made. I go to the documentation for XML http request, and let us go to an example right. The way a request object works is that once a request is made the browser will tell us if the response is received, and a request can be in multiple stages. So, request can be a request is just opened a request has been sent the response is loading the response has loaded successfully.

There are various states that a request can be in and in our case when we want to capture the response. What will have to do is detect the change in the state. So, request dot on ready state change if the request state changes then, execute this function and what this function does is that it checks the current state of the request object and if that is equal to and if that is equal to x m l http request dot done which means that the request has successfully completed we want to take some action, but if its not done yet we do not want to do anything right. Let us ignore that, what is the action that we want to take we first want to check if this was a successful request. Then what we want to do is check if request dot status is 200.

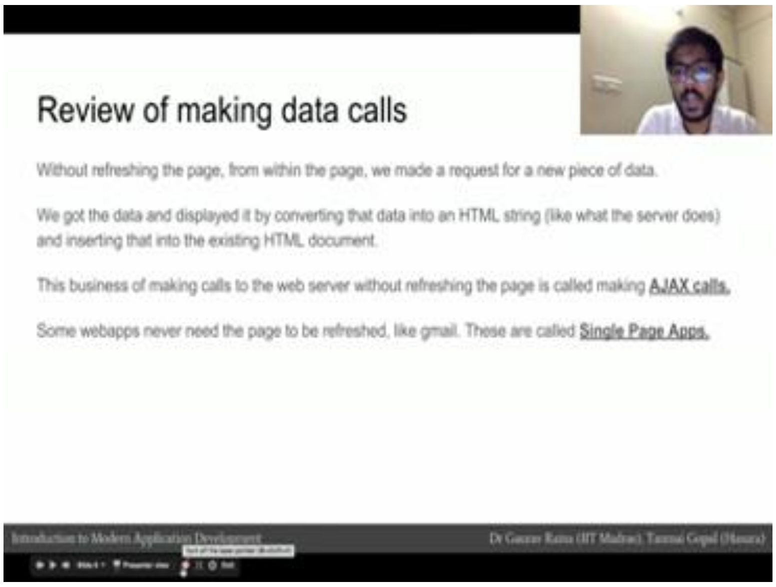
Now, if the request status is 200; that mean, that the request status has successfully completed; then what we want to do is extract the value from this request. So, what we want to do is we want to say request dot see want to capture the response text. So, we will say request dot response text and this gives us access, to the respond value which will have a number and we will say that our counter variable. So, you will say var counter is equal to this response text. Then what we will do is we will change and we can remove the old code here is the old code here as well.

Now, what we are doing is we are saying that if the request results in 200 the request status is, 200 then take the response and put it in the counter variable and take the counter variable and put it inside the span. This is what we do once we receive the response from the request right, but let us we have not actually made that made the request yet. Let us make the request right and let us change this comment to create a request object.

So, javascript makes us think in strange ways sometime because we need to think about what to do once we receive the request response, before we actually make the request right and. Now, let us make the request by doing a request dot open and we want to make a get request to our own servers that is coco 98 dot 99 dot (Refer Time: 00:00) dot (Refer Time: 00:00) I o and we want to go to the counter variable; this when I am actually making the request. Once this request is made and the request state changes to the request is done then our code will be executed here. Let us try to see this in action refresh our page here; cool.

Let us click on this and you can see that its working right its changed here let us go to inspect element and go to the network tab and let us just clear this. So, resize this. So, every time I click here you can see that a request is made right every time I am clicking here, and your request is made which also means that if I go the counter end point by myself. The value has become 13 once, I click on this button the new value will be 14 right because that is the counter variable that is been. This is me making a request and getting the response and showing the response on this page without refreshing the page so.

(Refer Slide Time: 13:34)



Review of making data calls

Without refreshing the page, from within the page, we made a request for a new piece of data.

We got the data and displayed it by converting that data into an HTML string (like what the server does) and inserting that into the existing HTML document.

