VIDEO CHAT APP-UI

What is our GOAL for this MODULE?

The goal of this module is to learn to write HTML code for Video Chat App, learn to create a script to handle responsiveness.

What did we ACHIEVE in the class TODAY?

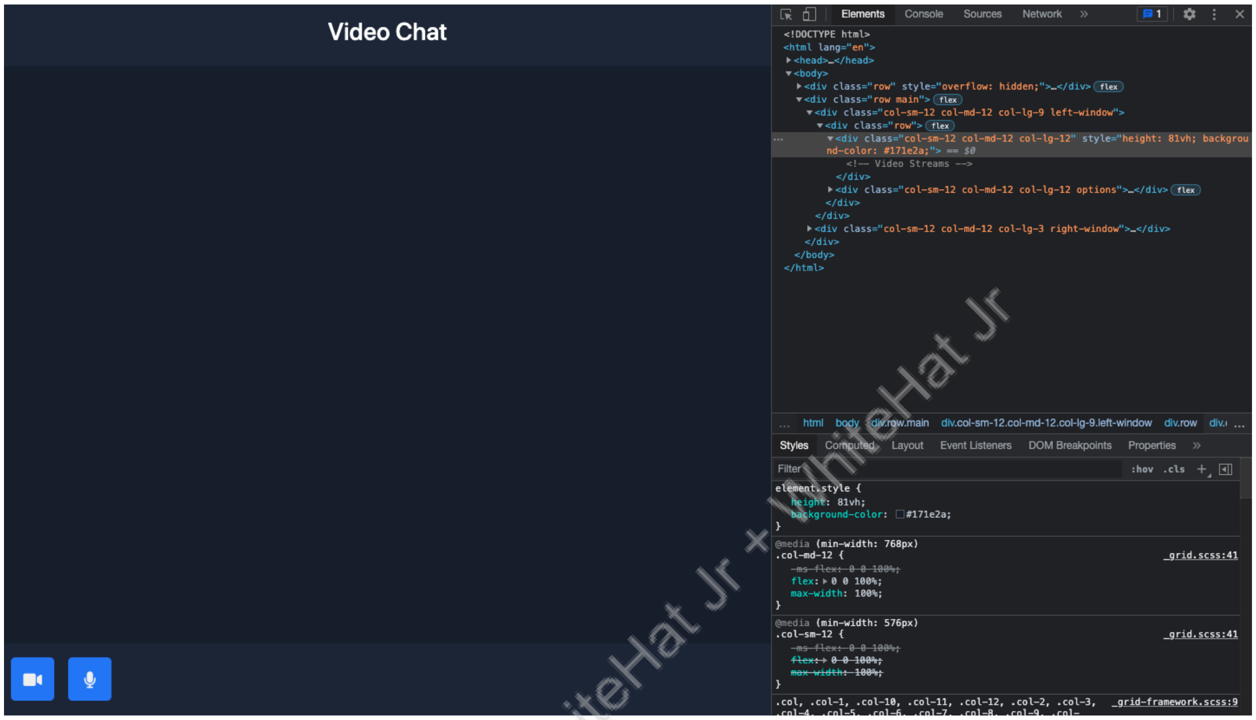
- Worked on the Bootstrap library.
- Learned to add relevant HTML and CSS for responsiveness.
- Learned to add jQuery code to make the chat button functional.
- Learned to ddd jQuery code to make the back button functional.

Which CONCEPTS/ CODING BLOCKS did we cover today?

- Chat and back button functionality.
- Video chat App-UI:
- Bootstrap

How did we DO the activities?

