DATA COMMUNICATION

ASSIGNMENT

Group -1

Implementation of CSMA/CA.

Team members

Aravind Shreyas – 1MS18CS025
Dheeraj Bhat – 1MS18CS040
Divya – 1MS18CS043
Gaurav V – 1MS18CS046

RAMAIAH
Institute of
Technology

Certificate

This report is submitted for the evaluation of Assignment component for the subject "DATA COMMUNICTION" with the subject code CS44 during the term Jan-May 2020.

Submitted by-

Aravind Shreyas – 1MS18CS025
Dheeraj Bhat – 1MS18CS040
Divya – 1MS18CS043
Gaurav V – 1MS18CS046

Signature of Faculty

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

RAMAIAH INSTITUTE OF TECHNOLOGY
(Autonomous Institute, Affiliated to VTU)
BANGALORE-560054
www.msrit.edu
Jan-May 2020

TABLE OF CONTENTS

ACKNOWLEDGEMENT

I express my sincere gratitude to Asst. Prof. Mamtha Jadhav V, Dept. of Computer Science and Engineering, MSRIT, for her stimulating guidance, continuous encouragement and supervision throughout the course of present work.

PROBLEM STATEMENT

Write a C/C++/Java/Web program to implement CSMA/CA. Consider at least 6 stations that are sharing a communication channel. These stations have to sense the channel before transmitting. If the channel is free, it should wait for IFS time and again check the channel if it is free, calculate R wait for R slots and sense the channel and if it is free to send data otherwise it should use an exponential back-off timer and try for retransmission later.

INTRODUCTION

What is CSMA/CA?

CSMA is a basic method that controls the communication of multiple participants on a shared and decentralized transmission medium. CSMA/CA is mainly used in wireless networks.

Components of CSMA/CA include:

1. Carrier Sense (CS): The initial idea is that participants may only send data over the network if the transmission medium is free. The carrier status detection checks the channel any time, and data is not sent until it is available.

2. Multiple Access (MA): Several stations share a transmission medium. It is crucial for functioning communication that all of them adhere to a binding protocol.

3. Collision Avoidance (CA): A complex schedule tries to ensure that two or more participants do not start transmission at the same time to avoid collisions. If overlapping does occur, this will be detected and the transmission will be tried again.

The most important rule, which you'll know from communication situations when there are several participants is only one person may send their information at once. Else, "collisions" of data packets is said to occur.

CSMA/CA tries to reduce the frequency of these collisions and provide a plan at the same time on how to proceed if a collision does occur.

How does CSMA/CA Work?

The basic idea behind CSMA/CA is the "Listen Before Talk" (LBT) principle.

Within CSMA/CA, the distributed coordination function (DCF) controls the time a station waits before initiating transmission in a free medium. DCF also assigns certain time slots to network participants for further actions, creating a binding time structure. This procedure is the focus of collision avoidance: a complex time structure that makes it possible to avoid collisions. DCF takes various intervals into account when creating the time structure.

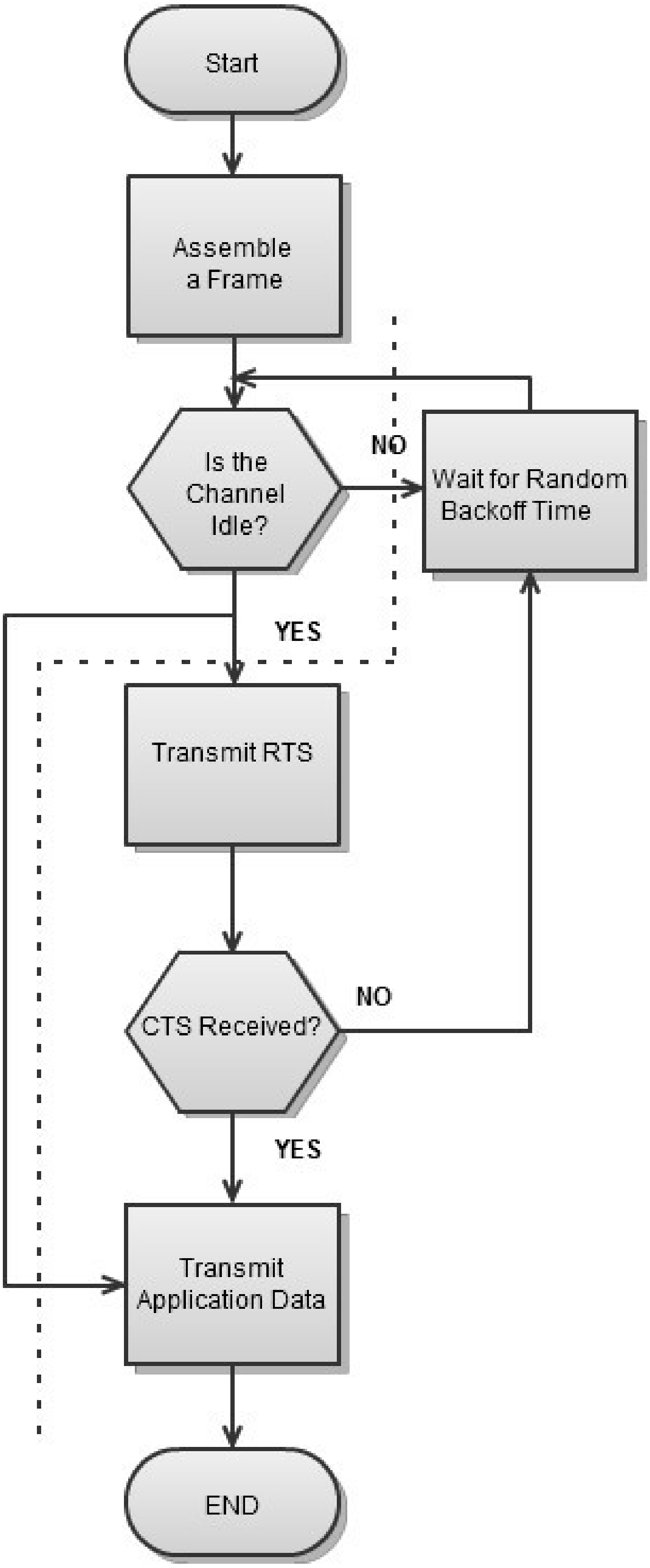
DCF interframe space (DIFS): In the first step, participants must monitor the network for the duration of the DIFS to determine whether it's currently free. For CSMA/CA, this means that no other station within range is sending out a transmission at the same time. The DIFS results from the SIFS almost double the slot time, which is between 28 and 50 µs long.

Contention window: If participants determine that the channel is free, they wait a random amount of time before they start sending. This duration corresponds to the contention window. This time window doubles with each collision and corresponds to the binary exponential back off (BEB) that is familiar from CSMA/CD.

Short interframe space (SIFS): After sending the data packet, the recipient node sends a notification – if the RTS/CTS procedure is also utilized. However, this station also waits for a fixed time before sending. SIFS is the time it takes to process a data package. The duration depends on the IEEE-802.11 standard and is between 10 µs and 16 µs.

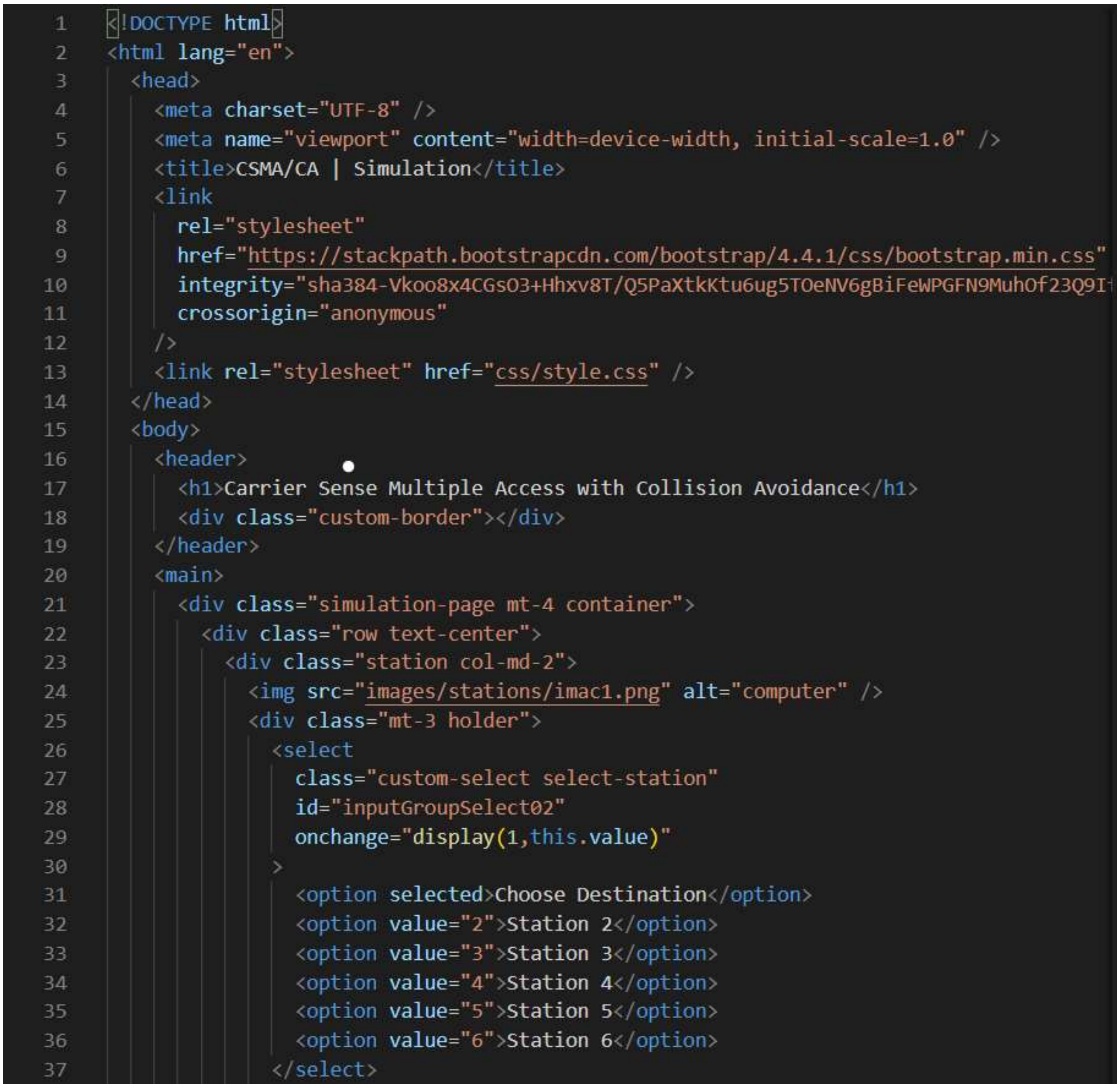
The frames "Request to Send" (RTS) and "Clear to Send" (CTS) are part of the optional extension CSMA/CA RTS/CTS. This procedure is upstream of the actual data transmission. If a participant determines that the transmission medium is free, the device first sends an RTS frame to the participant that is to receive the data. With this, the output computer makes it clear that it wants to start transmission and will occupy the transmission medium for a certain time.

