Websocket

Gold - Chapter 8 - Topic 1

Selamat datang di Chapter 8 Topik 1 online course Fullstack Web dari Binar Academy!

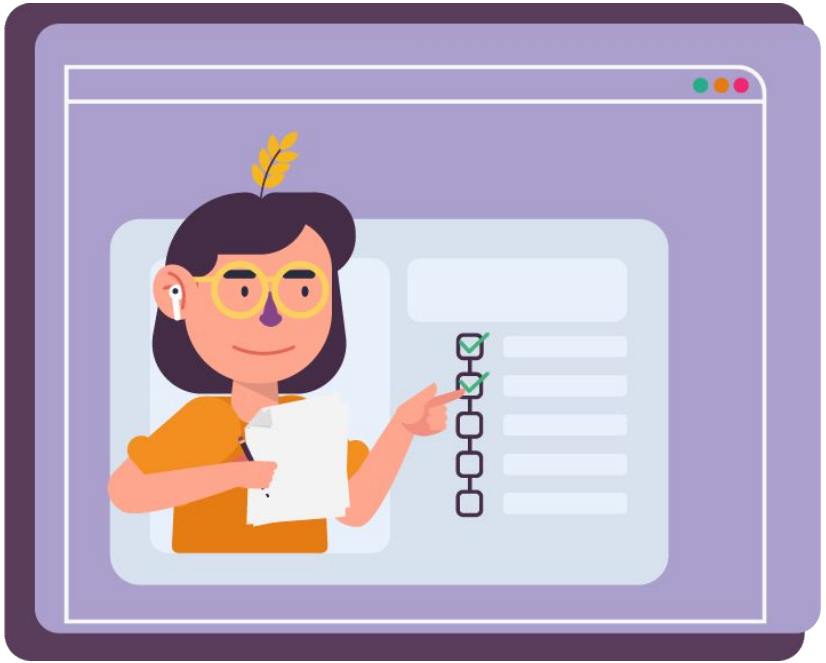
Sebagai pembuka course, chapter 8 bakal ngajak kamu buat mempelajari cara untuk melakukan **unit testing dan deployment dan beberapa tools yang bisa melengkapi fitur web aplikasi yang kamu kembangkan.** Mulai dari pengenalan websocket, Next.Js, media handling, EsLint, , unit testing + TDD, dan deployment + CI/CD.

Di topik pertama ini, kita bakal bahas dulu tentang **konsep, fungsi dan contoh penggunaan websocket dengan menggunakan Socket IO.**

Mari kita lanjut!

Detailnya, kita bakal bahas hal-hal berikut ini:

- Pengenalan konsep Websocket dan fungsinya
- Memahami contoh implementasi Socket IO

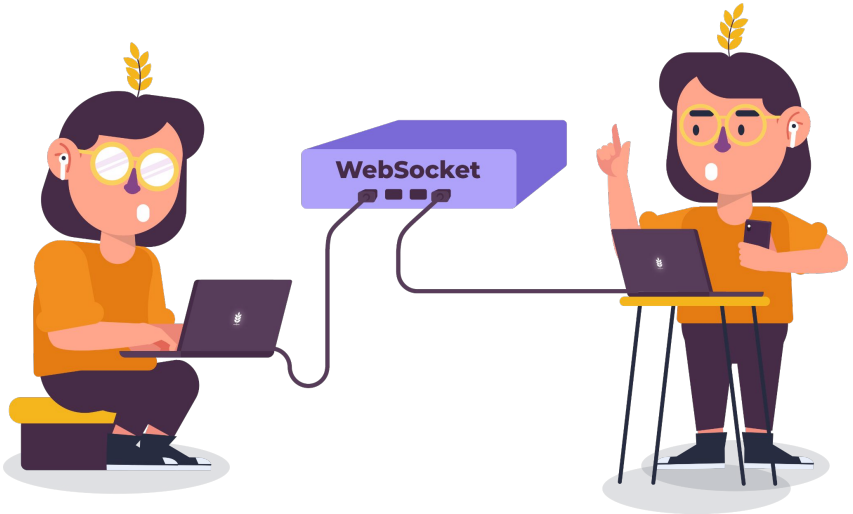


Kayak yang udah di-mention Sabrina sebelumnya, kita bakal bahas tentang websocket. Sebelum mulai, kamu sudah tahu belum apa sih websocket itu?

