Real Time Applications With Node.js

Hazırlayan: Enes DURMUŞ

I. What Does A Real-Time App Do?

- Real-time apps perform many functions within a period and the user feels that it is occurring in real-time or instantly.

II. Where Are Real-Time Apps Used?

- Community storage solutions, VoIP (voice over internet protocol), some eCommerce transactions, instant messaging, chatting, online gaming, and video conference apps are the best instances of real-time apps.

III. Role of Node.js in Real-Time Apps

- The applications where speed is important and scalability is something that requires continuous focus, Node.js helps with its event-driven features and non-blocking I/O.
- Node.js offers constant two-way connections to applications like forums, social media, stock exchange software, and ad servers. For real-time, data-intensive apps and IoT devices, Node.js is considered the technology of choice as it is scalable and quick also.

IV. Advantages of Node.js in Real-Time Application Development

- **Event-Based Server**

  Real-time applications deal with many real-time users. Node.js development supports response depending on the event-driven server that assists in non-blocking functioning.

- **Data Sync**

  A Node.js developer makes the proper use of the non-blocking I/O feature.

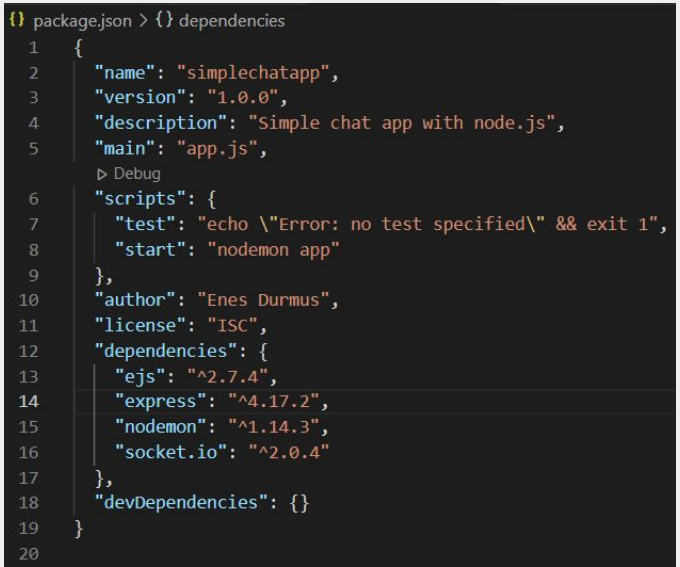
IV. Advantages of Node.js in Real-Time Application Development

- **Scalable and Fast**

  Since Node is a JavaScript-based program, it pulls the application faster like JS. Therefore, an application with the single-threaded model and the event loop can deal with several client requests easily.

Init A Project For A Simple Chat App

When we init our program and import dependencies our package.json look like this. For chat app we need express which is backend framework for node.js, socket.io, ejs which is Template engine helps to create an HTML template with minimal code, and nodemon.



Create app.js

When program runs firstly app.js executed.

We make necessary configurations.

app.js > io.on('connection') callback

```
1   const express = require('express')
2   const app = express()
3
4   //set the template engine ejs
5   app.set('view engine', 'ejs')
6
7   //middlewares
8   app.use(express.static('public'))
9
10  //routes
11  app.get('/', (req, res) => {
12      res.render('index')
13  })
14
15  //Listen on port 3000
16  server = app.listen(3000)
17
18  //socket.io instantiation
19  const io = require("socket.io")(server)
20
21
22  //listen on every connection
23  io.on('connection', (socket) => {
24      console.log('New user connected')
25
26      //default username
27      socket.username = "Anonymous"
28
29      //listen on change_username
30      socket.on('set_username', (data) => {
31          socket.username = data.username
32      })
```