This business of making calls to the web server without refreshing the page is called making **AJAX calls.**

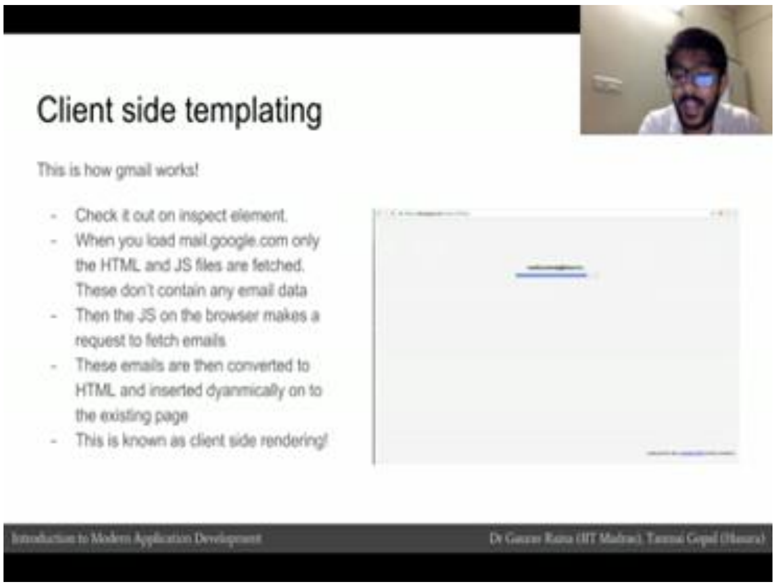
Some webapps never need the page to be refreshed, like gmail. These are called **Single Page Apps.**

Introduction to Modern Application Development — Dr Gaurav Raina (IIT Madras) Tanmai Gopal (Hasura)

Now, what we did is we actually made a data call we did not make a request for an HTML page we did not make a request for a javascript file or a CSS file, but we actually made a request just for a piece of data and then we took that piece of data, and we converted it to an HTML string and converting the data into an HTML string was very similar to what the server did and once you convert it that into an HTML string, we insert it that HTML a string into the existing HTML document.

This business of making calls with the web server without refreshing the page is called making AJAX calls. In fact, some modern web apps these days do not need to be refreshed at all. So, you will never see a page load a prime example; of this is an app like Gmail these web apps are called single page apps because there, is only 1 page that loads and all the action all the change in data and all the change in the u I happens within that single page this activity of templating on the browser, and where I used the word templating to refer to converting data into HTML.

(Refer Slide Time: 14:26)



Client side templating

This is how gmail works!

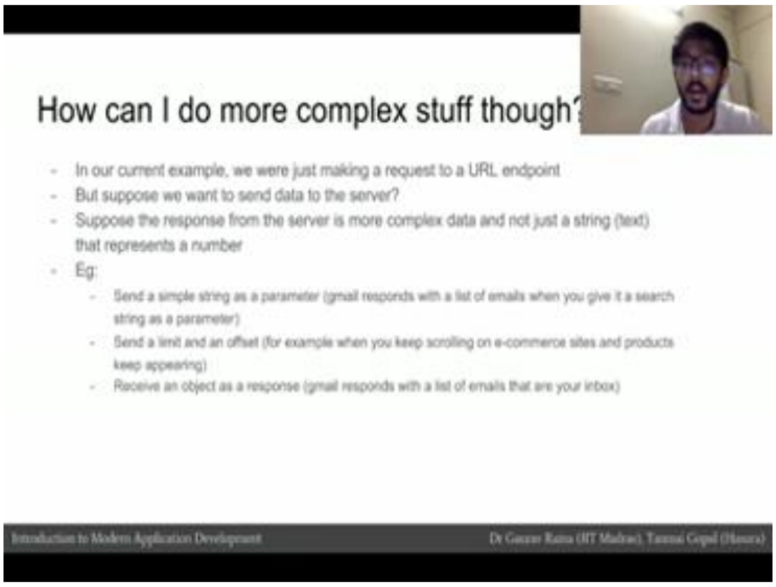
- Check it out on inspect element.
- When you load mail.google.com only the HTML and JS files are fetched. These don't contain any email data
- Then the JS on the browser makes a request to fetch emails
- These emails are then converted to HTML and inserted dyanmically on to the existing page
- This is known as client side rendering!