Hehe... Don’t worry be happy guys! Kita akan belajar materi ini dari awal kok, tenang aja. Markicus ~

Apa sih Websocket itu?

Websocket adalah **protokol komunikasi** yang pada dasarnya menggunakan TCP. Websocket memiliki kegunaan **untuk menyambungkan 2 atau lebih device sehingga bisa berkomunikasi secara real-time.**



Websocket ini berbeda dengan HTTP biasa lho~

Kalau HTTP hanya bisa meng-handle 2 device dalam 1 waktu (Client & Server) dan HTTP juga hanya bisa meng-handle unary request (request-response).

Sedangkan websocket, **bisa meng-handle lebih dari 2 device dalam 1 waktu** untuk berkomunikasi dan juga **memperbolehkan semua device untuk melakukan request, atau memberi response.** Jadi, semua yang terhubung ke websocket bisa kontribusi di dalam koneksi tersebut. Untuk lebih jelasnya kita akan bahas ini di poin berikutnya, setelah kita tau apa itu websocket.

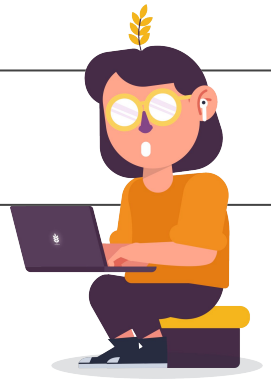
Cara kerja websocket bisa kita analogikan kayak kamu lagi nongkrong ke coffee shop bareng circle kamu. Pastinya semua orang di circle itu berkontribusi dalam obrolan tongkrongan. Semua orang di circle ga akan hanya jadi pendengar aja, tapi mereka juga bisa jadi orang yang akan menceritakan sesuatu.

Nah coffee shop ini memiliki fungsi untuk mempertemukan kalian, dan memfasilitasi tempat untuk nongkrong.

Kalo kita kaitin sama websocket, coffee shop ini kayak **websocket server, jadi dia akan menjadi tempat dimana satu client bertemu dengan client lainnya.** Kamu dan temen-temen kamu adalah client-client dari websocket server tersebut.

Websocket vs HTTP

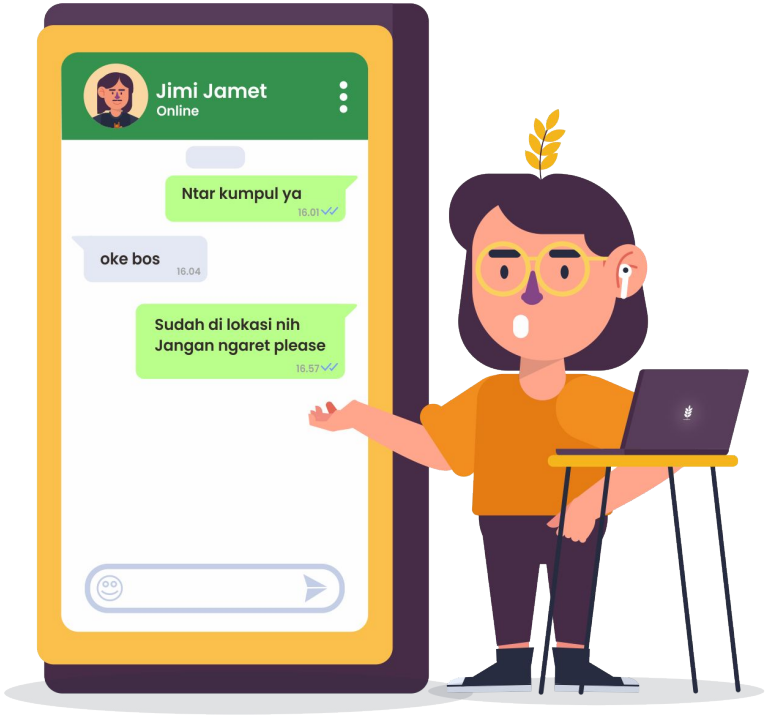
	Websocket	HTTP
Inisiasi komunikasi	• Client request dan server akan memberi respons • Server respons tanpa request dari client	Client request dan server akan memberi respons
Arah komunikasi	Dua arah	Satu arah



Hhmm.. sedikit kebayang tentang konsep websocket, terus kegunaan websocket itu apa aja sih?

