**Antrian multi poli**

**Struktur Direktori yang Diperbarui**

plaintext

Salin kode

poli-antrian/

├── node\_modules/

├── src/

│ ├── controllers/

│ │ └── poliController.js

│ ├── models/

│ │ └── poli.js

│ ├── routes/

│ │ ├── adminRoutes.js

│ │ ├── poliRoutes.js

│ │ └── queueRoutes.js

│ ├── views/

│ │ ├── addPoli.html

│ │ ├── patient.html

│ │ └── poli.html

│ ├── public/

│ │ └── styles.css

│ └── server.js

├── .env

├── package.json

└── README.md

**File .env**

plaintext

Salin kode

# Konfigurasi MongoDB

MONGODB\_URI=mongodb://localhost:27017/poli-antrian

# Port server

PORT=3000

# Kunci rahasia untuk pengaturan lainnya

SECRET\_KEY=secretpass123

# Kunci API untuk layanan eksternal (jika diperlukan)

API\_KEY=apikey123

**File server.js**

javascript

Salin kode

// src/server.js

const express = require('express');

const mongoose = require('mongoose');

const bodyParser = require('body-parser');

const socketio = require('socket.io');

const http = require('http');

const path = require('path');

require('dotenv').config();

const adminRoutes = require('./routes/adminRoutes');

const poliRoutes = require('./routes/poliRoutes');

const queueRoutes = require('./routes/queueRoutes');

const app = express();

const server = http.createServer(app);

const io = socketio(server);

// Middleware

app.use(bodyParser.json());

app.use(bodyParser.urlencoded({ extended: true }));

// Menyediakan file statis dari folder public

app.use(express.static(path.join(\_\_dirname, 'public')));

// MongoDB connection

mongoose.connect(process.env.MONGODB\_URI, {

useNewUrlParser: true,

useUnifiedTopology: true

})

.then(() => {

console.log('Connected to MongoDB');

})

.catch(err => {

console.error('MongoDB connection error:', err.message);

process.exit(1); // Exit the process on connection error

});

// Socket.io connection

app.set('io', io);

// Routes

app.use('/api/admin', adminRoutes);

app.use('/api/poli', poliRoutes);

app.use('/api/queue', queueRoutes);

// Rute untuk mengarahkan ke halaman HTML di views

app.get('/patient', (req, res) => {

res.sendFile(path.join(\_\_dirname, 'views', 'patient.html'));

});

app.get('/poli', (req, res) => {

res.sendFile(path.join(\_\_dirname, 'views', 'poli.html'));

});

app.get('/addPoli', (req, res) => {

res.sendFile(path.join(\_\_dirname, 'views', 'addPoli.html'));

});

// Server start

const PORT = process.env.PORT || 3000;

server.listen(PORT, () => {

console.log(`Server running on port ${PORT}`);

});

**File models/poli.js**

javascript

Salin kode

// src/models/poli.js

const mongoose = require('mongoose');

const PoliSchema = new mongoose.Schema({

name: { type: String, required: true, unique: true }

});

const Poli = mongoose.model('Poli', PoliSchema);

module.exports = Poli;

**File controllers/poliController.js**

javascript

Salin kode

// src/controllers/poliController.js

const Poli = require('../models/poli');

exports.getAllPolis = async (req, res) => {

try {

const polis = await Poli.find();

res.json(polis);

} catch (err) {

res.status(500).json({ message: err.message });

}

};

exports.createPoli = async (req, res) => {

const { name } = req.body;

const poli = new Poli({ name });

try {

const newPoli = await poli.save();

res.status(201).json(newPoli);

} catch (err) {

res.status(400).json({ message: err.message });

}

};

**File routes/poliRoutes.js**

javascript

Salin kode

// src/routes/poliRoutes.js

const express = require('express');

const router = express.Router();

const poliController = require('../controllers/poliController');

router.get('/', poliController.getAllPolis);

router.post('/', poliController.createPoli);

module.exports = router;