A step by step diagrammatical representation of the working of CSMA/CA

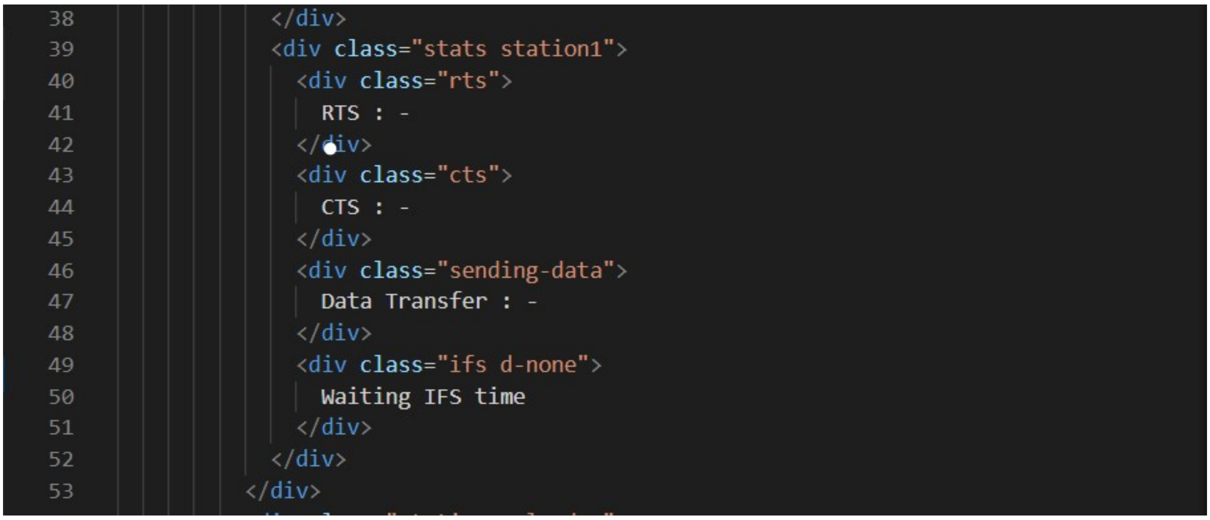


COMPLETE CODE

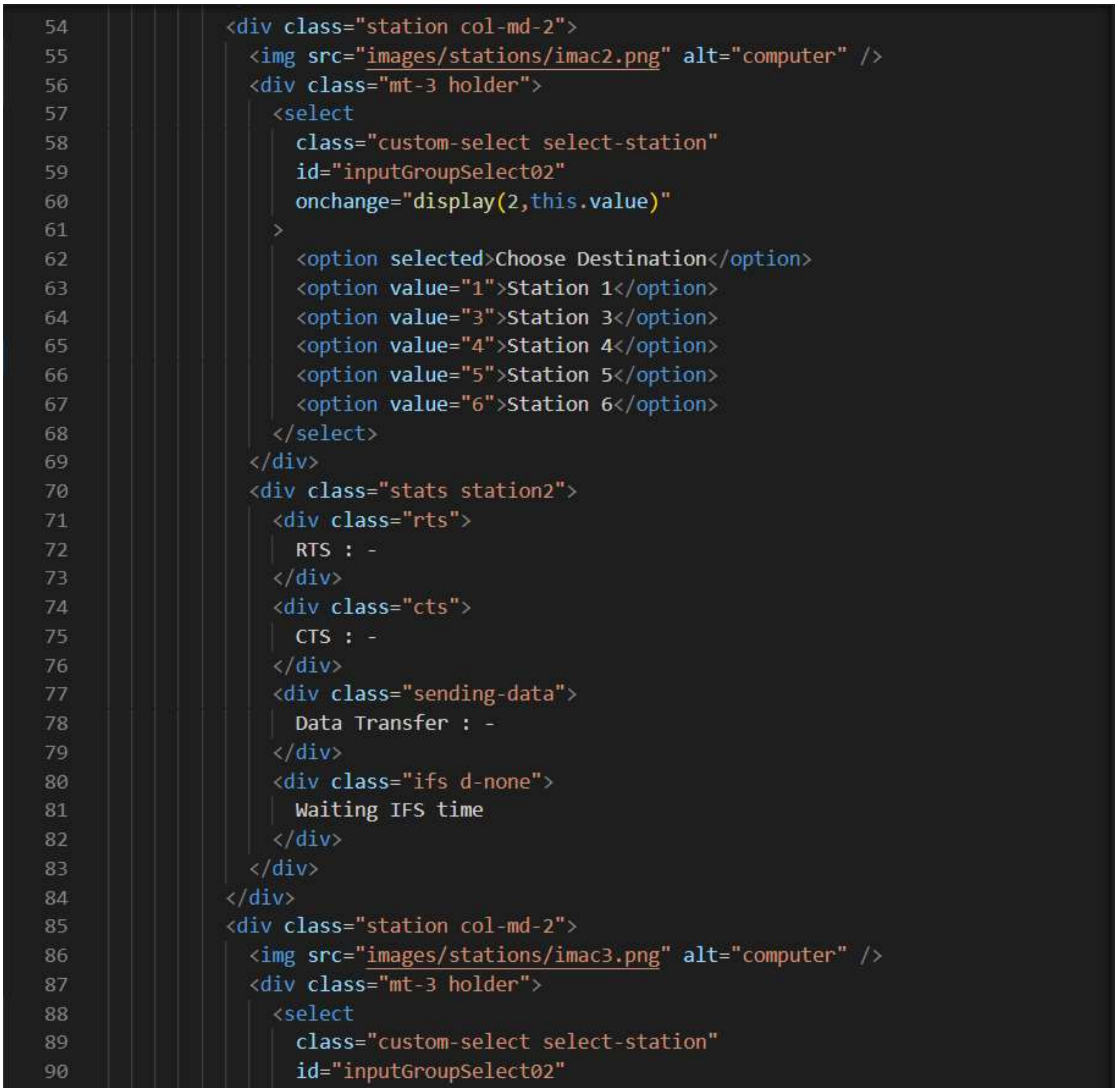
HTML File:-



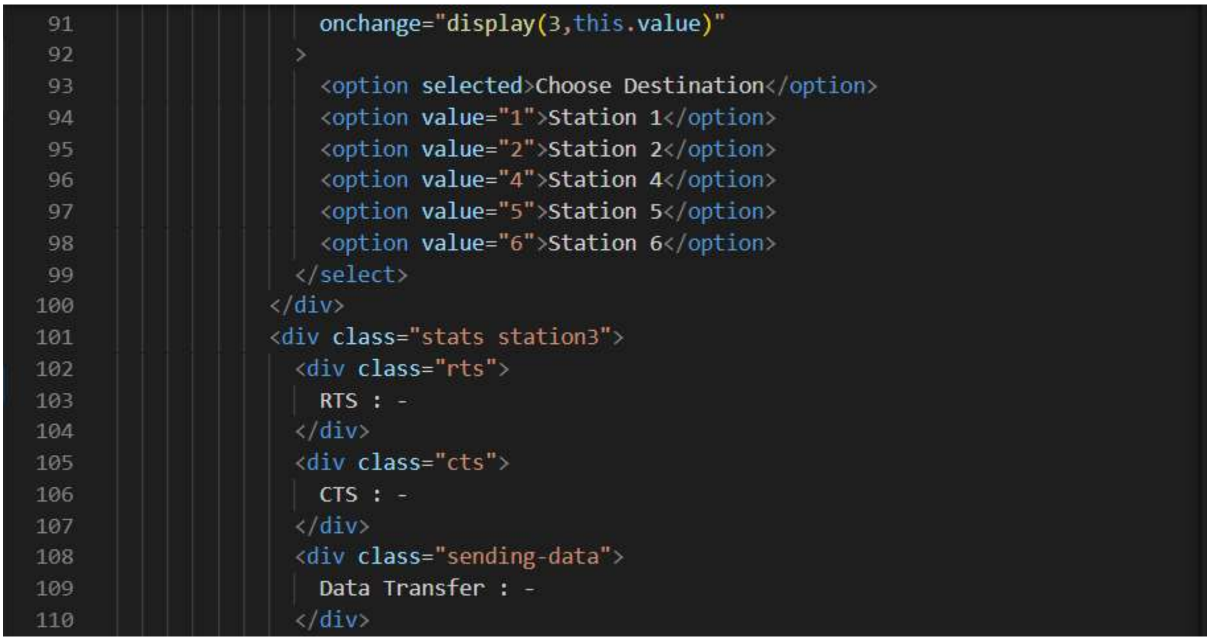
```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>CSMA/CA | Simulation</title>
    <link
      rel="stylesheet"
      href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css"
      integrity="sha384-Vkoo8x4CGsO3+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9I
      crossorigin="anonymous"
    />
    <link rel="stylesheet" href="css/style.css" />
  </head>
  <body>
    <header>
      <h1>Carrier Sense Multiple Access with Collision Avoidance</h1>
      <div class="custom-border"></div>
    </header>
    <main>
      <div class="simulation-page mt-4 container">
        <div class="row text-center">
          <div class="station col-md-2">
            <img src="images/stations/imac1.png" alt="computer" />
            <div class="mt-3 holder">
              <select
                class="custom-select select-station"
                id="inputGroupSelect02"
                onchange="display(1,this.value)"
              >
                <option selected>Choose Destination</option>
                <option value="2">Station 2</option>
                <option value="3">Station 3</option>
                <option value="4">Station 4</option>
                <option value="5">Station 5</option>
                <option value="6">Station 6</option>
              </select>
```