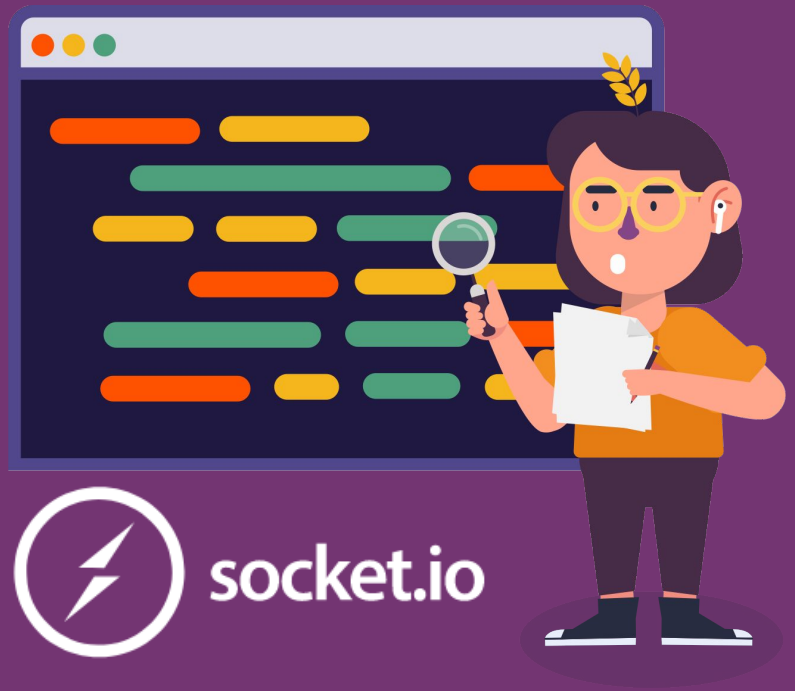
Nah karena pola komunikasi pake websocket ini bersifat realtime, banyak use case yang bisa dipecahin pake websocket sebagai contoh :

- Aplikasi chatting
- Status online/ offline dari user
- Status user lagi typing atau enggak
- Game online
- Live dashboard, dan sebagainya.



Oke, kamu kan udah tahu nih websocket itu apa, sekarang kita coba praktekin yuk! Tapi kita harus praktek dimana ya?

Nah ada library yang bakal bantuin kita dalam mengimplementasikan websocket lho, namanya adalah socket.io. Yuk kita eksplor bareng-bareng!



Socket.io

(Bidirectional and low-latency communication for every platform)

Socket.io adalah sebuah **library yang berfungsi untuk membuat realtime communication jauh lebih mudah.**

Socket.io ini digunakan di dua tempat, yaitu client dan server. Jadi kalo kamu mencoba socket.io pastikan kamu menuliskan keyword spesifik mau implementasi yang mana yah!

Kalo kamu mau cari tau lebih lanjut tentang socket.io, bisa coba cek **link ini**.

Nah, karena socket.io ini ada di Front-End dan Back-End, kita akan bahas satu per satu cara instalasinya seperti apa. Kita akan mulai dulu dari pembahasan dari sisi **Back-End** nya ya!

Socket.io di Back-end

Instalasi

Untuk melakukan instalasi socket.io kita perlu tahu target aplikasi kita terlebih dahulu. Karena kita disini belajar Full-stack web development otomatis kita perlu tahu implementasi socket.io di frontend maupun backend.

Nah kali ini kita akan mulai membahas dari backend terlebih dahulu.