Introduction to Modern Application Development

Dr Gaurav Raina (IIT Madras), Tanmai Gopal (Hasura)

I will be chosen activity that we usually did on the server this action is called client side templating and this is how all of Gmail works. In fact, if you open up inspect element when Gmail is loading you see that when you load mail dot Google dot com only a blank page with a kind of loading of loading bar loads and the j s files load and the CSS files load and the javascript files have the javascript that when, executed make a request to the Gmail server and fetch the email content and you will actually, be able to see the request that are fetching emails and in the response, you see the email data been given and the javascript that is loaded converts the data that is sent back from the browser and converts that into HTML and that is how your inbox is actually rendered.

(Refer Slide Time: 15:16)



How can I do more complex stuff though?

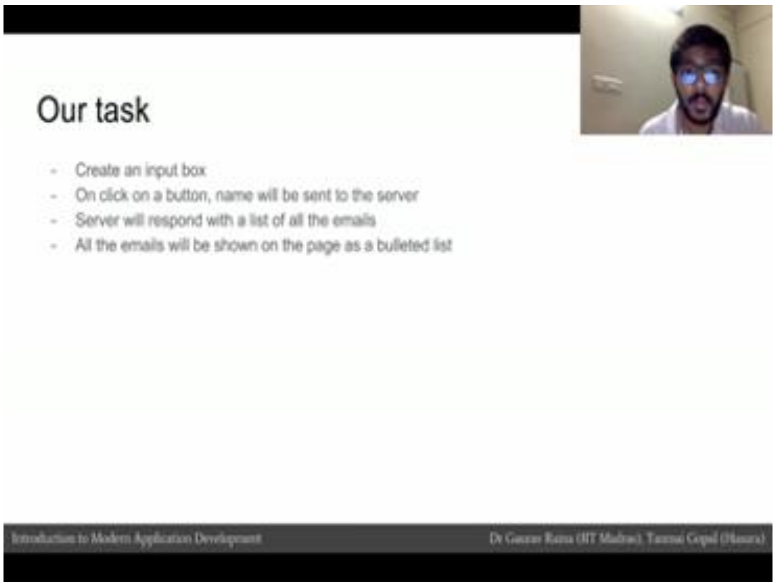
- In our current example, we were just making a request to a URL endpoint
- But suppose we want to send data to the server?
- Suppose the response from the server is more complex data and not just a string (text) that represents a number
- Eg:
    - Send a simple string as a parameter (gmail responds with a list of emails when you give it a search string as a parameter)
    - Send a limit and an offset (for example when you keep scrolling on e-commerce sites and products keep appearing)
    - Receive an object as a response (gmail responds with a list of emails that are your inbox)

Introduction to Modern Application Development   Dr Gaurav Raina (IIT Madras), Tanmai Gopal (Hasura)

If we think about these more complex use cases like Gmail it is not just about making a request to a URL end point sometimes we need to send data to the server. So, for example, in Gmail when you are searching through your email and you give it a search string or on some modern Ecommerce sites, when you keep scrolling and new products keep loading in this case we are not just making a request for getting new pieces of data we are making a request, but we are also sending a parameter.

So, we are sending some data for example, a query string or we are saying that we have already loaded ten products give us 10 to 20 and then give us products from 20 to 30. So, we are sending some kind of data to the server in all of these cases also that response, from the server is not just an integer or a string representation of an integer, which is what we looked at when we were using the counter example in this case the response is far more complicated for example, you have an object or an array of data items that are being sent back as a response.

(Refer Slide Time: 16:14)



