Name : rathod nency vijaybhai

Sem: 7th

Roll no: 90

Div:B

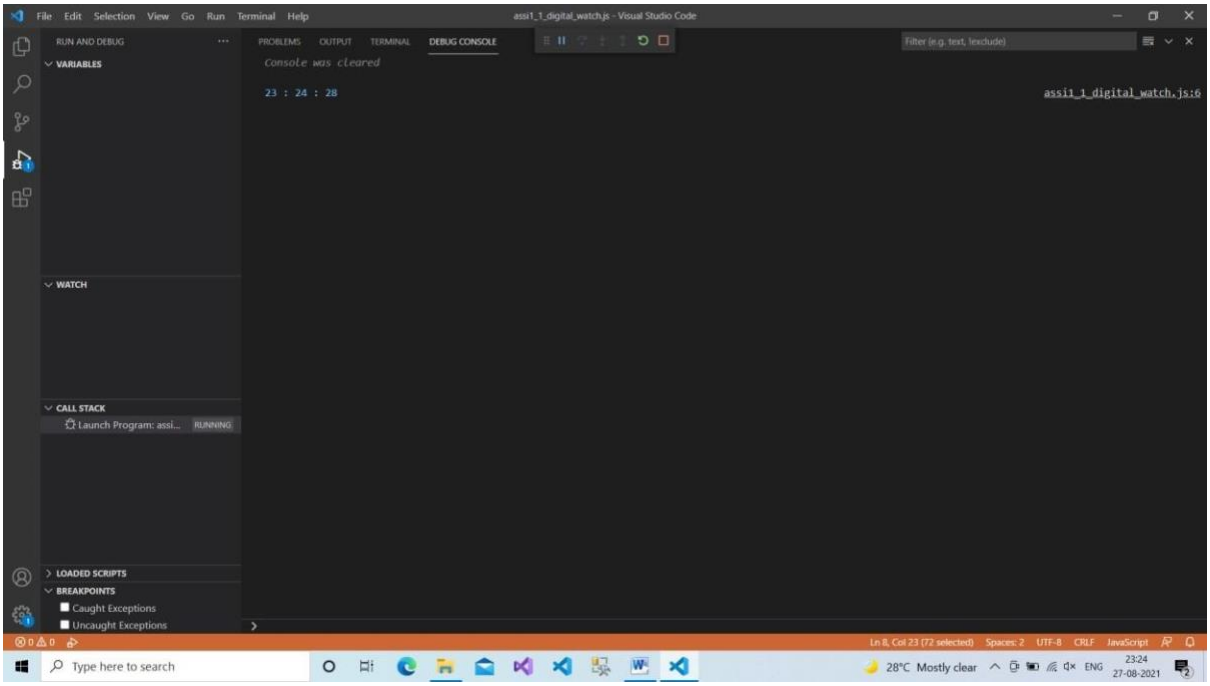
Sub:NodeJS

Practical Assignment – 1

==========Q-1==========

```
function currentTime() {      var date = new Date(); /*
creating object of Date class */     var hour =
date.getHours();      var min = date.getMinutes();
var sec = date.getSeconds();      console.log(hour + " : "
+ min + " : " + sec);      var t = setTimeout(function ()
{           console.clear();           currentTime();
    }, 1000);
}
currentTime();
```

