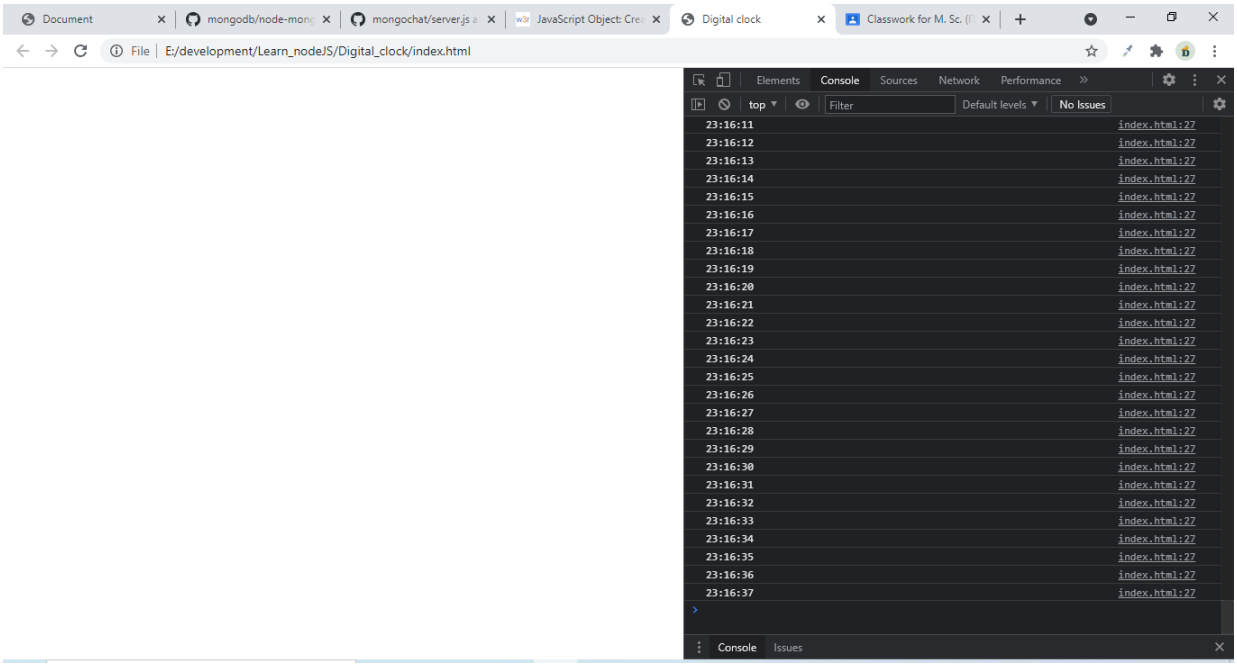
***Name:Muskan Vinod Aneja***
***Roll No:2***
***Subject :Application Development using MEAN stack***
***Sem:7***
***Assignment:Practical Assignment 1***
***----------***

1) Command line based digital clock with date and time.

```
<!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>Digital clock</title>
</head>
<body>

</body>
<script>
        function my_Clock()
    {
    this.cur_date   =   new   Date(); this.hours =
    this.cur_date.getHours();
    this.minutes = this.cur_date.getMinutes(); this.seconds =
    this.cur_date.getSeconds();
    }
my_Clock.prototype.run = function ()
    {
    setInterval(this.update.bind(this), 1000);
    };
my_Clock.prototype.update = function ()
    {
    this.updateTime(1);
    console.log(this.hours + ":" + this.minutes + ":" + this.seconds);
    };
my_Clock.prototype.updateTime = function (secs)
    {
```