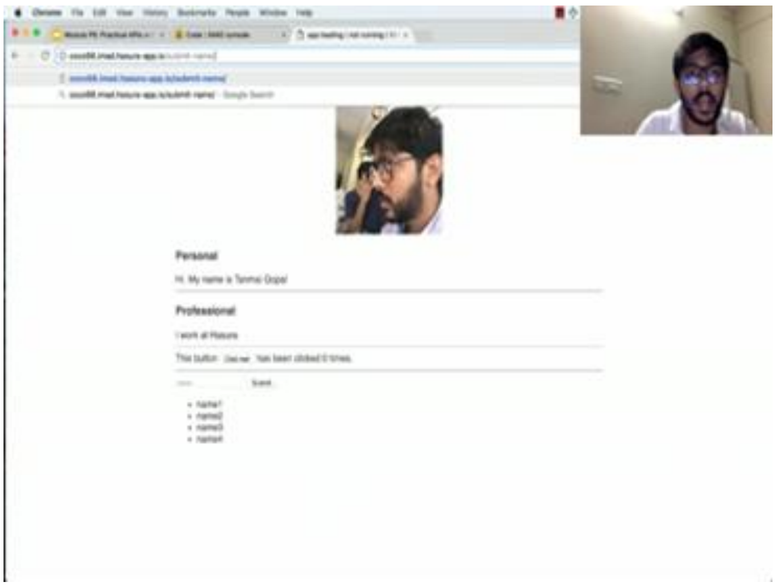
Our task

- Create an input box
- On click on a button, name will be sent to the server
- Server will respond with a list of all the emails
- All the emails will be shown on the page as a bulleted list

Introduction to Modern Application Development   Dr Gaurav Raina (IIT Madras), Tanmai Gopal (Hasura)

How do we approach these more complex scenarios? So, we will try to understand this by doing a small task in this small task we will add an input box on our page and when we click a button the data from the data will be taken from the input box, which will say be a name and it will be sent to the server this server will then respond, with all the names that have been submitted to the server. So, far and once we get these list of names we will show them on the page as a bulleted list. Let us see how this task is done.

(Refer Slide Time: 16:39)



Personal

Hi. My name is Tanmai Gopal

Professional

I work at Hasura

This button Click me! has been clicked 0 times.

Submit

- name1
- name2
- name3
- name4

I head to the I mad console now I want to change the index dot HTML to add just put a breakpoint here and let us add an input box let us say input type equal to text and let us say I d equal to name, Its input we want a place holders. Let us put a place holder that says name and we will see what the place holder text is and then, we will also have a little button which is a submit button and say it will say submit and once this is done we will want to see a list of all the names. This is kind of what we want the list to be like name 1, name 2, and name 3.

Let us save this let us checkout what this looks like. This is kind of what we want to see where, if I enter a name and I click on submit I want the first item to change to a s d a f a s t f and then, change to name 1, name 2, name 3. That is kind of what I would like to see right and let us just clean this up a little more. That we have more spacing here maybe we can change this to add perhaps have a horizontal line. So, we will put another as horizontal line here that will look little neater.

This is this is kind of what we want to implement right and we have the h t m l. Now let us write the javascript for it. I go back here I quickly change this to a horizontal line and let us go to our main dot j s file. So; now, we say submit name. Let us capture the name. So, we will say the name input is equal to document dot get element by I d and the name of the input box was name, and the value of the name is name input of value what we will need to do next is var submit button is equal to document dot get element by I d and we will call this submit button and submit dot on click will be a function unless write the code for this function.

This function should make a request to the server and send the name capture a list of names and render it as a list right. So, already we can see that 2, 3 things are missing the first thing is that, we need to have the submit button label attach let us go to index dot HTML and let us have a I d for this input button. Let us say I d is equal to submit button. Now, we can actually capture the submit button.

Let us again go over the making a request business a last and let us just say for now, we have a list of names is an example. Let us say var name is equal to name 1, name 2, name 3. So, here is a list of our names let us convert this into an HTML string. I will say var list is equal to that is (Refer Slide Time: 19:46) it is an empty string and I will iterate over these elements. I will say start with 0 go on till names dot length I plus plus and for each element in the list add this string to the list element. I will say l I plus names of i.

That is the name that we have in this iteration, and let us close the l I element. That is a list element and now once, we have this we need to insert that HTML into our unordered list. So, we have var u l is equal to document dot get element by I d name list. So, we will do u l dot inner HTML is equal to list. This should change the list that we have. Let us try to test this code out let us initially make sure that the list has nothing and let us make sure that we have the I d which we used the name list.

Now if I click on this button name list should have these 3 names; name 1, name 2, name 3, let us just add name 4 here as an exercise. This would mean that our rendering or templating if it is working then we will figure out later on how to make a request to the server. Let us just test if this is working we refresh, the page let us click on this button and see what happens and as soon as I clicked on this button you see that name 1, name 2, name 3, 4 got displayed here.