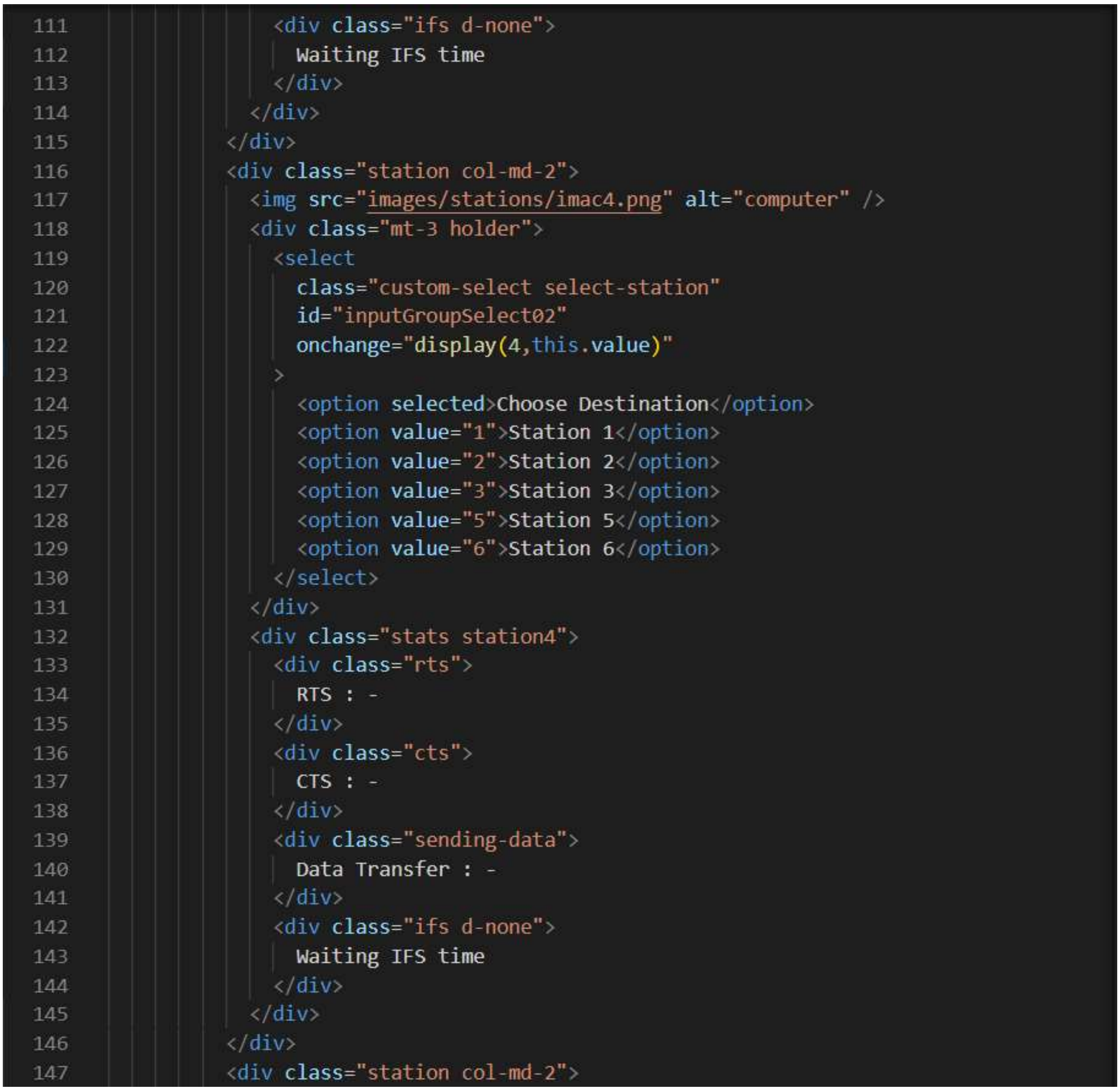
```html
            </div>
            <div class="stats station1">
              <div class="rts">
                RTS : -
              </div>
              <div class="cts">
                CTS : -
              </div>
              <div class="sending-data">
                Data Transfer : -
              </div>
              <div class="ifs d-none">
                Waiting IFS time
              </div>
            </div>
          </div>
```



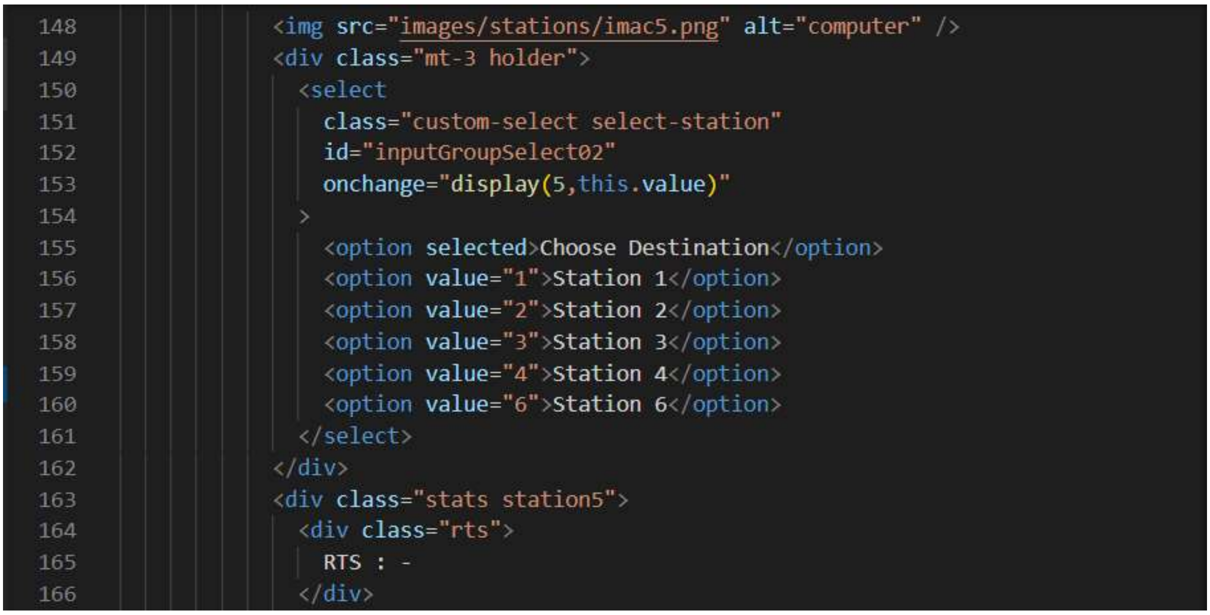
```html
<div class="station col-md-2">
  <img src="images/stations/imac2.png" alt="computer" />
  <div class="mt-3 holder">
    <select
      class="custom-select select-station"
      id="inputGroupSelect02"
      onchange="display(2,this.value)"
    >
      <option selected>Choose Destination</option>
      <option value="1">Station 1</option>
      <option value="3">Station 3</option>
      <option value="4">Station 4</option>
      <option value="5">Station 5</option>
      <option value="6">Station 6</option>
    </select>
  </div>
  <div class="stats station2">
    <div class="rts">
      RTS : -
    </div>
    <div class="cts">
      CTS : -
    </div>
    <div class="sending-data">
      Data Transfer : -
    </div>
    <div class="ifs d-none">
      Waiting IFS time
    </div>
  </div>
</div>
<div class="station col-md-2">
  <img src="images/stations/imac3.png" alt="computer" />
  <div class="mt-3 holder">
    <select
      class="custom-select select-station"
      id="inputGroupSelect02"
```



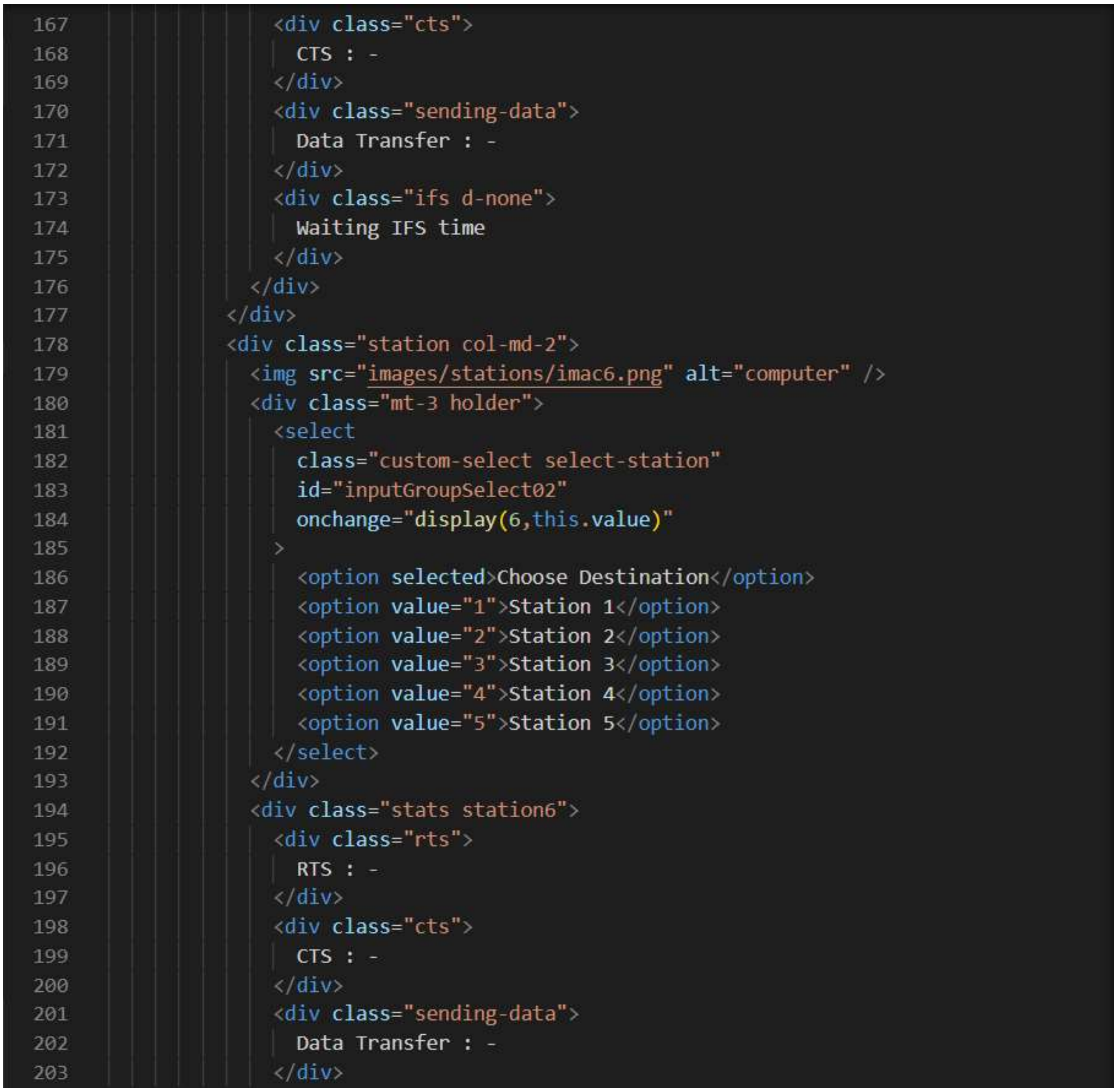
```html
      onchange="display(3,this.value)"
    >
      <option selected>Choose Destination</option>
      <option value="1">Station 1</option>
      <option value="2">Station 2</option>
      <option value="4">Station 4</option>
      <option value="5">Station 5</option>
      <option value="6">Station 6</option>
    </select>
  </div>
  <div class="stats station3">
    <div class="rts">
      RTS : -
    </div>
    <div class="cts">
      CTS : -
    </div>
    <div class="sending-data">
      Data Transfer : -
    </div>
```