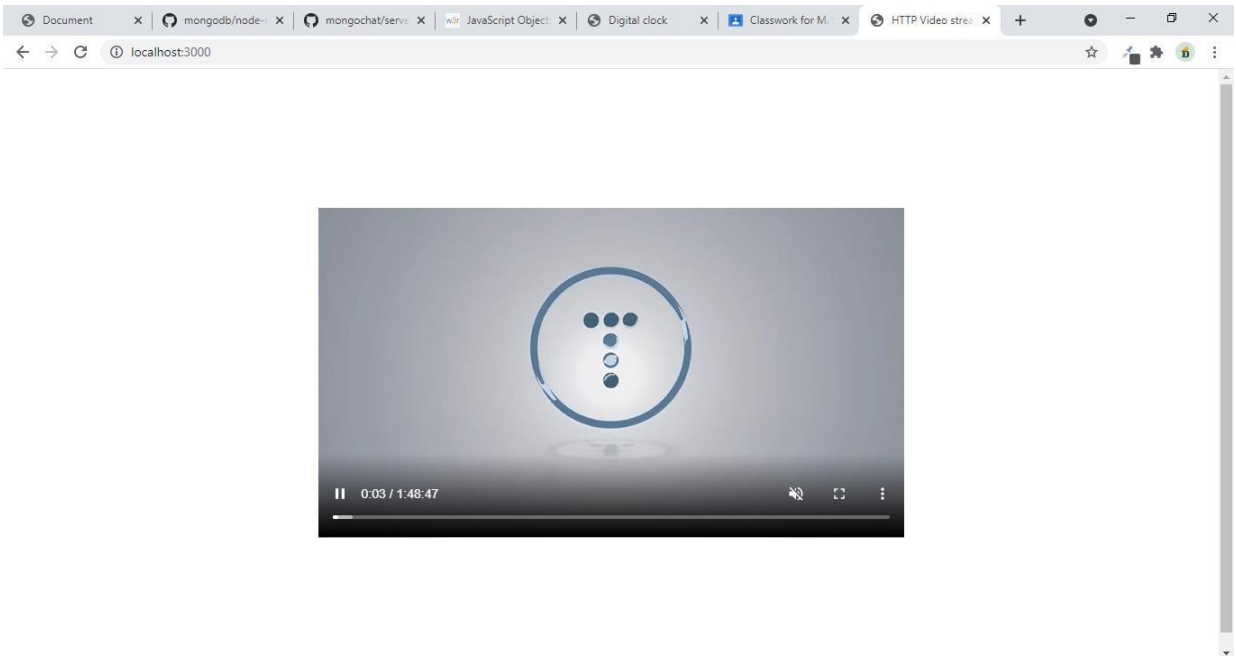
```
    this.seconds+=  secs; if
  (this.seconds >= 60)
    {
    this.minutes++;
    this.seconds= 0;
    }
  if (this.minutes >= 60)
    {
    this.hours++;
    this.minutes=0;
    }
if (this.hours >= 24)
    {
    this.hours = 0;
    }
};
var clock = new my_Clock();
    clock.run();
</script>
</html>
```



## 2) Develop a Video streaming server using nodejs.

```
const path = require("path"); const
express = require("express"); const app =
express();
const PORT = 3000;
const fs = require("fs");

app.get("/",(req,res)=>{
    res.sendFile(path.join(_dirname,"index.html"));
});

app.get("/video",(req,res)=>{
    const range = req.headers.range; if(!range)
    {
        res.status(400).send("Requires range headers.                    ");
    }
    const videoPath = "React JS Crash Course 2021.mp4";
    const videoSize = fs.statSync("React JS Crash Course
2021.mp4").size;
    const CHUNK_SIZE = 10**6; //1MB
    const start = Number(range.replace(/\D/g, ""));
    const end = Math.min(start + CHUNK_SIZE, videoSize - 1); const
    contentLenght = end -start + 1;
    const headers = {
        'Content-Range':`bytes ${start}-${end}/${videoSize}`,
        "Accept-Ranges":"bytes",
        "Content-Length": contentLenght,
```

```
            "Content-Type":"video/mp4"
        };
                                                    res.writeHead(206,headers);
                                                    const videoStream =
                                                    fs.createReadStream(videoPath,{start,end});
                                                }); videoStream.pipe(res);

app.listen(PORT,()=>{
    console.log(`Server is listening on PORT ${PORT}`);
});
```



3) Live cricket score application with websocket in nodejs.