Now, we want to figure out how to make a request to the server, but before you make a request to the server the server should actually be listening to the request. Let us add an end point on the server. Let us say we add 1 called app dot get slash submit hyphen name, but slash in the beginning, slash submit hyphen name and if that happens, we will execute a particular function and this function will. So, what will these functions do we want our function to get the current name somehow.

So, get the name from the request object and let us say; we extracted this name somehow right and once we extract the name, value we want to concatenate it to our overall list of names. So, we will have let us say names dot push name. Once it is pushed let us somehow return that response. So, we need to do something like a rest dot send of name right and. Let us initialize the names. So, var names equal to this.

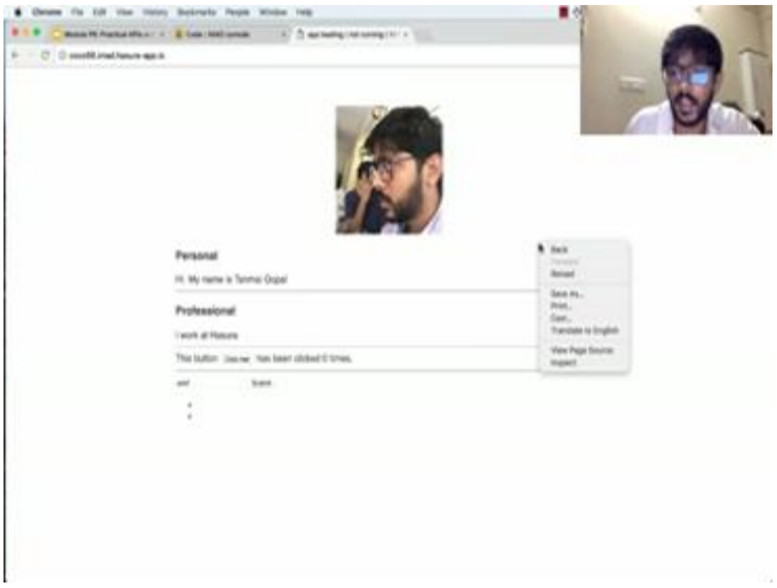
What is definitely not clear and we can clearly see that 2, 3 things are deeply missing here right. The first part is how are we going to extract the name from the request and the second part is how are we going to send because name is an array, now and this is a javascript object how can we send a javascript object or a javascript array as a string because we know that rest dot send you know, we can actually just send down bytes or string or a text value how can we send an array that that does not make sense.

Let us take this 1 step at a time the first in the view, we want to do is let us see if we can extract the name from request. Let us use the same technique that we used here right. Let us send the data as a part of the URL object. So, we can do slash name right. The URL request that will be making a slash submit name and then there will be a parameter and whatever this parameter is will be called the variable name. So, we will use the same technique as we used earlier and we will say this is equal to rec dot params dot name.

This is 1 part of the problem solved, the next thing that we want to do is convert this array into a string and the best way of converting javascript objects. In this case our in this case we had a very simple example; where we just have an array of strings, but you know what if we had an array of objects for example, if you look at if you look at this data structure its actually in object which contains; a more objects nested objects inside it.

These are complicated objects how do we convert these objects to string and this is where a format called JSON comes in and so, JSON stands for javascript object notation. So, javascript object notation is a way of converting javascript objects into strings and. So, what we can actually do is we can say JSON dot string fy and this will convert the array into a string. Let us put this in action and see what this looks like and let us go to slash submit hyphen name and let us enter a test name here.

(Refer Slide Time: 25:16)



Personal

Hi. My name is Tanmai Gopal

Professional

I work at Hasura

This button Click me has been clicked 0 times.

I am mentioning my name Tanmai. So, you can see Tanmai and every time I make a new request another element gets added to the array and If I change this to something else you can see this something else gets added here right. So, Ramu you see, Ramu is added here this is 1 way of getting the information as a part of the URL and then sending that information back as JSON. Let us look at another way of sending data and this is called a query parameter. So, query parameter if you remember from a week 1, was the bit of a URL that is usually after a question mark. Let us remove it from here and. Now, what we expect our request to be is something like this slash submit name question mark name equal to something right and this is the bit that we want.