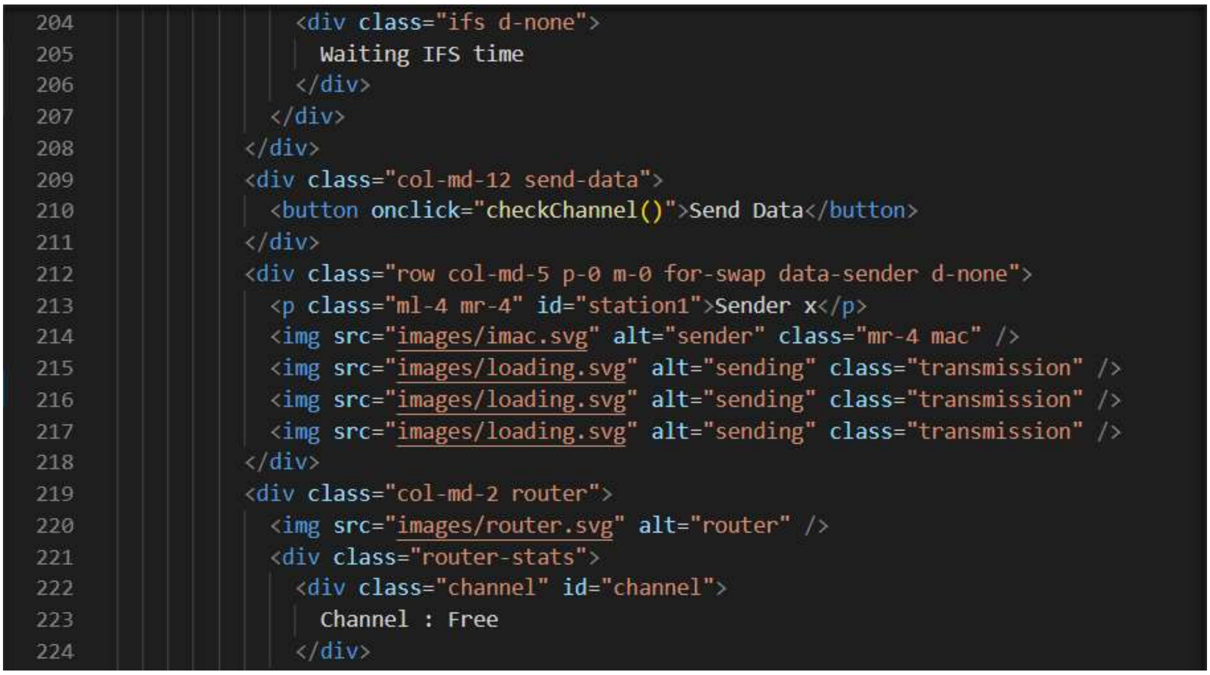
```html
    <div class="ifs d-none">
      Waiting IFS time
    </div>
  </div>
</div>
<div class="station col-md-2">
  <img src="images/stations/imac4.png" alt="computer" />
  <div class="mt-3 holder">
    <select
      class="custom-select select-station"
      id="inputGroupSelect02"
      onchange="display(4,this.value)"
    >
      <option selected>Choose Destination</option>
      <option value="1">Station 1</option>
      <option value="2">Station 2</option>
      <option value="3">Station 3</option>
      <option value="5">Station 5</option>
      <option value="6">Station 6</option>
    </select>
  </div>
  <div class="stats station4">
    <div class="rts">
      RTS : -
    </div>
    <div class="cts">
      CTS : -
    </div>
    <div class="sending-data">
      Data Transfer : -
    </div>
    <div class="ifs d-none">
      Waiting IFS time
    </div>
  </div>
</div>
<div class="station col-md-2">
```



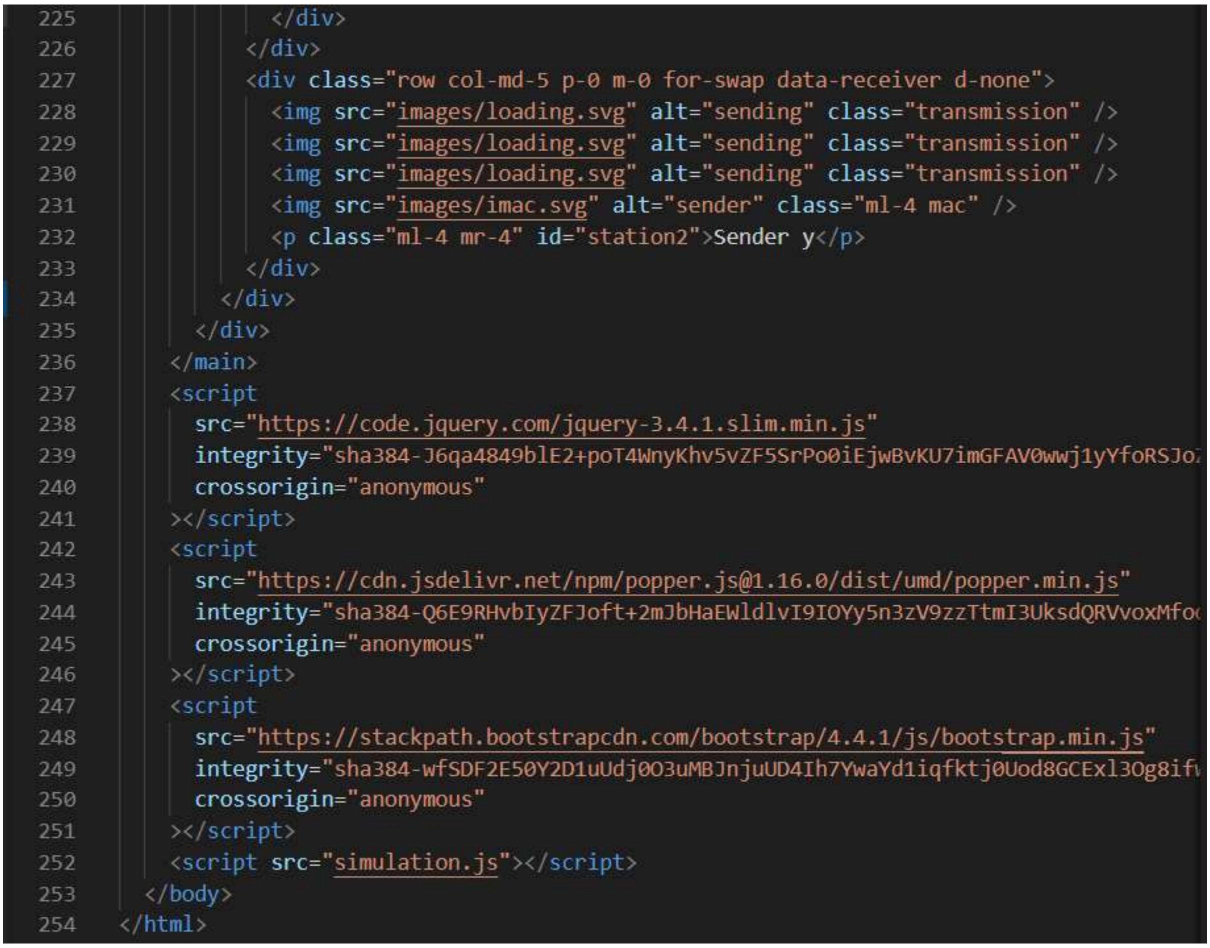
```html
  <img src="images/stations/imac5.png" alt="computer" />
  <div class="mt-3 holder">
    <select
      class="custom-select select-station"
      id="inputGroupSelect02"
      onchange="display(5,this.value)"
    >
      <option selected>Choose Destination</option>
      <option value="1">Station 1</option>
      <option value="2">Station 2</option>
      <option value="3">Station 3</option>
      <option value="4">Station 4</option>
      <option value="6">Station 6</option>
    </select>
  </div>
  <div class="stats station5">
    <div class="rts">
      RTS : -
    </div>
```



```
              <div class="cts">
                CTS : -
              </div>
              <div class="sending-data">
                Data Transfer : -
              </div>
              <div class="ifs d-none">
                Waiting IFS time
              </div>
            </div>
          </div>
          <div class="station col-md-2">
            <img src="images/stations/imac6.png" alt="computer" />
            <div class="mt-3 holder">
              <select
                class="custom-select select-station"
                id="inputGroupSelect02"
                onchange="display(6,this.value)"
              >
                <option selected>Choose Destination</option>
                <option value="1">Station 1</option>
                <option value="2">Station 2</option>
                <option value="3">Station 3</option>
                <option value="4">Station 4</option>
                <option value="5">Station 5</option>
              </select>
            </div>
            <div class="stats station6">
              <div class="rts">
                RTS : -
              </div>
              <div class="cts">
                CTS : -
              </div>
              <div class="sending-data">
                Data Transfer : -
              </div>
```