**Score.json**

```
{
    "runs":"180", "wicket":5
}
```

**Index.js**

```
const fs=require("fs");
const express = require("express"); const
app = express();
const path = require("path");
const server = require("http").createServer(app); const PORT
= process.env.PORT||3000;
const io = require("socket.io")(server,{cors:{origin:"*"}});



app.use(express.static(path.join(_dirname,"index.html")));

app.get("/",(req,res)=>{
    res.sendFile(path.join(__dirname,"index.html"));
});
```

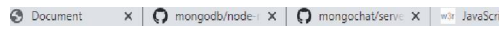
```
server.listen(PORT,()=>{
        console.log(`Server is running on PORT ${PORT}`);
});

io.on("connection",(socket)=>{
    var json_file = fs.readFileSync("./score.json");

    var data = JSON.parse(json_file);
    console.log(data); io.emit('output',data);

});
```

**Index.html**

```
<!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>
        <script src="https://cdnjs.cloudflare.com/ajax/libs/socket.io/4.1.3/
socket.io.min.js"></script>
```

```
</head>
<body>
    <table>
        <thead>
            <tr>
                <th colspan="2" align="center">Cricket Live
score</td>
                </tr>
                <tr>
                    <th>Runs</th>
                    <th>Wickets</th>
                </tr>
        </thead>
        <tbody>
         <tr>
            <td  id="runs"></td>
              <td  id="wicket"></td>
         </tr>
        </tbody>
    </table>
</body>
<script>
      const  socket  =  io("http://localhost:3000");
      socket.on('connection');
```

```
        var runs = document.getElementById("runs");
        var Wickets = document.getElementById("wicket");

        function getData()
        {
            socket.on('output',(data)=>{ runs.textContent
                = data.runs; Wickets.textContent =
                data.wicket;
            });
        }
        getData(); setInterval(getData(),1000);
</script>
</html>
```




€- 1 Iorahosi:3000

Cricket Live score

Runs Wickets

180 5


p ob x | @ crass ork or n x | , '. Ream ie Jso x @ com true I x + e a x

*- 1 Iorahos‹:3000

Cricket Live score Runs

Wickets

250 5

4) Develop a web based chat bot of any domain using client and server JavaScript. Use ws or socket.io or any library. Use any of mysql/mongodb for data storage.

## Index.js

```
const express = require("express"); const
app = express();
const server = require("http").createServer(app); const path =
require("path");
const io = require("socket.io")(server,{cors:{origin:"*"}}); const PORT =
process.env.PORT || 3000;
const mongo = require("mongodb");
const { MongoClient } = require("mongodb");

const url = "mongodb://127.0.0.1:27017"; const
client =  new  MongoClient(url); const dbname =
"mongochat";



app.use(express.static(path.join(__dirname,"")));

app.get("/",(req,res)=>{
    res.sendFile(path.join(__dirname,"index.html"));
})

server.listen(PORT,()=>{
    console.log(`Server is listening to PORT ${PORT}`);
});
```

```
// Connect to socket io.on('connection',async
(socket)=>{
	// User connection socket.id....
	console.log("User connected with : "+socket.id);

	// Connecting with MongoDB Database....
	await client.connect();
	console.log("Connection   with MongoDB  established.                    ");

	socket.on('message',(data)=>{ console.log(data);
	});

	sendStatus = function(s)
	{
		socket.emit('status',s);
	}

	var db = client.db(dbname);
	var chat_data = db.collection("chats");
	var getResult = chat_data.find().toArray((err,res)=>{

		if(err)
		{
			throw err;
		}
```

```
        socket.emit('output',res);
    });


    socket.on('input',(data)=>{
        let name = "Dhawal Parmar"; let
        message = data.message;

        if(message=="")
        {
            sendStatus("Please enter message.                    ");
        }
        else
        {
                                                    chat_data.insertOne({ name:
=> {
                                                        name, message: message },
                                                        () socket.emit('output',
                                                        [data]);

                                                        sendStatus({
                                                            sta
                                                            tus
                                                            :
                                                            'Go
                                                            t
                                                            yo
                                                            ur
```

```
        me                                                  cle
        ssa                                                 ar:
        ge..                                                tru
        ',                                                  e
                                                });
                                            });
                                    }


});

socket.on('clear',(data)=>{
```

```
            chat_data.deleteMany({},()=>{
                socket.emit("cleared");
            });
        })



        // Database...
        // const db = client.db(dbname);
        // const collection = db.collection('chats');

        // Check data from collection
        // const getResult = await collection.find().toArray();
        // console.log(getResult);

});
```

## Index.html

```
<!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">
        <link rel="stylesheet" href="bootstrap.min.css">
        <title>Document</title>
```