**File routes/queueRoutes.js**

javascript

Salin kode

// src/routes/queueRoutes.js

const express = require('express');

const router = express.Router();

const queueController = require('../controllers/queueController');

router.get('/', queueController.getAllQueues);

router.get('/:poliName', queueController.getQueueByPoli);

router.post('/next', queueController.incrementQueue);

router.post('/previous', queueController.decrementQueue);

module.exports = router;

**File views/addPoli.html**

html

Salin kode

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Tambah Poli</title>

<link rel="stylesheet" href="/styles.css">

</head>

<body>

<h1>Tambah Poli</h1>

<form id="addPoliForm">

<label for="poliName">Nama Poli:</label>

<input type="text" id="poliName" name="poliName" required>

<button type="submit">Tambah</button>

</form>

<script>

document.getElementById('addPoliForm').addEventListener('submit', async (event) => {

event.preventDefault();

const poliName = document.getElementById('poliName').value;

try {

const response = await fetch('/api/poli', {

method: 'POST',

headers: { 'Content-Type': 'application/json' },

body: JSON.stringify({ name: poliName })

});

if (!response.ok) {

throw new Error('Failed to add poli');

}

alert('Poli berhasil ditambahkan');

document.getElementById('poliName').value = ''; // clear input field

} catch (error) {

console.error('Error adding poli:', error.message);

alert('Error adding poli');

}

});

</script>

</body>

</html>

**File views/patient.html**

html

Salin kode

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Pasien</title>

<link rel="stylesheet" href="/styles.css">

<script src="/socket.io/socket.io.js"></script>

</head>

<body>

<h1>Pasien</h1>

<div id="queues"></div>

<script>

const socket = io();

async function fetchQueues() {

try {

const response = await fetch('/api/queue');

if (!response.ok) {

throw new Error('Failed to fetch queues');

}

const queues = await response.json();

const queueContainer = document.getElementById('queues');

queueContainer.innerHTML = '';

queues.forEach(queue => {

const queueDiv = document.createElement('div');

queueDiv.innerHTML = `<h2>${queue.poliName}</h2><h3>Nomor Antrian Saat Ini: <span>${queue.currentNumber}</span></h3>`;

queueContainer.appendChild(queueDiv);

});

} catch (error) {

console.error('Error fetching queues:', error.message);

}

}

socket.on('queueUpdated', () => {

fetchQueues();

});

fetchQueues();

</script>

</body>

</html>

**File views/poli.html**

html

Salin kode

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Petugas Poli</title>

<link rel="stylesheet" href="/styles.css">

<script src="/socket.io/socket.io.js"></script>

</head>

<body>

<h1>Petugas Poli</h1>

<div>

<label for="poliSelect">Pilih Poli:</label>

<select id="poliSelect"></select>

</div>

<div>

<h2 id="poliName">Poli</h2>

<h3>Nomor Antrian Saat Ini: <span id="currentNumber">0</span></h3>

<button onclick="previous()">Previous</button>

<button onclick="next()">Next</button>

</div>

<script>

const socket = io();

async function fetchPolis() {

try {

const response = await fetch('/api/poli');

if (!response.ok) {

throw new Error('Failed to fetch polis');

}

const polis = await response.json();

const poliSelect = document.getElementById('poliSelect');

poliSelect.innerHTML = '';

polis.forEach(poli => {

const option = document.createElement('option');

option.value = poli.name;

option.textContent = poli.name;

poliSelect.appendChild(option);

});

if (polis.length > 0) {

poliSelect.value = polis[0].name;

fetchQueue(polis[0].name);

}

} catch (error) {

console.error('Error fetching polis:', error.message);

}

}

async function fetchQueue(poliName) {

try {

const response = await fetch(`/api/queue/${poliName}`);

if (!response.ok) {

throw new Error('Failed to fetch queue');

}

const queue = await response.json();

document.getElementById('poliName').innerText = queue.poliName;

document.getElementById('currentNumber').innerText = queue.currentNumber;