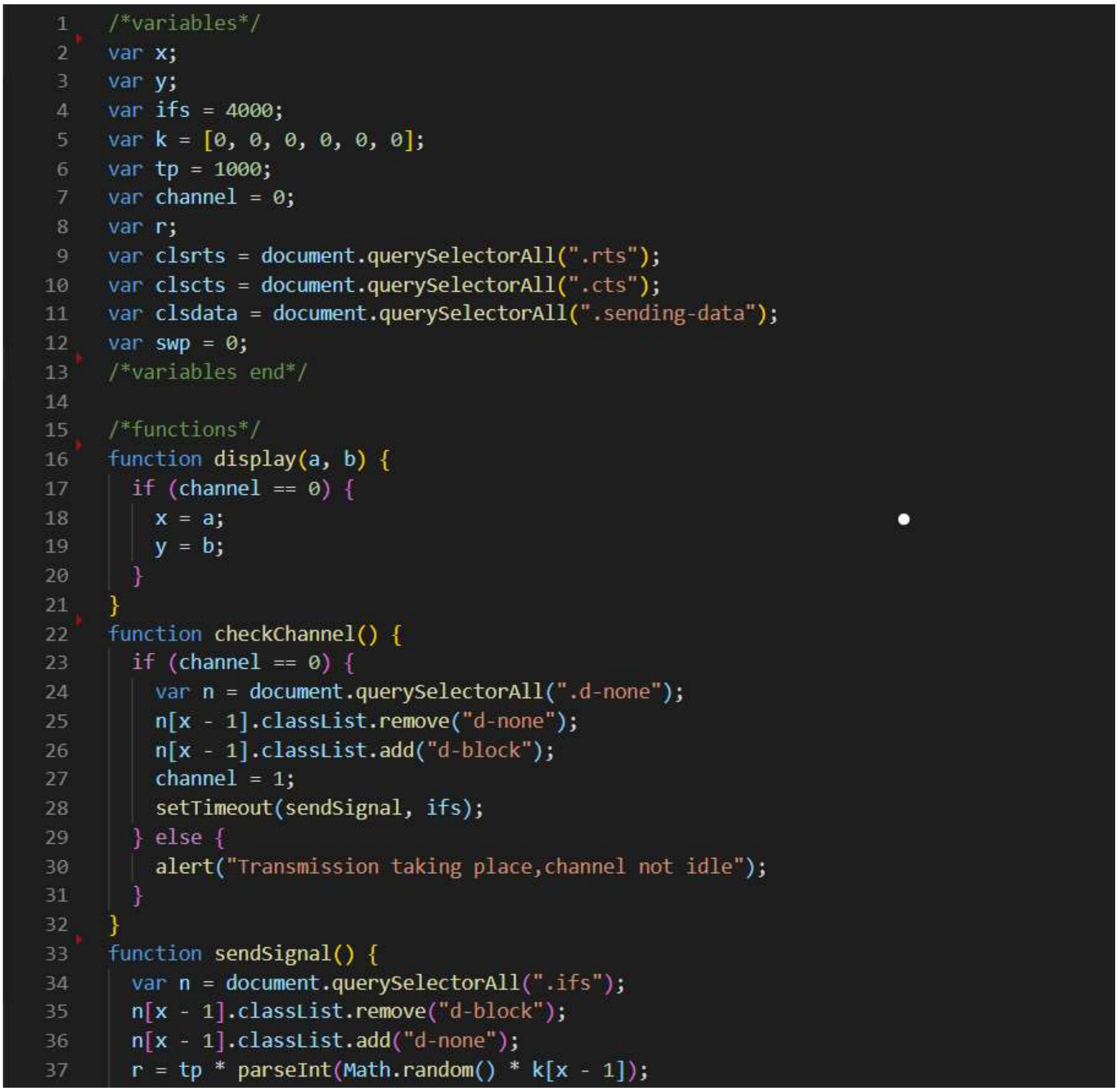


```
              <div class="ifs d-none">
                Waiting IFS time
              </div>
            </div>
          </div>
          <div class="col-md-12 send-data">
            <button onclick="checkChannel()">Send Data</button>
          </div>
          <div class="row col-md-5 p-0 m-0 for-swap data-sender d-none">
            <p class="ml-4 mr-4" id="station1">Sender x</p>
            <img src="images/imac.svg" alt="sender" class="mr-4 mac" />
            <img src="images/loading.svg" alt="sending" class="transmission" />
            <img src="images/loading.svg" alt="sending" class="transmission" />
            <img src="images/loading.svg" alt="sending" class="transmission" />
          </div>
          <div class="col-md-2 router">
            <img src="images/router.svg" alt="router" />
            <div class="router-stats">
              <div class="channel" id="channel">
                Channel : Free
              </div>
```



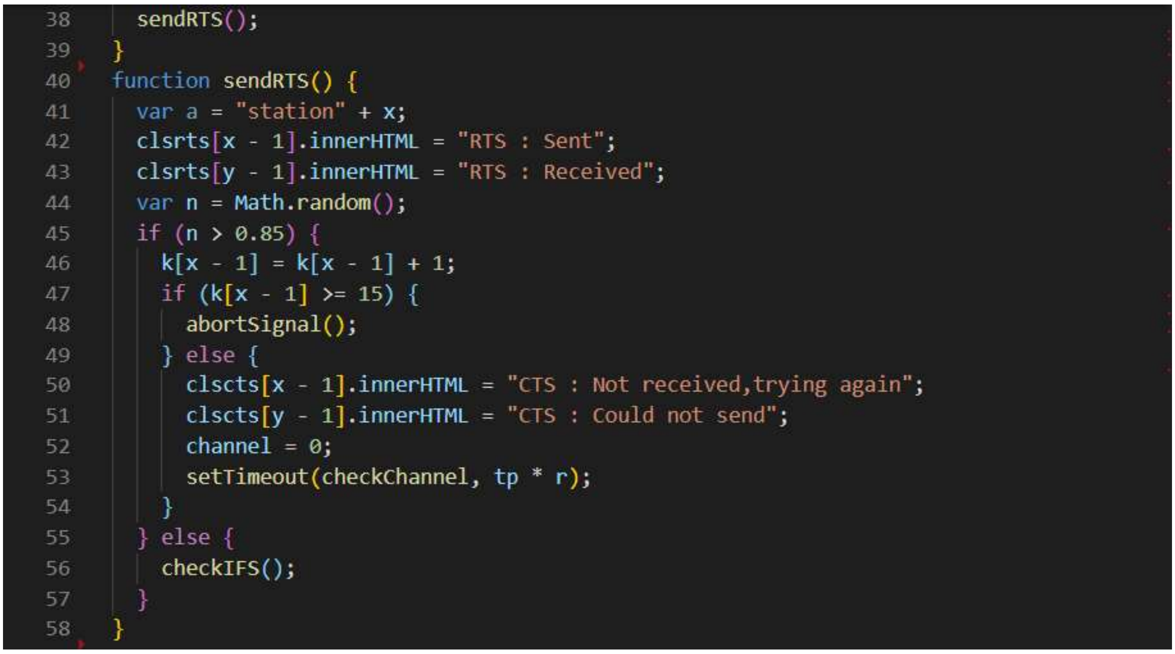
```
            </div>
          </div>
          <div class="row col-md-5 p-0 m-0 for-swap data-receiver d-none">
            <img src="images/loading.svg" alt="sending" class="transmission" />
            <img src="images/loading.svg" alt="sending" class="transmission" />
            <img src="images/loading.svg" alt="sending" class="transmission" />
            <img src="images/imac.svg" alt="sender" class="ml-4 mac" />
            <p class="ml-4 mr-4" id="station2">Sender y</p>
          </div>
        </div>
      </div>
    </main>
    <script
      src="https://code.jquery.com/jquery-3.4.1.slim.min.js"
      integrity="sha384-J6qa4849blE2+poT4WnyKhv5vZF5SrPo0iEjwBvKU7imGFAV0wwj1yYfoRSJo
      crossorigin="anonymous"
    ></script>
    <script
      src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js"
      integrity="sha384-Q6E9RHvbIyZFJoft+2mJbHaEWldlvI9IOYy5n3zV9zzTtmI3UksdQRVvoxMfo
      crossorigin="anonymous"
    ></script>
    <script
      src="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.js"
      integrity="sha384-wfSDF2E50Y2D1uUdj0O3uMBJnjuUD4Ih7YwaYd1iqfktj0Uod8GCExl3Og8if
      crossorigin="anonymous"
    ></script>
    <script src="simulation.js"></script>
  </body>
</html>
```