1. We learned how to make the chat option available for mobile view of the app.

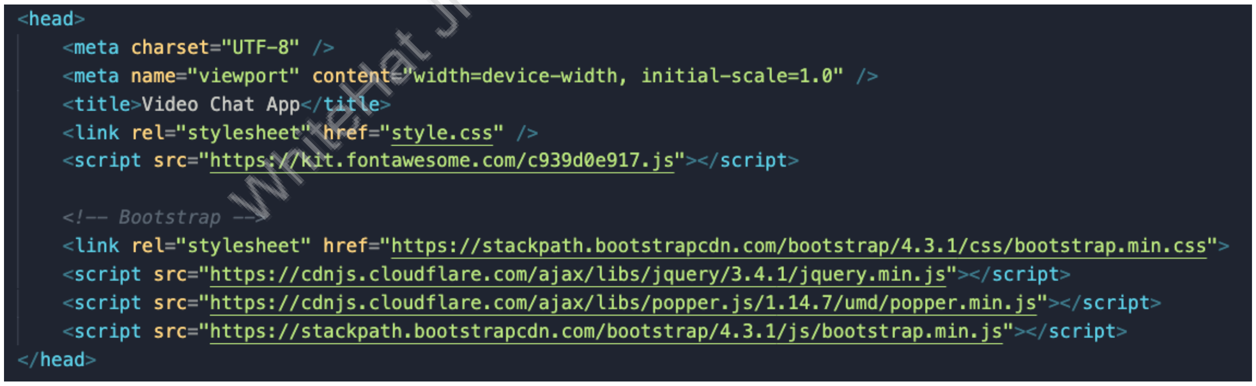


```
<!DOCTYPE html>
<html lang="en">
 ▶<head>…</head>
 ▼<body>
   ▶<div class="row" style="overflow: hidden;">…</div> flex
   ▼<div class="row main"> flex
     ▼<div class="col-sm-12 col-md-12 col-lg-9 left-window">
       ▼<div class="row"> flex
         ▼<div class="col-sm-12 col-md-12 col-lg-12" style="height: 81vh; backgrou
         nd-color: #171e2a;"> == $0
             <!-- Video Streams -->
           </div>
         ▶<div class="col-sm-12 col-md-12 col-lg-12 options">…</div> flex
         </div>
       </div>
     ▶<div class="col-sm-12 col-md-12 col-lg-3 right-window">…</div>
     </div>
   </body>
</html>
```

```
... html body div.row.main div.col-sm-12.col-md-12.col-lg-9.left-window div.row div.( ...
```

```
element.style {
   height: 81vh;
   background-color: #171e2a;
}
@media (min-width: 768px)
.col-md-12 {                                                    _grid.scss:41
   ~~-ms-flex: 0 0 100%;~~
   flex: ▶ 0 0 100%;
   max-width: 100%;
}
@media (min-width: 576px)
.col-sm-12 {                                                    _grid.scss:41
   ~~-ms-flex: 0 0 100%;~~
   ~~flex: ▶ 0 0 100%;~~
   ~~max-width: 100%;~~
}
.col, .col-1, .col-10, .col-11, .col-12, .col-2, .col-3,   _grid-framework.scss:9
.col-4, .col-5, .col-6, .col-7, .col-8, .col-9, .col-
```

2. Import .css file



```
<head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Video Chat App</title>
    <link rel="stylesheet" href="style.css" />
    <script src="https://kit.fontawesome.com/c939d0e917.js"></script>

    <!-- Bootstrap -->
    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">
    <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"></script>
    <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"></script>
</head>
```

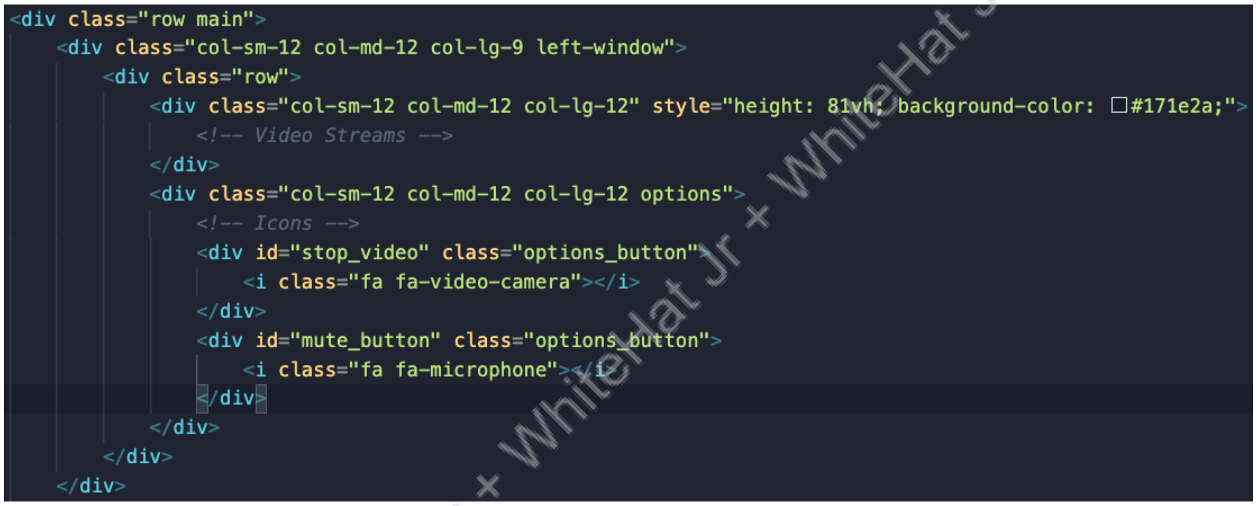
3. Bootstrap is a famous styling library used to make websites responsive. This means that it enables a website to be designed in a way that it looks fine in all kinds of displays, such as in desktop, tablets as well as mobile phones.

