```
//Øving 7 i nettverksprogrammering TDAT2004
//31.03.2020 Kasper Vedal Gundersen

"use strict";

const net = require("net");
const crypto = require("crypto");

const httpServer = net.createServer(connection => {
  connection.on("data", () => {
    let content = `<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8" />
    <style>
    div {
      text-align: center;
    }
    body {
      background-color: #303030;
    }
    </style>
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min
.js"></script>
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.
min.js"></script>
  </head>
  <body>
    <div>
      <h1 style="color: white">WebSocket</h1>
      <input type="text" id="inputMelding" name="inputMelding"/>
      <button onClick="sendMelding()">Send melding</Button>
      <div id="melding" style="color: white"></div>
      <img id="wow" width="40%"></img>
    </div>
    <script>
      let ws = new WebSocket('ws://localhost:3001');
      ws.onmessage = event => {document.getElementById("melding").appendChild(
document.createTextNode(event.data))};
      ws.onopen = () => ws.send('Melding: ');
      function sendMelding() {
        var meld = document.getElementById("inputMelding").value;
        ws.send(meld);
      }
    </script>
  </body>
</html>
`;
    connection.write(
```

```
      "HTTP/1.1 200 OK\r\nContent-
Length: " + content.length + "\n\n" + content
    );
  });
});
httpServer.listen(3000, () => {
  console.log("HTTP server listening on port 3000");
});

var connections = new Set();
const wsServer = net.createServer(connection => {
  console.log("Klient tilkoblet");
  var state = 0;
  connections.add(connection);

  connection.on("data", data => {
    if (state === 0) {
      console.log("Headers fra klient:\n" + data.toString());
      var headers = data.toString().split("\r\n");
      var key = headers
        .find(header => header.indexOf("Sec-WebSocket-Key") !== -1)
        .split(": ")[1];
      var acceptValue = generateAcceptValue(key);
      const responseHeaders = [
        "HTTP/1.1 101 Switching Protocols",
        "Upgrade: websocket",
        "Connection: Upgrade",
        `Sec-WebSocket-Accept: ${acceptValue}`
      ];
      connection.write(responseHeaders.join("\r\n") + "\r\n\r\n");
      state = 1;
    } else {
      console.log("Data fra klient: ", data);
      let bytes = Buffer.from(data);
      let lengde = bytes[1] & 127;
      let maskStart = 2;
      if (lengde == 126) {
        maskStart = 4;
      } else if (lengde == 127) {
        maskStart = 10;
      }
      let dataStart = maskStart + 4;
      let output = "";
      for (let i = dataStart; i < dataStart + lengde; i++) {
        let byte = bytes[i] ^ bytes[maskStart + ((i - dataStart) % 4)];
        output += String.fromCharCode(byte);
      }
      for (let sock of connections) {
        send(output, sock);
```

```
      }
    }
  });

  connection.on("message", message => {
    console.log("Melding fra klient: ", message);
  });

  connection.on("ready", message => {
    console.log("Tilkobling opprettet");
  });

  connection.on("error", error => {
    console.error("Error: ", error);
  });

  connection.on("end", () => {
    console.log("Klient avsluttet");
    connections.delete(connection);
  });
});

const generateAcceptValue = acceptKey => {
  return crypto
    .createHash("sha1")
    .update(acceptKey + "258EAFA5-E914-47DA-95CA-C5AB0DC85B11", "binary")
    .digest("base64");
};

function send(melding, socket) {
  var ramme = new Buffer(melding.length > 125 ? 4 : 2);
  ramme[0] = 0x81;
  if (melding.length > 125) {
    ramme[1] = 126;
    var lengde = melding.length;
    ramme[2] = lengde >> 8;
    ramme[3] = lengde & 0xff;
  } else {
    ramme[1] = melding.length;
  }
  socket.write(ramme, "binary");
  socket.write(melding, "utf8");
}

wsServer.listen(3001, () => {
  console.log("WebSocket server listening on port 3001");
});
```