} catch (error) {

console.error('Error fetching queue:', error.message);

}

}

async function next() {

const poliName = document.getElementById('poliSelect').value;

try {

const response = await fetch('/api/queue/next', {

method: 'POST',

headers: { 'Content-Type': 'application/json' },

body: JSON.stringify({ poliName })

});

if (!response.ok) {

throw new Error('Failed to update queue');

}

socket.emit('updateQueue');

fetchQueue(poliName);

} catch (error) {

console.error('Error updating queue:', error.message);

}

}

async function previous() {

const poliName = document.getElementById('poliSelect').value;

try {

const response = await fetch('/api/queue/previous', {

method: 'POST',

headers: { 'Content-Type': 'application/json' },

body: JSON.stringify({ poliName })

});

if (!response.ok) {

throw new Error('Failed to update queue');

}

socket.emit('updateQueue');

fetchQueue(poliName);

} catch (error) {

console.error('Error updating queue:', error.message);

}

}

socket.on('queueUpdated', () => {

const poliName = document.getElementById('poliSelect').value;

fetchQueue(poliName);

});

document.getElementById('poliSelect').addEventListener('change', (event) => {

fetchQueue(event.target.value);

});

fetchPolis();

</script>

</body>

</html>

**File routes/adminRoutes.js**

Kosongkan atau tambahkan sesuai kebutuhan jika diperlukan:

javascript

Salin kode

// src/routes/adminRoutes.js

const express = require('express');

const router = express.Router();

// Tambahkan rute admin di sini jika diperlukan

module.exports = router;

**File controllers/queueController.js**

javascript

Salin kode

// src/controllers/queueController.js

const Poli = require('../models/poli');

let queues = {};

exports.getAllQueues = (req, res) => {

res.json(Object.keys(queues).map(poliName => ({

poliName,

currentNumber: queues[poliName]

})));

};

exports.getQueueByPoli = (req, res) => {

const { poliName } = req.params;

res.json({ poliName, currentNumber: queues[poliName] || 0 });

};

exports.incrementQueue = (req, res) => {

const { poliName } = req.body;

if (!queues[poliName]) {

queues[poliName] = 0;

}

queues[poliName]++;

req.app.get('io').emit('queueUpdated');

res.json({ poliName, currentNumber: queues[poliName] });

};

exports.decrementQueue = (req, res) => {

const { poliName } = req.body;

if (!queues[poliName]) {

queues[poliName] = 0;

}

queues[poliName] = Math.max(queues[poliName] - 1, 0);

req.app.get('io').emit('queueUpdated');

res.json({ poliName, currentNumber: queues[poliName] });

};

**File public/styles.css**

css

Salin kode

/\* src/public/styles.css \*/

body {

font-family: Arial, sans-serif;

margin: 0;

padding: 0;

display: flex;

flex-direction: column;

align-items: center;

justify-content: center;

min-height: 100vh;

background-color: #f5f5f5;

}

h1 {

margin-bottom: 20px;

}

form, div {

background: #fff;

padding: 20px;

border-radius: 8px;

box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);

margin-bottom: 20px;

}

label, input {

display: block;

width: 100%;

margin-bottom: 10px;

}

button {

padding: 10px;

background: #007BFF;

border: none;

color: #fff;

cursor: pointer;

border-radius: 4px;

}

button:hover {

background: #0056b3;

}

**Menjalankan Server**

Pastikan untuk menjalankan server dengan:

sh

Salin kode

npm run dev

Kemudian akses halaman-halaman berikut di browser Anda:

* **Halaman Petugas Poli**: http://localhost:3000/poli
* **Halaman Pasien**: http://localhost:3000/patient
* **Halaman Tambah Poli**: http://localhost:3000/addPoli

Semua file yang diperlukan telah disertakan dan diatur dalam struktur direktori yang benar. Anda dapat menyesuaikan lebih lanjut sesuai dengan kebutuhan aplikasi Anda.