Bootstrap follows a box model, and works in ***rows*** and ***columns***. This means that everything that our page consists of is made up of ***rows*** and ***columns***.

Always remember while working with bootstrap is that the content should always be inside a ***column*** instead of directly being inside a ***row.***

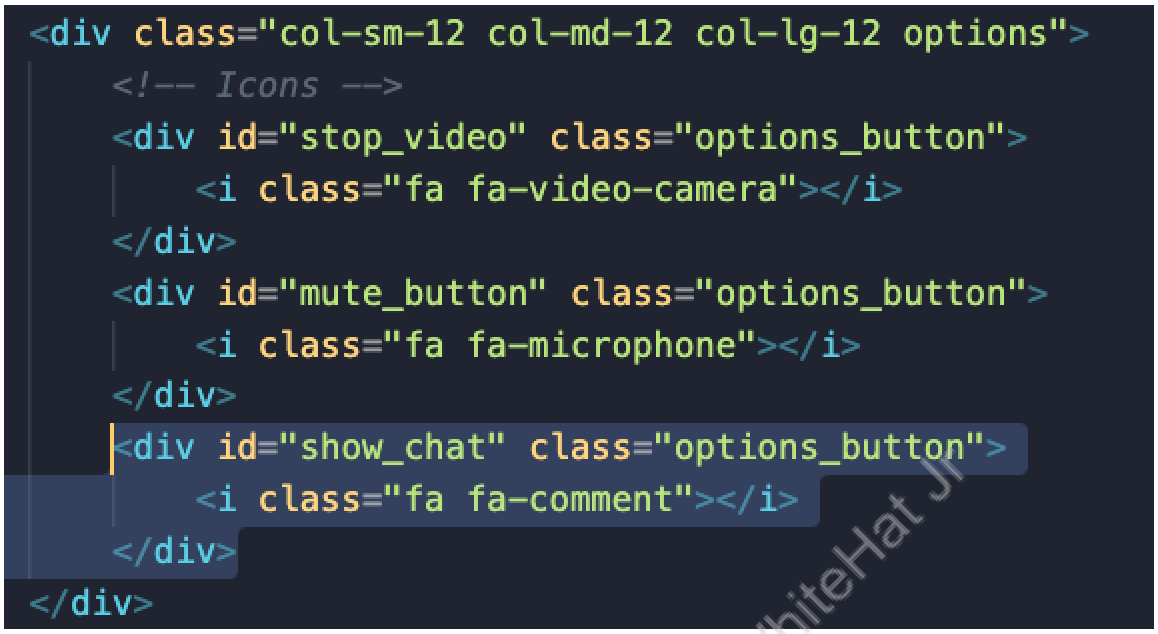
In bootstrap, a container can be divided into 12 different sections in terms of width.

- ***col*** defines a bootstrap column.
- ***sm*** defines column's width in small screen (mobile)
- ***md*** defines column's width in medium screen (tablet)
- ***lg*** defines column's width in large screen (desktop or laptop)
- ***text-center*** simply means to have all the text in the center of this column.
- ***p-3*** is for padding. The number **3** here could have been anything from **1-5**.