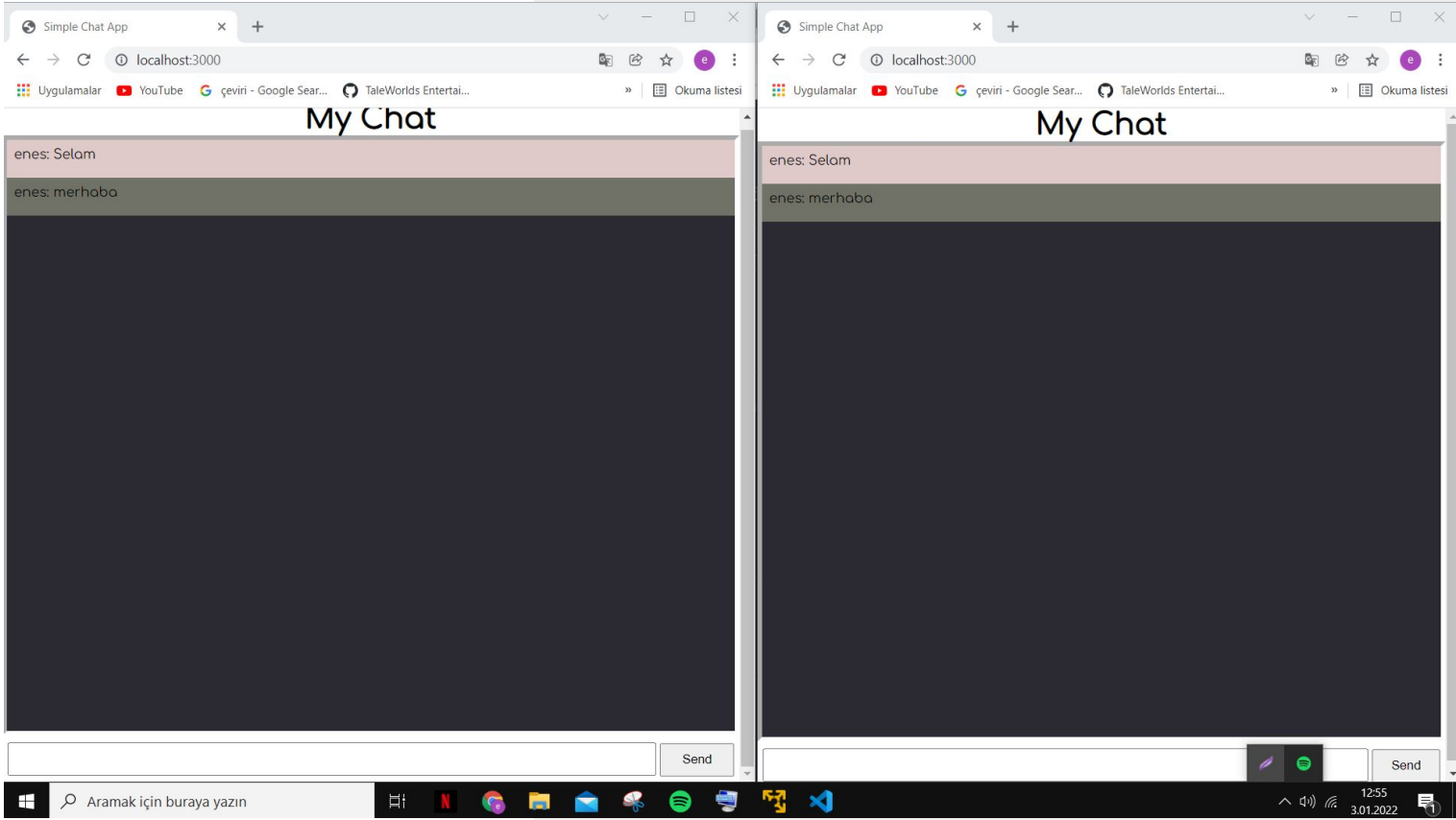
Create client.js and index.ejs

Index.ejs will be the page that users see. And client.js will handle the send and receive operations to server.

views > index.ejs > html > body

```
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" const="text/html;charset=UTF-8" />
    <link href="http://fonts.googleapis.com/css?family=Comfortaa" rel="stylesheet" type="text/css">
    <link rel="stylesheet" type="text/css" href="style.css" >
    <script src="https://cdnjs.cloudflare.com/ajax/libs/socket.io/2.0.4/socket.io.js"></script>
    <title>Simple Chat App</title>
  </head>

  <body>
    <header>
      <h1>My Chat</h1>
    </header>

    <section id="chatroom">
      <section id="feedback"></section>
    </section>

    <section id="input_container">
      <input id="message" class="vertical-align" type="text" />
      <button id="send_message" class="vertical-align" type="button">Send</button>
    </section>

    <script src="http://code.jquery.com/jquery-latest.min.js"></script>
    <script src="chat.js"></script>
  </body>
</html>

```

Result


Simple Chat App
localhost:3000
Uygulamalar
YouTube
çeviri - Google Sear...
TaleWorlds Entertai...
Okuma listesi
My Chat
enes: Selam
enes: merhaba
Send
Aramak için buraya yazın
12:55
3.01.2022