Javascript File:-

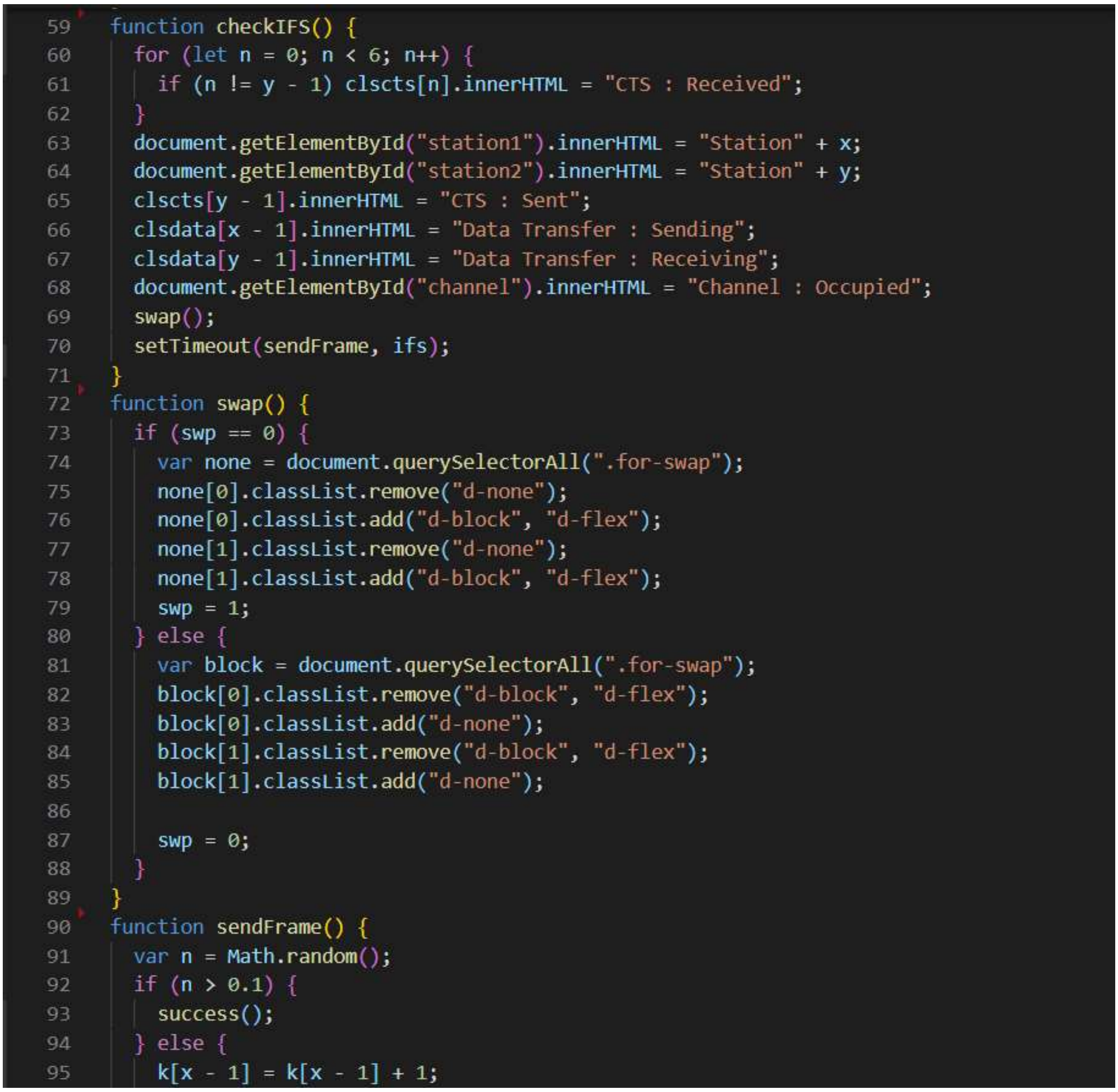


```
/*variables*/
var x;
var y;
var ifs = 4000;
var k = [0, 0, 0, 0, 0, 0];
var tp = 1000;
var channel = 0;
var r;
var clsrts = document.querySelectorAll(".rts");
var clscts = document.querySelectorAll(".cts");
var clsdata = document.querySelectorAll(".sending-data");
var swp = 0;
/*variables end*/

/*functions*/
function display(a, b) {
  if (channel == 0) {
    x = a;
    y = b;
  }
}
function checkChannel() {
  if (channel == 0) {
    var n = document.querySelectorAll(".d-none");
    n[x - 1].classList.remove("d-none");
    n[x - 1].classList.add("d-block");
    channel = 1;
    setTimeout(sendSignal, ifs);
  } else {
    alert("Transmission taking place,channel not idle");
  }
}
function sendSignal() {
  var n = document.querySelectorAll(".ifs");
  n[x - 1].classList.remove("d-block");
  n[x - 1].classList.add("d-none");
  r = tp * parseInt(Math.random() * k[x - 1]);
```

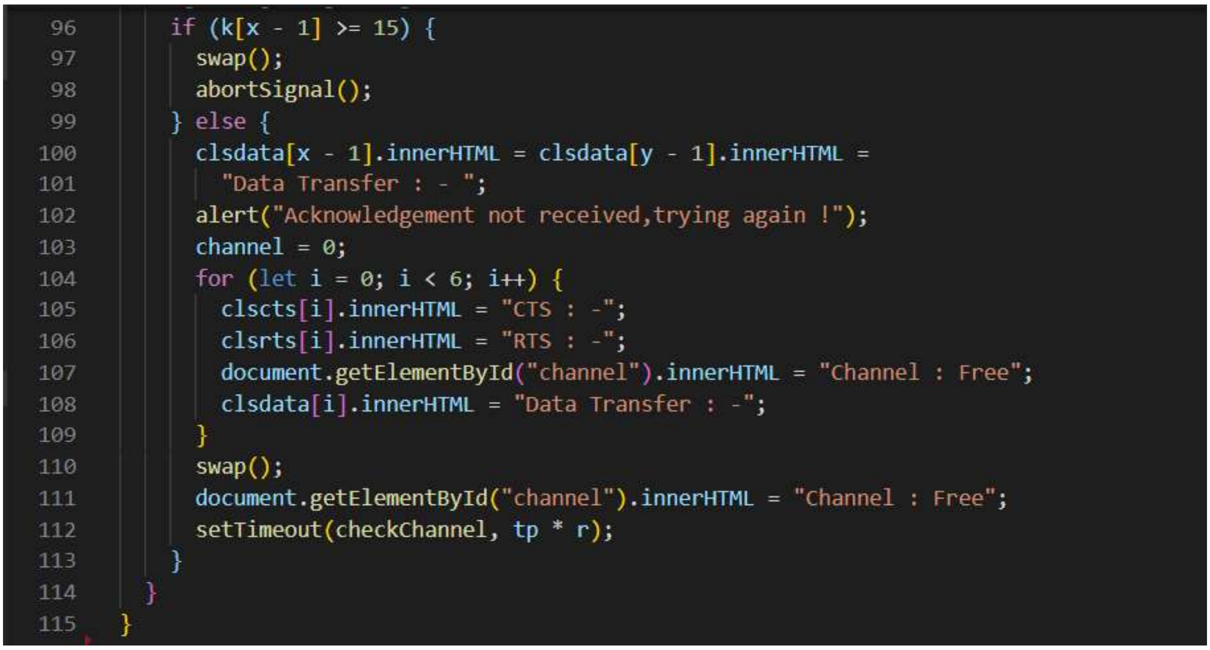


```
  sendRTS();
}
function sendRTS() {
  var a = "station" + x;
  clsrts[x - 1].innerHTML = "RTS : Sent";
  clsrts[y - 1].innerHTML = "RTS : Received";
  var n = Math.random();
  if (n > 0.85) {
    k[x - 1] = k[x - 1] + 1;
    if (k[x - 1] >= 15) {
      abortSignal();
    } else {
      clscts[x - 1].innerHTML = "CTS : Not received,trying again";
      clscts[y - 1].innerHTML = "CTS : Could not send";
      channel = 0;
      setTimeout(checkChannel, tp * r);
    }
  } else {
    checkIFS();
  }
}
```