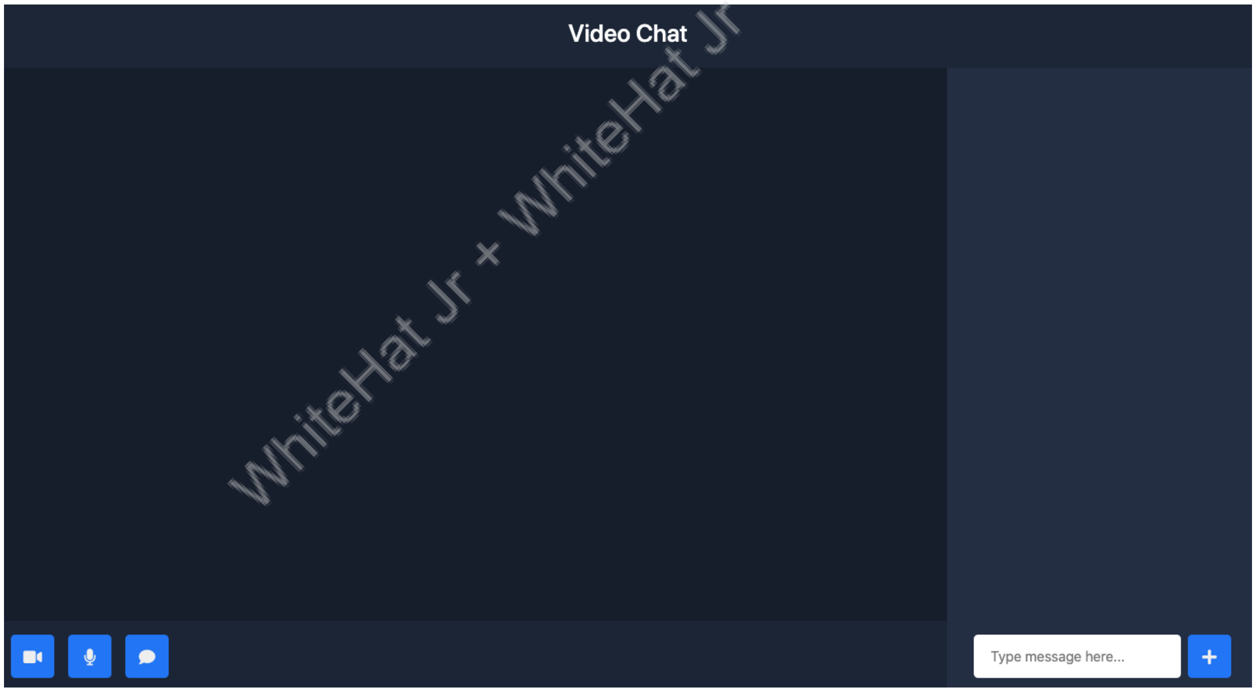
```
<div class="row main">
    <div class="col-sm-12 col-md-12 col-lg-9 left-window">
        <div class="row">
            <div class="col-sm-12 col-md-12 col-lg-12" style="height: 81vh; background-color: ☐#171e2a;">
                <!-- Video Streams -->
            </div>
            <div class="col-sm-12 col-md-12 col-lg-12 options">
                <!-- Icons -->
                <div id="stop_video" class="options_button">
                    <i class="fa fa-video-camera"></i>
                </div>
                <div id="mute_button" class="options_button">
                    <i class="fa fa-microphone"></i>
                </div>
            </div>
        </div>
    </div>
```

4. Add the icon for messages next to the other icons. Class for the button would be "option_button" and id "show_chat"



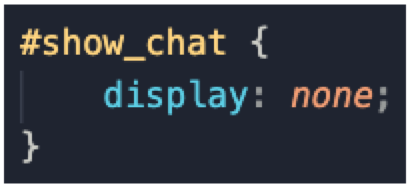
```
<div class="col-sm-12 col-md-12 col-lg-12 options">
    <!-- Icons -->
    <div id="stop_video" class="options_button">
        <i class="fa fa-video-camera"></i>
    </div>
    <div id="mute_button" class="options_button">
        <i class="fa fa-microphone"></i>
    </div>
    <div id="show_chat" class="options_button">
        <i class="fa fa-comment"></i>
    </div>
</div>
```

