Implementation in .js

Currently, the lab application can compute the Hamming code for four information bits with no parity bit. Modify the application to select with/without parity bits and between 4 and 8 information bits.

Client-side (existing comments into the code where the changes should be done):

- Modify the **app.js** to calculate the parity bit and the fourth control bit.
- Modify the **index.html** to select with/without parity bit.

Server-side (existing comments into the code where the changes should be done):

- Modify **hamming.js** to calculate the Z vector according to the vector length sent by the client (4 bits info without parity bit is the current functional example).
- Add comments into the console with the calculated Z vector.

Code snippets:

```
var c4=this.parity(parseInt(bits[1].data)+parseInt(bits[2].data)+parseInt(bits[3].data)); // The control bit from the position 4 is calculated
var c2=this.parity(parseInt(bits[0].data)+parseInt(bits[2].data)+parseInt(bits[3].data)); // The control bit from the position 2 is calculated
var c1=this.parity(parseInt(bits[0].data)+parseInt(bits[1].data)+parseInt(bits[3].data)); // The control bit from the position 1 is calculated
// var C0 = this. ...parity bit

var c0=this.parity(parseInt(c1+c2+bits[0].data)+c4+parseInt(bits[1].data)+parseInt(bits[2].data)+parseInt(bits[3].data)); //c0 – parity bit
```

4 info bits

Client

```
…
bits[0].data – first info bit (a3)
bits[1].data – second info bit (a5)
bits[2].data – third info bit (a6)
bits[3].data – fourth info bit (a7)
…
```

8 info bits

```
…
bits[0].data – first info bit (a3)
bits[1].data – second info bit (a5)
bits[2].data – third info bit (a6)
bits[3].data – fourth info bit (a7)
bits[4].data – fifth info bit (a9)
bits[5].data – sixth info bit (a10)
bits[6].data – seventh info bit (a11)
bits[7].data – eight info bit (a12)
…
```

```
…
var c8=this.parity(parseInt(bits[5].data)+parseInt(bits[6].data)+parseInt(bits[7].data)+parseInt(bits[8].data));
```

Server

Received vector (without parity bit) bits[0]	Received vector (with parity bit) bits[0] bits[1]	Received vector (without parity bit) bits[0]	Received vector (with parity bit) bits[0] bits[1]

```
bits[1]
bits[2]
bits[3]
bits[4]
bits[5]
bits[6]
```

```
bits[2]
bits[3]
bits[4]
bits[5]
bits[6]
bits[7]
```

```
bits[1]
bits[2]
bits[3]
bits[4]
bits[5]
bits[6]
bits[7]
bits[8]
bits[9]
bits[10]
bits[11]
```

```
bits[2]
bits[3]
bits[4]
bits[5]
bits[6]
bits[7]
bits[8]
bits[9]
bits[10]
bits[11]
bits[12]
```

```
// ...
var z0=parity(bits[0] + bits[1] + bits[2] + bits[3] + bits[4] + bits[5] + bits[6] + bits[7]);
// ...
if (z0 ==0 && …)
```