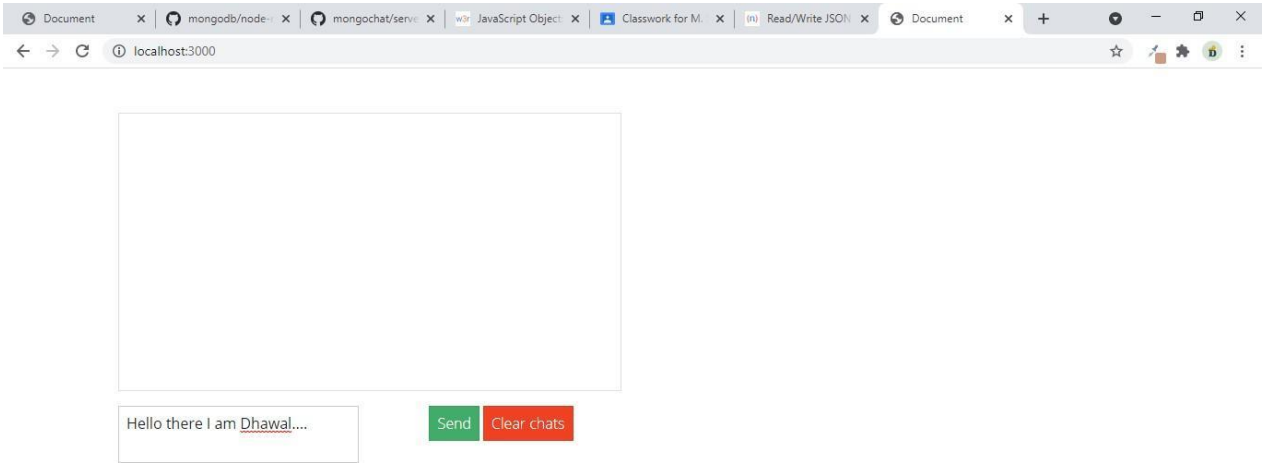
```
    <script src="https://cdnjs.cloudflare.com/ajax/libs/socket.io/4.1.3/socket.io. min.js"></script>
    <style>
        #messages{
            height:300px;
        }

    </style>
</head>
<body>
    <div class="container mt-5">
        <div class="row">
            <div  class="col-md-6  col-sm-6">
                <div class="card">
                    <div class="card-block" id="messages"></div>
                </div>
                <div class="row mt-3 text-center">

                    <div  class="col-sm-6">
                    <textarea id="textarea" style="resize: none;"
placeholder="enter message"          class="form-control"></textarea>
                      </div>

                      <div class="col">
                    <button id="sendBtn" class="btn btn-
success">Send</button>
                    <button id="clearBtn" class="btn btn-danger">Clear
chats</button>
                </div>
```

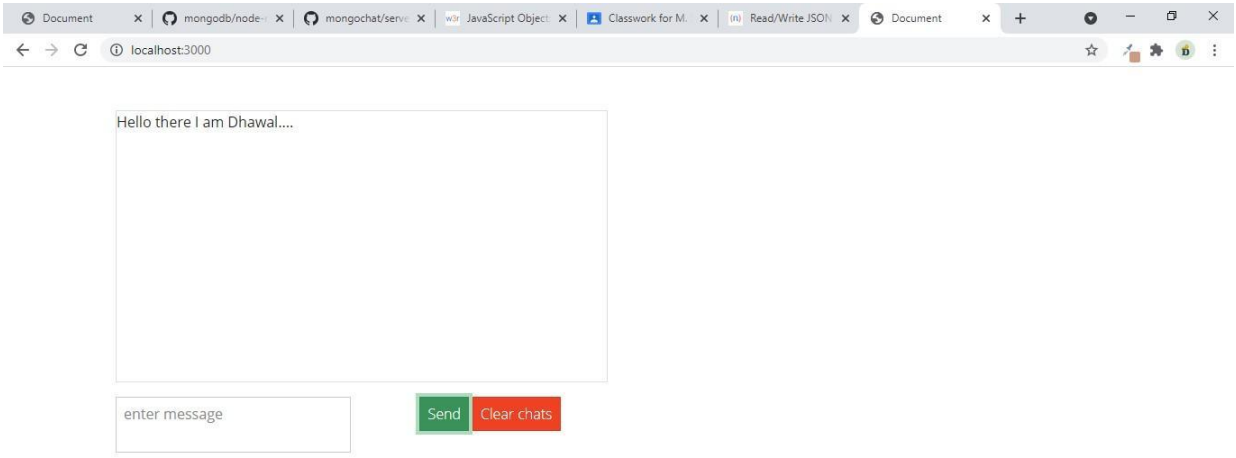
```
                    </div>
                </div>
            </div>
        </div>
</body>
<script>

        var element = (id)=>{
            return document.getElementById(id);
        };

        var messages = element("messages"); var
        sendBtn =  element("sendBtn"); var
        clearBtn = element("clearBtn"); var
        textarea = element("textarea");

        const socket = io("http://localhost:3000");

        socket.on('connection');

        // socket.emit('output'); if(socket!=undefined)
        {
            socket.on('output',(data)=>{ if(data.length)
                {
                    for(var x=0;x<data.length;x++)
                    {
```

```
                    var message = document.createElement("div");
                    message.setAttribute("class","chat-message");
                    message.textContent = data[x].message;
                    messages.appendChild(message);

messages.insertBefore(message,messages.firstChild);
                }
            }
        });

        sendBtn.addEventListener("click",()=>{

            socket.emit('input',{ message:
                textarea.value
            });

                                                textarea.value = "";
                                                event.preventDefault
                                                ();
                                            });



        clearBtn.addEventListener("click",()=>{
            socket.emit("clear");
        });

        socket.on("cleared",()=>{
            messages.textContent = "";
        });
    }
```