5. Till now it looks like this:



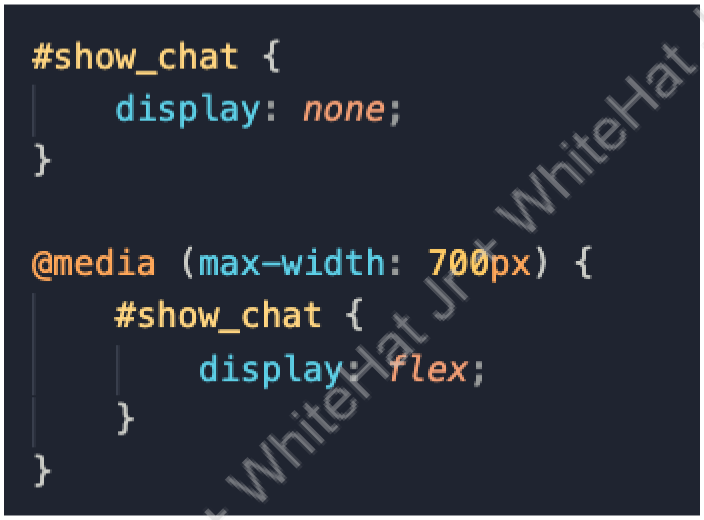
6. But to make the icon visible only in mobile view,for that we need to do changes:

   - First, change the value of property ***display*** to ***none*** to make the message option invisible on the desktop.



```
#show_chat {
    display: none;
}
```

- Second, CSS has one special feature called ***media queries***, in which we can write different CSS for different screen sizes! In the code below, media query is mentioned by using ***@media*** keyword, and defined it's condition that the ***max-width*** for it to work shall be ***700px***.

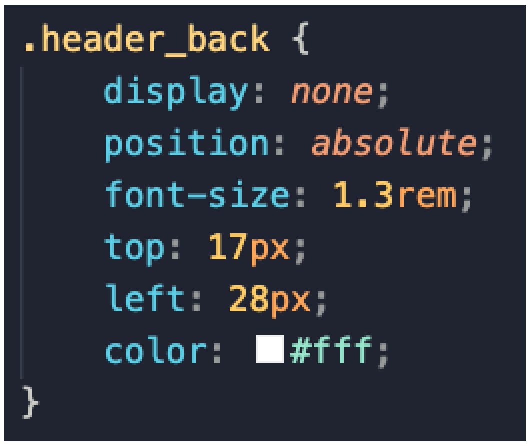


```
#show_chat {
    display: none;
}

@media (max-width: 700px) {
    #show_chat {
        display: flex;
    }
}
```

Note: media queries should ***always*** go at the end of the CSS file, or your styles may not reflect properly.

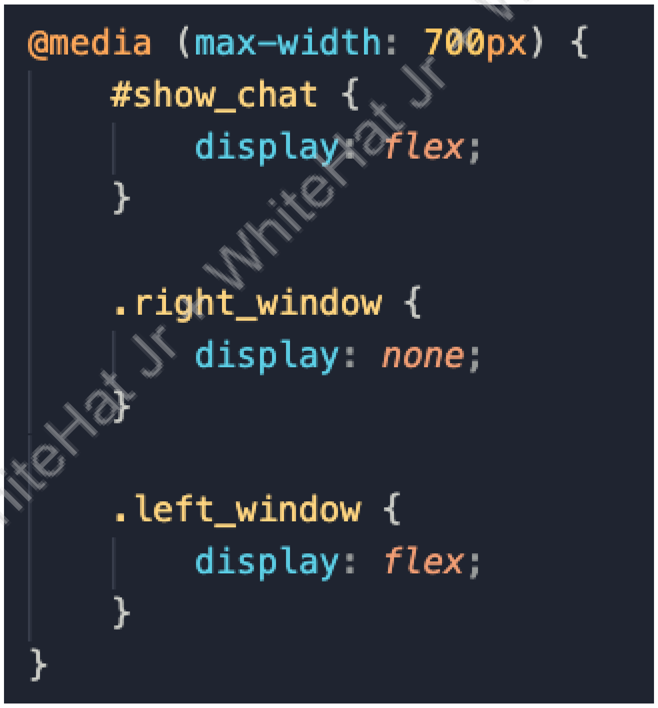
7. Add a **back button** to navigate back to the main screen after using the chat option



```css
.header_back {
    display: none;
    position: absolute;
    font-size: 1.3rem;
    top: 17px;
    left: 28px;
    color: #fff;
}
```