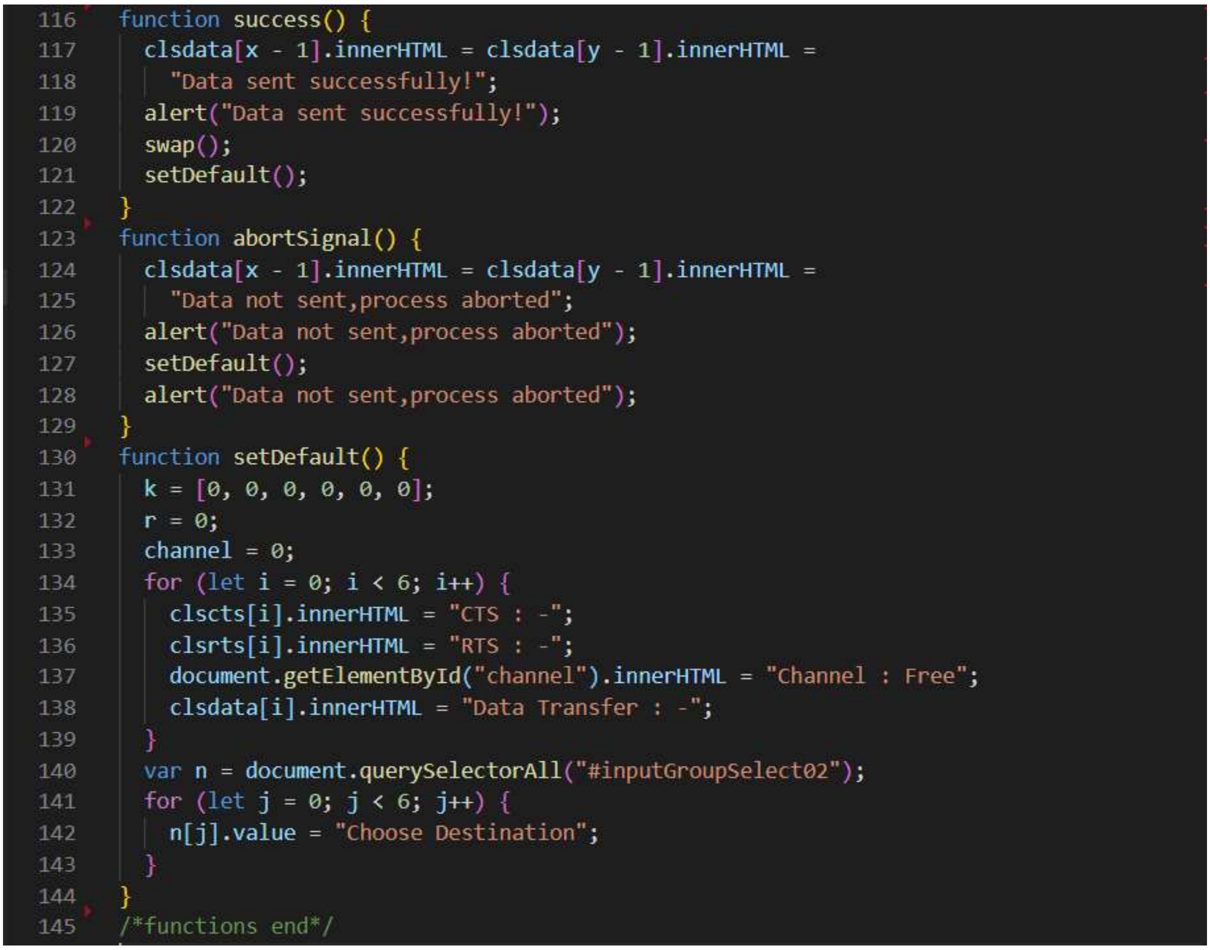


```javascript
function checkIFS() {
  for (let n = 0; n < 6; n++) {
    if (n != y - 1) clscts[n].innerHTML = "CTS : Received";
  }
  document.getElementById("station1").innerHTML = "Station" + x;
  document.getElementById("station2").innerHTML = "Station" + y;
  clscts[y - 1].innerHTML = "CTS : Sent";
  clsdata[x - 1].innerHTML = "Data Transfer : Sending";
  clsdata[y - 1].innerHTML = "Data Transfer : Receiving";
  document.getElementById("channel").innerHTML = "Channel : Occupied";
  swap();
  setTimeout(sendFrame, ifs);
}
function swap() {
  if (swp == 0) {
    var none = document.querySelectorAll(".for-swap");
    none[0].classList.remove("d-none");
    none[0].classList.add("d-block", "d-flex");
    none[1].classList.remove("d-none");
    none[1].classList.add("d-block", "d-flex");
    swp = 1;
  } else {
    var block = document.querySelectorAll(".for-swap");
    block[0].classList.remove("d-block", "d-flex");
    block[0].classList.add("d-none");
    block[1].classList.remove("d-block", "d-flex");
    block[1].classList.add("d-none");

    swp = 0;
  }
}
function sendFrame() {
  var n = Math.random();
  if (n > 0.1) {
    success();
  } else {
    k[x - 1] = k[x - 1] + 1;
```



```javascript
    if (k[x - 1] >= 15) {
      swap();
      abortSignal();
    } else {
      clsdata[x - 1].innerHTML = clsdata[y - 1].innerHTML =
        "Data Transfer : - ";
      alert("Acknowledgement not received,trying again !");
      channel = 0;
      for (let i = 0; i < 6; i++) {
        clscts[i].innerHTML = "CTS : -";
        clsrts[i].innerHTML = "RTS : -";
        document.getElementById("channel").innerHTML = "Channel : Free";
        clsdata[i].innerHTML = "Data Transfer : -";
      }
      swap();
      document.getElementById("channel").innerHTML = "Channel : Free";
      setTimeout(checkChannel, tp * r);
    }
  }
}
```



```
function success() {
  clsdata[x - 1].innerHTML = clsdata[y - 1].innerHTML =
    "Data sent successfully!";
  alert("Data sent successfully!");
  swap();
  setDefault();
}
function abortSignal() {
  clsdata[x - 1].innerHTML = clsdata[y - 1].innerHTML =
    "Data not sent,process aborted";
  alert("Data not sent,process aborted");
  setDefault();
  alert("Data not sent,process aborted");
}
function setDefault() {
  k = [0, 0, 0, 0, 0, 0];
  r = 0;
  channel = 0;
  for (let i = 0; i < 6; i++) {
    clscts[i].innerHTML = "CTS : -";
    clsrts[i].innerHTML = "RTS : -";
    document.getElementById("channel").innerHTML = "Channel : Free";
    clsdata[i].innerHTML = "Data Transfer : -";
  }
  var n = document.querySelectorAll("#inputGroupSelect02");
  for (let j = 0; j < 6; j++) {
    n[j].value = "Choose Destination";
  }
}
/*functions end*/
```

RESULT SCREENSHOTS

Image 1: The Landing Page.

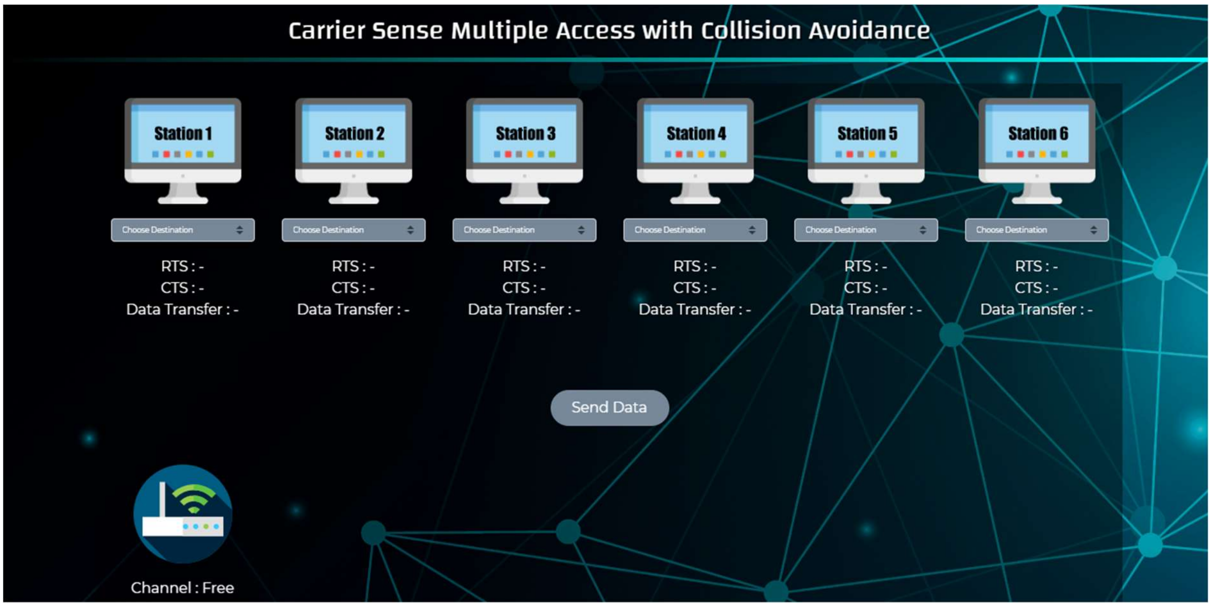


Image 2: Selecting Destination Station.

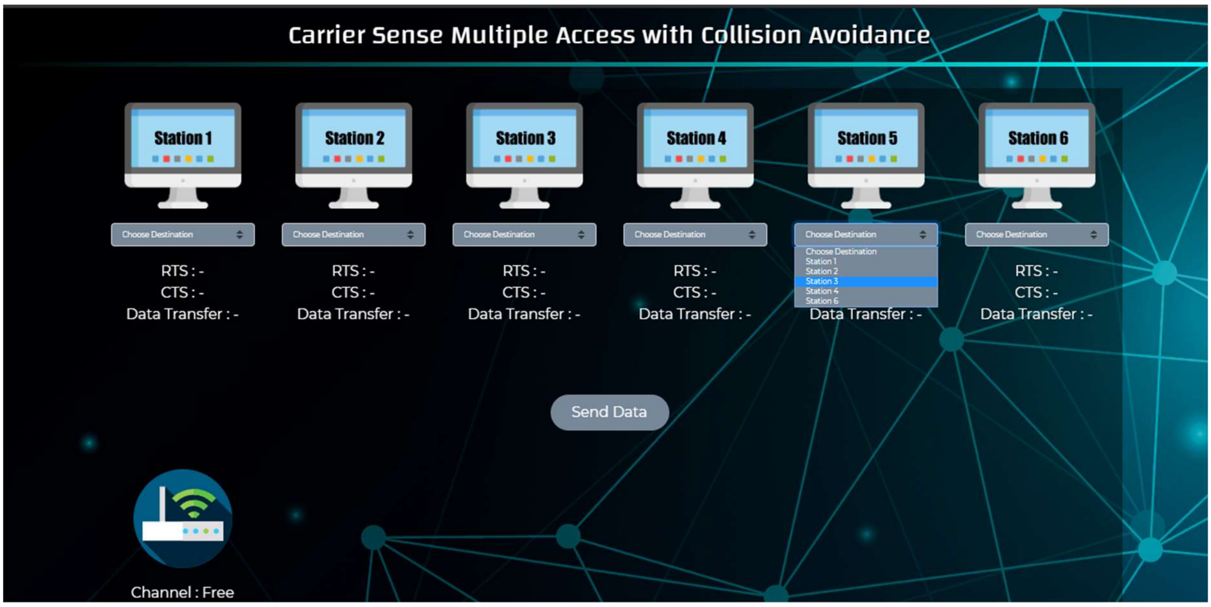
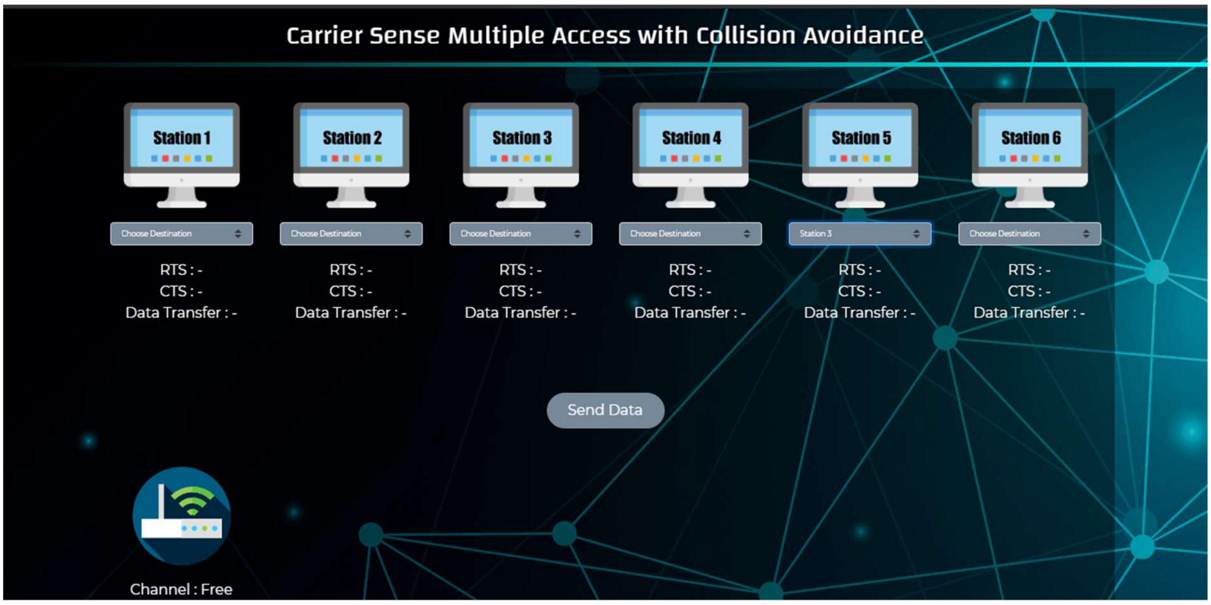