Untuk instalasi di backend kamu hanya perlu **setup express seperti biasanya dan instal package socket.io.**

Membuat aplikasi Express sederhana

Karena socket.io (websocket) ini pasti membutuhkan server, maka dari itu kita perlu menyediakan 1 server yang akan digunakan untuk menjalankan websocket.

```
const express = require("express");
const app = express();

app.get("/", (req, res) => {
  res.status(200).json({
    status: "OK",
    message: "Hello World",
  });
})

app.listen(8000, () => {
  console.log("Listening on port 8000");
})
```

Inject socket.io di dalam aplikasi express!

Nah setelah kamu punya aplikasi express, sekarang saatnya kamu coba melakukan inject socket.io di dalamnya.

Di dalam callback **io.on("connection")**, terdapat fungsi yang berfungsi untuk meng-handle koneksi yang masuk ke websocket, tiap koneksi ini memungkinkan dia untuk mengirim pesan dan aktivitas. Nah kode di samping adalah contoh implementasi dari chat room. Jadi ketika client mengirim pesan dengan event **chat message** maka server akan melakukan broadcast ke channel tersebut dan semua client akan mendapatkannya.

Repo dari implementasi express socket.io di back-end bisa kalian lihat disini.

```
const http = require('http');
const { Server } = require("socket.io");
const app = require("./app")

const server = http.createServer(app);
const io = new Server(server);

io.on("connection", (socket) => {
  console.log("INFO:", "seseorang telah bergabung ke chat room!");

  socket.on('chat message', (msg) => {
    io.emit('incoming message', msg);
  });

  socket.on("disconnect", () => {
    console.log("INFO:", "seseorang telah pergi dari chat room!")
  })
})

server.listen(8000, () => {
  console.log("INFO:", "Listening on port 8000");
})
```

Sekarang kita udah punya back-end yang bisa dipakai untuk meng-handle implementasi chat room, sekarang kita coba cari tahu nih implementasinya di Front-End kayak apa ya kira–kira?

Instalasi Socket.io di Front-End

Nah untuk melakukan instalasi socket.io di dalam front-end, kamu harus punya project react dulu. Mudah lah ya, kita tinggal jalanin **CRA**.

Setelah kamu punya project react, install **socket.io-client** karena react adalah client dari sebuah socket.io dengan :

```
yarn add socket.io-client
```

Lalu modifikasi file App.js seperti code di samping.

```
import logo from "./logo.svg";
import io from "socket.io-client";
import "./App.css";

const socket = io.connect(process.env.REACT_APP_BACKEND_URL);

function App() {
  return (
    <div className="App">
      <img className="App-logo" src={logo} alt="react logo" />
      <div className="App-messages">
        {messages.map((message, index) => (
          <div className="App-message" key={index}>
            {message}
          </div>
        ))}
      </div>
      <form className="App-control">
        <input
          type="text"
          placeholder="Message ... "
        />
        <input className="App-button" type="submit" value="Send" />
      </form>
    </div>
  );
}

export default App;
```

```
function App() {
  const [message, setMessage] = useState("");
  const [messages, setMessages] = useState([]);

  function handleTextChange(e) {
    setMessage(e.target.value);
    console.log(message);
  }

  function handleSubmit(e) {
    e.preventDefault();
    if (!message || message === "") return;
    console.log("Submitted!");

    socket.emit("chat message", message);
  }

  useEffect(() => {
    socket.on("incoming message", (message) => {
      setMessages([...messages, message]);
    });
  }, [socket, messages]);

  ...
}
```

Lalu pasang state dan event handler di dalamnya

Disini kita memasang event handler ketika user melakukan submit form chat. Dan didalamnya kita akan melakukan emit event bernama chat message, agar diterima oleh server dan melakukan broadcast ke client yang lain.

Repo dari implementasi socket.io di back-end bisa kalian lihat disini.

Referensi

- socket.io
- https://www.youtube.com/watch?v=djMy4QsPWiI&t=1751s

Terima Kasih!

Next Topic

loading...

BINAR