8. The CSS related to the above created class "header_back".

9. Add styles for our left-window and right-window in our media query



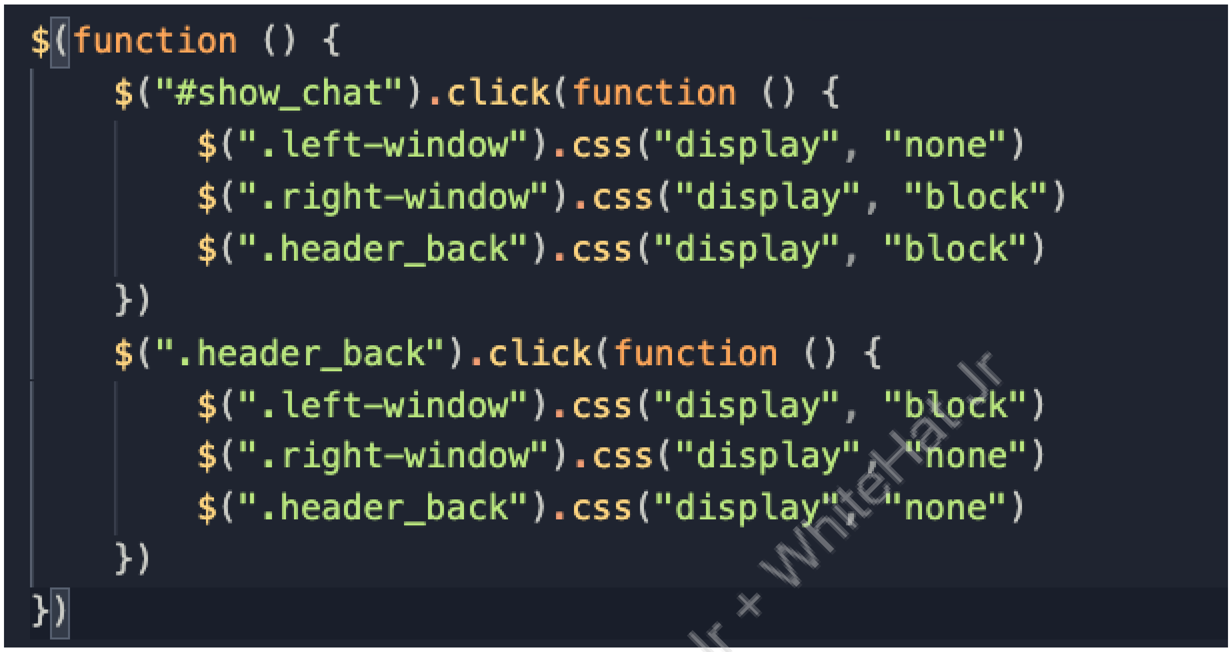
```css
@media (max-width: 700px) {
    #show_chat {
        display: flex;
    }

    .right_window {
        display: none;
    }

    .left_window {
        display: flex;
    }
}
```