Now, the question is how do we extract this part from the request object and express again gives us convenient way to extract this because this is called a query parameter string we can just do query dot name and the same thing will work. Let us quickly put this in action if you can see that we are getting an error and that error is coming from line number 6 on server dot j s which itself, is coming from line number ninety six in server dot j s and. It seems that this code is what is trying to be executed server 96 and; obviously, because 96 is using article the error seems to be coming from line number 46.

The first question that we have to ask is why is this getting executed and not this and the reason for that is because our URL match pattern comes after this the URL match pattern for article name is getting executed, and it is not able to find a article name that corresponds to submit name and the simple way to fix this problem is to remove this from here and put this here. This is something that was peculiar to express and the way express handles URL routing.

Now, let us try to save this and see what happens and. So, we see that its working we are getting Shyam, Shyam, Shyam, let us change this to ram we are getting ram and let us get something and. It is working. Now, we have a URL end point which takes a name and which returns a list of names. Let us add that code into our main dot j s.

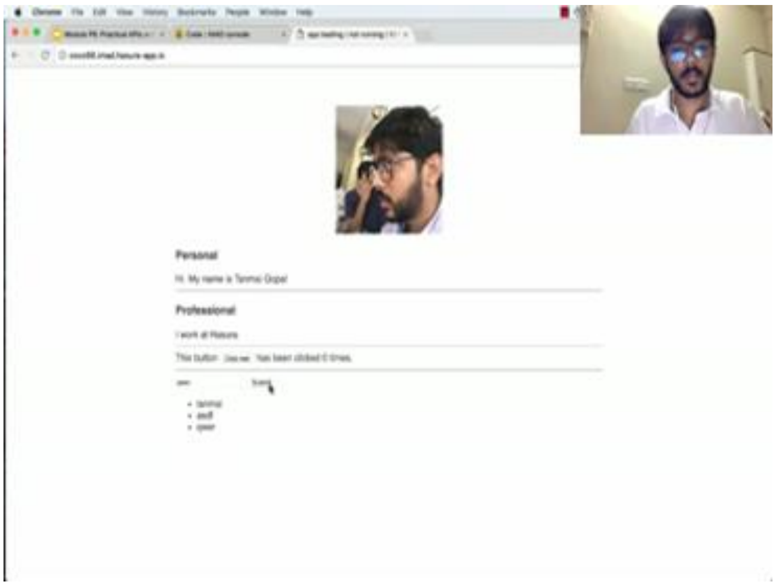
I am going to copy this code from here. What we will do is what we want to do is its requested successful this time instead of incrementing a counter, we want to do rendering our name list. Let us copy our code there. That is the bit that is rendering it and now what we want to do is change the place. The request is made. So, we will not make it to counter we will make it to submit name and we want to say name equal to and this value will come from the input box. So, we will do plus name here.

So, what is going to happen is that first we are going to select the input box we are going to extract the value from the input box and then whenever, the submit button is clicked will create a request object and we send the request to this URL once I send the request to this URL and we get a 200 response which is a successful response, we will get the list of names, which we have not changed yet. So, we will get the list of names and we will render the list of names.

Let us change this last bit here which instead of hard coding these list of names let us actually get that from the response and so, what we will do is we will say var name request dot response text, but again if you look at this names it will actually be a string it would not be a javascript array. Let us do the reverse of what will do in the server and we will do JSON dot parts not JSON dot string, if i. So, we will convert it from string back into an object or in this case an array right. So, we converted that into an array and then our code should continue to work.

Let us save this and let us head to the homepage let us enter something here. Let us say alright. So, we can see that nothing is getting rendered here. Let us try to.

(Refer Slide Time: 30:01)



Personal

Hi. My name is Tanmai Gopal

Professional

I work at Hasura

This button Click me has been clicked 0 times.

Submit

- tanmai
- asdf
- qwer

Find out what the bug is. The way I tried to find where the bug is I go to inspect element which is opens a developer tools window I over the course of the video I have been using the word inspect element, and open inspect element when actually this window is called developer tools. The right word is to open developer tools and I frequently call that inspect element that is the wrong name.