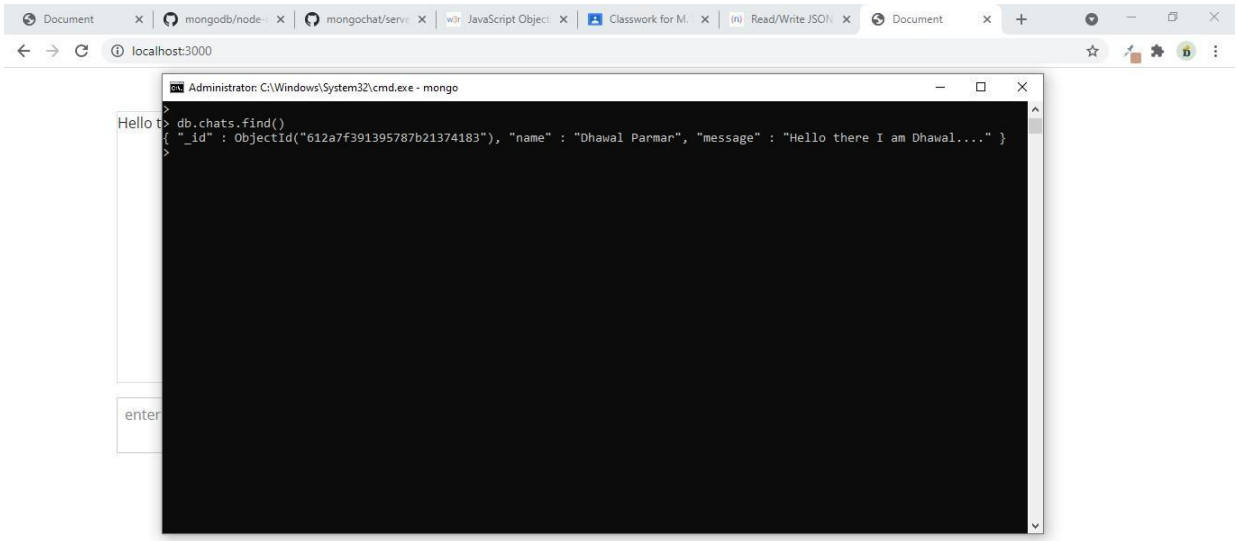
</script>

</html>


Document
mongodb/node
mongochat/serve
JavaScript Object
Classwork for M.
Read/Write JSON
Document
localhost:3000
Hello there I am Dhawal....
Send
Clear chats


Document
mongodb/node
mongochat/serve
JavaScript Object
Classwork for M.
Read/Write JSON
Document
localhost:3000
Hello there I am Dhawal....
enter message
Send
Clear chats


Type here to search
27°C
11:54 PM
8/28/2021


Document
mongodb/node-
mongochat/server
JavaScript Object
Classwork for M
Read/Write JSON
Document
localhost:3000
Hello t
enter
Administrator: C:\Windows\System32\cmd.exe - mongo
> db.chats.find()
{ "_id" : ObjectId("612a7f391395787b21374183"), "name" : "Dhawal Parmar", "message" : "Hello there I am Dhawal...." }
>