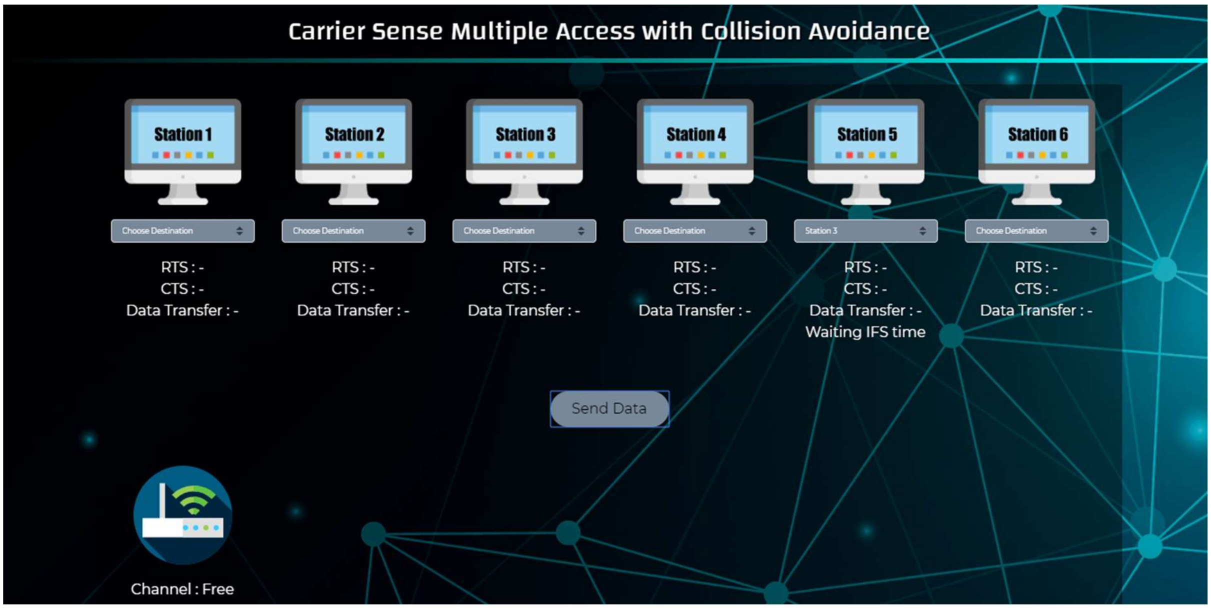
Image 3: Destination Station Selected.



RTS : -
CTS : -
Data Transfer : -

Channel : Free

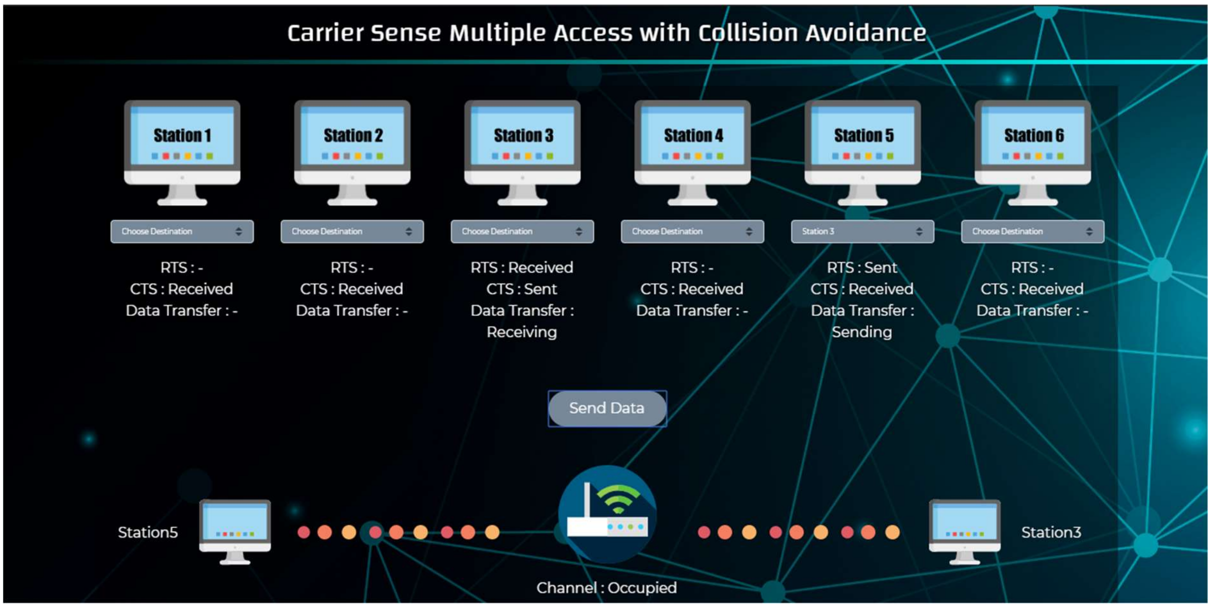
Image 4: Waiting for IFS time.



Station 5:
RTS : -
CTS : -
Data Transfer : -
Waiting IFS time

Channel : Free

Image 5: Data sent.



Carrier Sense Multiple Access with Collision Avoidance

Station 1 — Choose Destination — RTS : - CTS : Received Data Transfer : -

Station 2 — Choose Destination — RTS : - CTS : Received Data Transfer : -

Station 3 — Choose Destination — RTS : Received CTS : Sent Data Transfer : Receiving

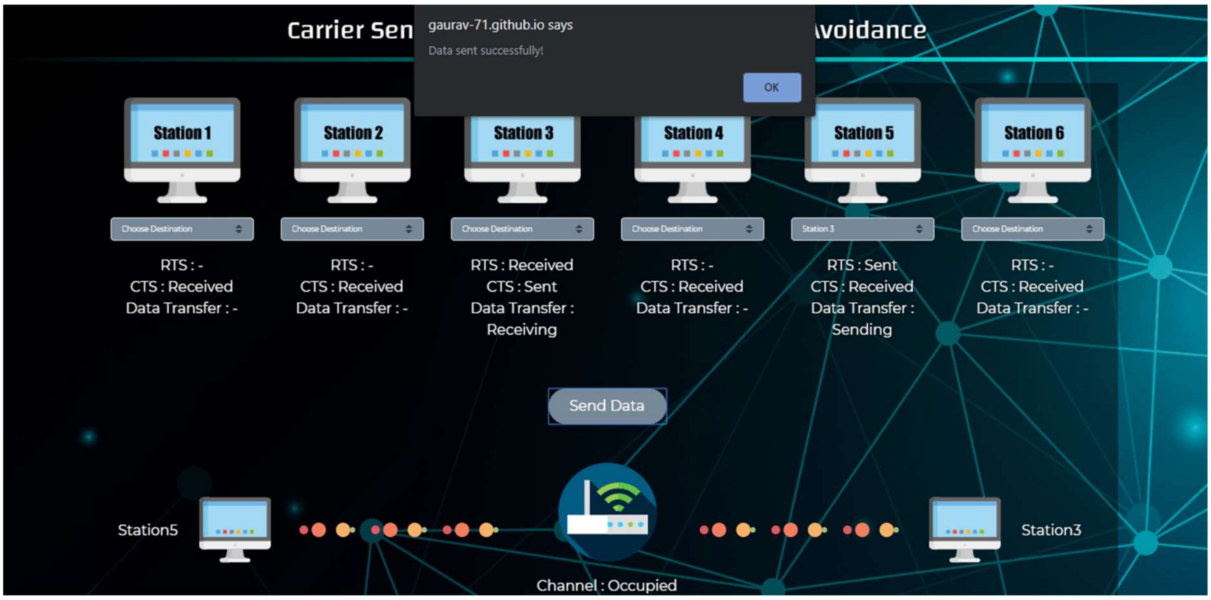
Station 4 — Choose Destination — RTS : - CTS : Received Data Transfer : -

Station 5 — Station 3 — RTS : Sent CTS : Received Data Transfer : Sending

Station 6 — Choose Destination — RTS : - CTS : Received Data Transfer : -

Channel : Occupied

Image 6: Confirmation of successful communication received.



gaurav-71.github.io says

Data sent successfully!

Carrier Sen… …voidance

Station 1 — Choose Destination — RTS : - CTS : Received Data Transfer : -

Station 2 — Choose Destination — RTS : - CTS : Received Data Transfer : -

Station 3 — Choose Destination — RTS : Received CTS : Sent Data Transfer : Receiving

Station 4 — Choose Destination — RTS : - CTS : Received Data Transfer : -

Station 5 — Station 3 — RTS : Sent CTS : Received Data Transfer : Sending

Station 6 — Choose Destination — RTS : - CTS : Received Data Transfer : -

Channel : Occupied

REFERENCES

1. Data Communication and Networking, Behrouz A Forouzan, McGraw Hill, 5th Edition, 2008.

2. https://www.wikipedia.org/wiki/Carrier-sense_multiple_access_with_collision_avoidance

3. https://www.youtube.com/watch?v=PcbTMSf0D2M

4. https://www.ionos.com/digitalguide/server/know-how/csmaca-carrier-sense-multiple-access-with-collision-avoidance/