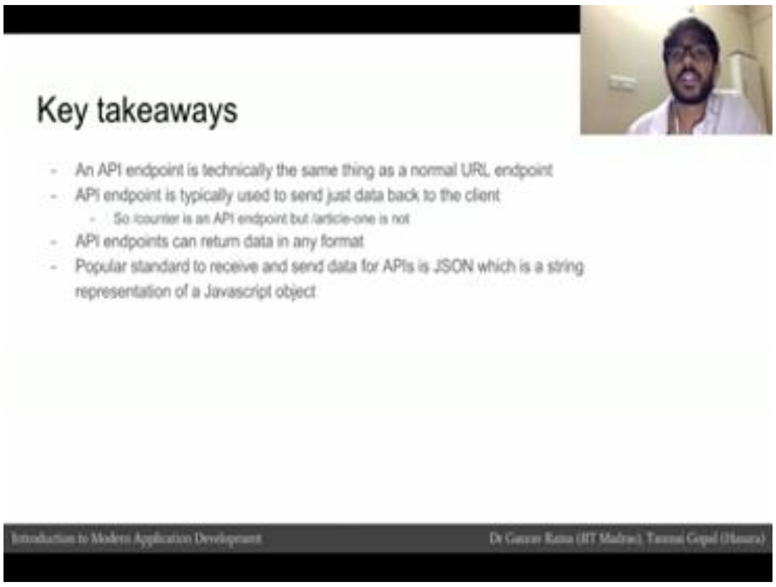
Now if we look at the request that is being sent it seems to be sending a request to a there is no name value that is being sent and which is why the response, that we getting are 2 empty strings because it is receiving an empty string. Let us try to find out why we are sending an empty string. The way I debugged basis typically go to the sources tab and I try to find out why we are not able to capture the right value.

Let us just try to execute this code on the console. I am going to take the same code and I am going to copy it on the console and I will see if. Name input is in fact, the input box; let us then try to capture this value. This is also correct name is. In fact, a d f f which is exactly what I have written here so far. So, good and then when I am clicking on click something seems to go wrong where name is not getting rendered here. It seems that when the code comes to this portion at the value of name is not available anymore.

So, and the reason why we think this is happening is because the value of name was extracted once here we should be extracting the value of name on click because by the time the function by the time our code gets here and starts executing this code, the value of name, is the value of name that it was when the document first loaded it is not refreshing the value of name according to what is being typed and; that means, that are error is that we need to move this portion of the code inside the click function.

Let us try doing that right. This means that when I click on the button submit that is when, I try to extract the value from the input element I do not try to extract it in the very beginning when the document is loaded I extract it only when I click on the button. Let us save this let us try to enter a name here and you can see the name has appeared. Let us try to enter another name here a s t f and you can see that a s t f it takes a little bit of time to appear because, it is making the request to the server and the response is coming back.

(Refer Slide Time: 32:54)



Key takeaways

- An API endpoint is technically the same thing as a normal URL endpoint
- API endpoint is typically used to send just data back to the client
    - So /counter is an API endpoint but /article-one is not
- API endpoints can return data in any format
- Popular standard to receive and send data for APIs is JSON which is a string representation of a Javascript object

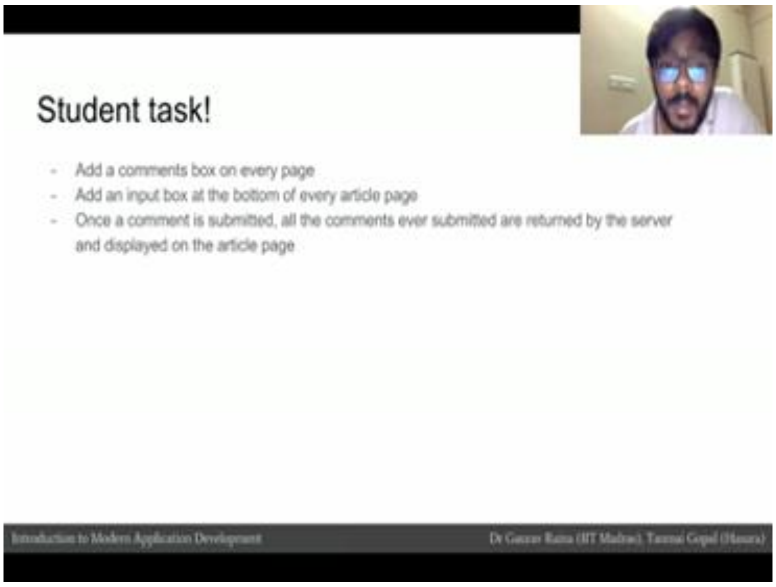
Introduction to Modern Application Development

Dr Gaurav Raina (IIT Madras), Tanmai Gopal (Hasura)

The key takeaways in the module are the following an API end point is technically the same thing as a normal URL end point it is just that a URL end point typically is used to all kinds of content like HTML javascript CSS or images, but then API end point is typically used only for exchanging data. So, you can request some data and you can response that contains data.