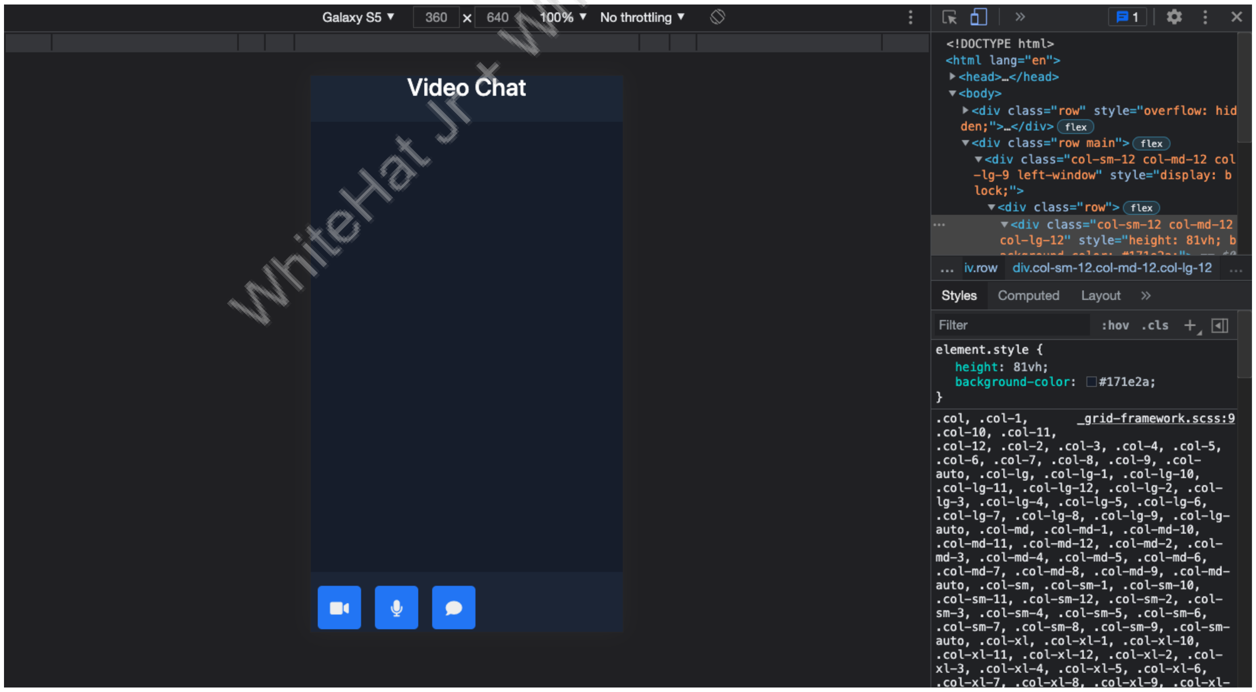
10. To make the Chat button and Back button functional using jQuery:

- Create a new file **script.js** and import in **index.html**.
- To create event handlers, create a $ function. Event Handlers must be mentioned in the $ function in jQuery.
- First, create the show_chat event handler. In this the left_window should be displayed and right_window and header_back must be displayed.
- Now create event handler for header_back:



```
$(function () {
    $("#show_chat").click(function () {
        $(".left-window").css("display", "none")
        $(".right-window").css("display", "block")
        $(".header_back").css("display", "block")
    })
    $(".header_back").click(function () {
        $(".left-window").css("display", "block")
        $(".right-window").css("display", "none")
        $(".header_back").css("display", "none")
    })
})
```

11. Now our output look like this :



```
<!DOCTYPE html>
<html lang="en">
 <head>…</head>
 <body>
   <div class="row" style="overflow: hidden;">…</div> flex
   <div class="row main"> flex
     <div class="col-sm-12 col-md-12 col-lg-9 left-window" style="display: block;">
       <div class="row"> flex
         <div class="col-sm-12 col-md-12 col-lg-12" style="height: 81vh; b
```

… iv.row div.col-sm-12.col-md-12.col-lg-12 …

Styles Computed Layout

```
element.style {
   height: 81vh;
   background-color: #171e2a;
}
.col, .col-1,          _grid-framework.scss:9
.col-10, .col-11,
.col-12, .col-2, .col-3, .col-4, .col-5,
.col-6, .col-7, .col-8, .col-9, .col-
auto, .col-lg, .col-lg-1, .col-lg-10,
.col-lg-11, .col-lg-12, .col-lg-2, .col-
lg-3, .col-lg-4, .col-lg-5, .col-lg-6,
.col-lg-7, .col-lg-8, .col-lg-9, .col-lg-
auto, .col-md, .col-md-1, .col-md-10,
.col-md-11, .col-md-12, .col-md-2, .col-
md-3, .col-md-4, .col-md-5, .col-md-6,
.col-md-7, .col-md-8, .col-md-9, .col-md-
auto, .col-sm, .col-sm-1, .col-sm-10,
.col-sm-11, .col-sm-12, .col-sm-2, .col-
sm-3, .col-sm-4, .col-sm-5, .col-sm-6,
.col-sm-7, .col-sm-8, .col-sm-9, .col-sm-
auto, .col-xl, .col-xl-1, .col-xl-10,
.col-xl-11, .col-xl-12, .col-xl-2, .col-
xl-3, .col-xl-4, .col-xl-5, .col-xl-6,
.col-xl-7, .col-xl-8, .col-xl-9, .col-xl-
```

What's next?

In the next class, you will be creating more database queries, to add both players' details. Read & write game state and player count.

EXTEND YOUR KNOWLEDGE:

Watch this video to learn more about creating forms using p5.DOM.js:
https://youtu.be/lAtoaRz78l4