==========output==========



==========Q2==========

```
const express
=require("express"); const fs =
require("fs");  const app =
express(); app.set("view engine",
"ejs"); app.get("/", function
(req, res) {
  res.render(__dirname + "/file/index.ejs");
});   app.get("/rendervideo", function
(req, res)
{   const path = "assets/3.mp4";
const stat = fs.statSync(path);
const fileSize = stat.size;    const
range = req.headers.range;

  if (range) {
    const parts = range.replace(/bytes=/, "").split("-");
const start = parseInt(parts[0], 10);
    const end = parts[1] ? parseInt(parts[1], 10) : fileSize - 1;
const chunkSize = end - start + 1;      const file =
fs.createReadStream(path, { start, end });     const head = {
      "Content-Range": `bytes ${start}-${end}/${fileSize}`,
      "Accept-Ranges": "bytes",
      "Content-Length": chunkSize,
      "Content-Type": "video/mp4",
```

```
    };     res.writeHead(206, head);     file.pipe(res);    }
else {
    const head = {
      "Content-Length": fileSize,
      "Content-Type": "video/mp4",      };     res.writeHead(200, head);
fs.createReadStream(path).pipe(res);
  }
});  app.listen(3000, function () {    console.log("Running on 3000 port");
});
```

==========Q3==========

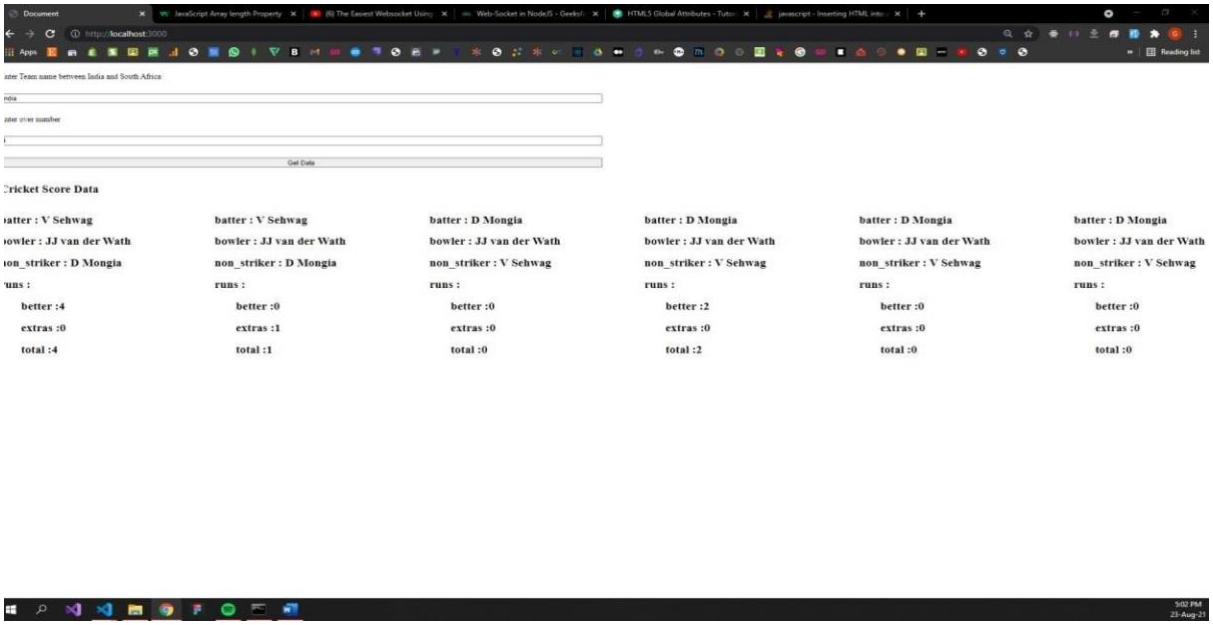
```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />     <meta
name="viewport" content="width=device-width, initial-scale=1.0" />
<title>Document</title>
  </head>
  <style>     .inner
{
      max-width: 20vw;
    }
    #scoredata {       display: flex;
flex-direction: row;       justifycontent:
space-between;
    }
  </style>
  <body>     <div       style="
display: flex;          flex-
direction: column;          max-
width: 50%;          height:
20vh;
        justify-content: space-around;
      "
    >
      <label for="">Enter Team name between India and South Africa</label>
      <input type="text" id="team" />
```

```
      <label for="">Enter over number</label>
      <input type="number" id="overnumber" max="20" min="0" />
      <button onclick="sendData()">Get Data</button>
</div>
    <h2>Cricket Score Data</h2>
    <div id="scoredata"></div>
  </body>
  <script src="https://cdn.socket.io/4.0.1/socket.io.js"></script>
  <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.j
s"></script>          <script>                    var    socket    =
io().connect("http://localhost:3000");        function sendData() {
var team = $("#team").val();       var over = $("#overnumber").val();
const cricdata = {              team: team,              overnumber:
over,
      };       console.log(cricdata);       socket.emit("getData",
cricdata);
    }     socket.on("sendData", function (data) {        console.log(data);
var score = "";        if (data.length > 0) {          for (let i = 0; i <
data.length; i++) {            score += `<div class="inner"> <h2>batter :
${data[i].batter}</h2><h2 >bowler : ${data[i].bowler} </h2><h2>non_striker :
${data[i].non_striker}</h2>
<h2>runs :</h2><h2>       better :${data[i].runs
.batter}</h2><h2>       extras :${data[i].runs.e
xtras}</h2><h2>       total :${data[i].runs.tota
l}</h2></div>`;
        }
        $("#scoredata").html(score);
      } else {
        $("#scoredata").html("<h1>NO DATA FOUND</h1>");
      }       console.log(score);
    });
  </script>
</html>
```

```
const express = require("express"); const app =
express(); const http =
require("http").Server(app).listen(3000);
```

```
const io = require("socket.io")(http);
app.set("view engine", "ejs");
app.use(express.urlencoded({ extended: true }));
var cricketdata = require("./cricket.json");
app.get("/", (req, res) =>
{   res.render("index");
}); io.on("connection", (socket) => {
console.log("User is connected");
socket.on("getData", (data) => {      for (var i = 0;
i < cricketdata.innings.length; i++) {
console.log(i);
if (cricketdata.innings[i].team == data.team) {
        // console.log(cricketdata.innings[i].overs[0].deliveries);
console.log(cricketdata.innings[i].overs[data.overnumber].deliveries);
socket.emit(           "sendData",
cricketdata.innings[i].overs[data.overnumber].deliveries
        );          break;
} else {
socket.emit("sendData",
[]);
      }
    }
  });
});
```