An API end point can also return data in any format. It can written data in a text in a string, but for more complex pieces of data and for more complex data structures, a very common format is called JSON and a JSON is a JSON format is basically the string representation of a javascript object.

(Refer Slide Time: 33:31)



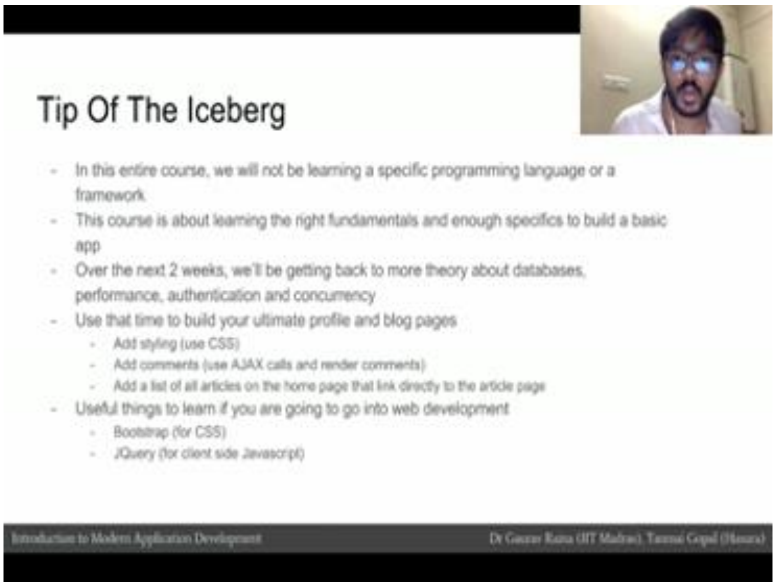
Student task!

- Add a comments box on every page
- Add an input box at the bottom of every article page
- Once a comment is submitted, all the comments ever submitted are returned by the server and displayed on the article page

Introduction to Modern Application Development   Dr Gaurav Raina (IIT Madras), Tanmai Gopal (Hasura)

The task for you guys is to try to add comment boxes on every article page have an input box let people type a comment into the input box and when something is typed display a list of all the comments that have ever been typed. So, far and display it on the bottom of the article page.

(Refer Slide Time: 33:53)



Tip Of The Iceberg

- In this entire course, we will not be learning a specific programming language or a framework
- This course is about learning the right fundamentals and enough specifics to build a basic app
- Over the next 2 weeks, we'll be getting back to more theory about databases, performance, authentication and concurrency
- Use that time to build your ultimate profile and blog pages
  - Add styling (use CSS)
  - Add comments (use AJAX calls and render comments)
  - Add a list of all articles on the home page that link directly to the article page
- Useful things to learn if you are going to go into web development
  - Bootstrap (for CSS)
  - JQuery (for client side Javascript)

Introduction to Modern Application Development   Dr Gaurav Raina (IIT Madras), Tanmai Gopal (Hasura)

Over the last few practical modules we have just touched the tip of the iceberg in this entire course we are not going to be learning any specific programming language of framework this course is only about learning the right fundamentals, and just enough specifics to build a basic app.

So, over the next 2 weeks when we return to more theory about databases performance security concurrency use that time to build, your ultimate profile and webapp. So, add a lot of styling use add comments add a list of articles on the homepage that link directly to the article page and add other features, that do whatever you want there is limit to what you should or you should not do the 2 important things for you to learn.

If you are going to get into web development more seriously especially on the front end side of things is to use bootstrap or bootstrap is a framework, for CSS and j query which is a framework for writing javascript. It makes writing javascript more easy and both these links you can see in the slides if you to go the slides.