==========output==========


Document
http://localhost:3000
Apps
Reading list
ater Team name between India and South Africa
ndia
ater over number
Get Data
Cricket Score Data
batter : V Sehwag
bowler : JJ van der Wath
non_striker : D Mongia
runs :
better :4
extras :0
total :4
batter : V Sehwag
bowler : JJ van der Wath
non_striker : D Mongia
runs :
better :0
extras :1
total :1
batter : D Mongia
bowler : JJ van der Wath
non_striker : V Sehwag
runs :
better :0
extras :0
total :0
batter : D Mongia
bowler : JJ van der Wath
non_striker : V Sehwag
runs :
better :2
extras :0
total :2
batter : D Mongia
bowler : JJ van der Wath
non_striker : V Sehwag
runs :
better :0
extras :0
total :0
batter : D Mongia
bowler : JJ van der Wath
non_striker : V Sehwag
runs :
better :0
extras :0
total :0

==========Q4==========

```
const express = require("express");
const socket =require("socket.io");
 const app = express();
const cors = require("cors");
require("dotenv").config();
 app.use(cors());
app.use(express.json());
app.set("view engine","ejs");
const server = app.listen(process.env.PORT || 3002, () => {
console.log("Server Running on Port 3002...");
});
io = socket(server);
 io.on("connection", (socket) =>
{   console.log(socket.id);
   socket.on("join_room", (data) => {
socket.join(data);
 console.log("User Joined Room: " + data);
  });
 socket.on("send_message", (data) => {
console.log(data);









    socket.to(data.room).emit("receive_message", data.content);
  });    socket.on("disconnect", () => {
console.log("USER DISCONNECTED");
  });
});
```