const express= require('express');

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

const app= express();

const users= [];

const songs= [];

const playlists= [];

app.use(bodyParser.json());

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

app.listen(3000, () => console.log('Server running on port 3000.'));

//1

app.post('/registerUser', (req, res) => {

let user= users.find(o => o.email\_user === req.body.email\_user);

if (user) {

res.status(400).json({

status: 'Fail',

message: 'Registrasi gagal! E-mail sudah terpakai sebelumnya.'

});

} else {

users.push({

id\_user: users.length+1,

email\_user: req.body.email\_user,

nama\_user: req.body.nama\_user,

password\_user: req.body.password\_user,

tipe\_user: parseInt(req.body.tipe\_user)

});

res.status(200).json({

status: 'Success',

message: `Registrasi user ${req.body.email\_user} berhasil!`,

registeredUser: users[users.length-1]

});

}

});

//2

app.post('/uploadSong', (req, res) => {

let user= users.find(o => o.id\_user === parseInt(req.body.id\_user));

let song= songs.find(o => o.judul\_lagu === req.body.judul\_lagu);

if (!user) {

res.status(400).json({

status: 'Fail',

message: 'Pengguna tidak ditemukan!'

});

} else {

if (user.tipe\_user !== 1) {

res.status(400).json({

status: 'Fail',

message: 'Pengguna tidak mempunyai hak untuk mengupload lagu.'

});

} else {

if (song) {

res.status(400).json({

status: 'Fail',

message: 'Judul lagu sudah ada sebelumnya.'

});

} else {

songs.push({

id\_lagu: !songs.length ? songs.length+1 : songs[songs.length-1].id\_lagu+1,

id\_artist: parseInt(req.body.id\_user),

judul\_lagu: req.body.judul\_lagu,

genre\_lagu: req.body.genre\_lagu,

durasi\_lagu: parseInt(req.body.durasi\_lagu)

});

res.status(200).json({

status: 'Success',

message: 'Upload lagu berhasil!',

uploadedSong: songs[songs.length-1]

});

}

}

}

});

//3

app.post('/createPlaylist', (req, res) => {

req.body.collection\_lagu= JSON.parse(`[ ${req.body.collection\_lagu} ]`).map(id => parseInt(id)).sort();

let user= users.find(o => o.email\_user === req.body.email\_user);

let song= req.body.collection\_lagu.every(id => songs.map(o => o.id\_lagu).indexOf(id) > -1);

if (!user) {

res.status(400).json({

status: 'Fail',

message: 'E-mail pengguna tidak ditemukan!'

});

} else {

/\*if (user.tipe\_user !== 0) {

res.status(400).json({

status: 'Fail',

message: 'Pengguna tidak mempunyai hak untuk membuat playlist.'

});

} else {

}\*/

if (!song) {

res.status(400).json({

status: 'Fail',

message: 'Lagu tidak ditemukan!'

});

} else {

playlists.push({

id\_playlist: playlists.length+1,

email\_user: req.body.email\_user,

judul\_playlist: req.body.judul\_playlist,

collection\_lagu: req.body.collection\_lagu

});

res.status(200).json({

status: 'Success',

message: 'Create playlist berhasil!',

createdPlaylist: playlists[playlists.length-1]

});

}

}

});

//4

app.get('/searchSongByKeyword/:keyword', (req, res) => {

let song= songs.filter(o =>

o.judul\_lagu.includes(req.params.keyword) ||

o.judul\_lagu.toLowerCase().includes(req.params.keyword) ||

o.judul\_lagu.toUpperCase().includes(req.params.keyword)

);

if (!song.length) {

res.status(400).json({

status: 'Fail',

message: 'Lagu tidak ditemukan!'

});

} else {

res.status(200).json({

status: 'Success',

message: song

});

}

});

//5

app.delete('/deleteSong/:judul\_lagu', (req, res) => {

let song= songs.find(o => o.judul\_lagu === req.params.judul\_lagu);

if (!song) {

res.status(400).json({

status: 'Fail',

message: 'Judul lagu tidak ditemukan!'

});

} else {

let index= songs.indexOf(song);

songs.splice(index, 1);

res.status(200).json({

status: 'Success',

message: 'Lagu berhasil di hapus!',

deletedSong: song

});

}

});

//6

app.get('/getPlaylist/:email\_user', (req, res) => {

let requestedPlaylists= playlists.filter(o => o.email\_user === req.params.email\_user);

if (!requestedPlaylists.length) {

res.status(400).json({

status: 'Fail',

message: 'Playlist tidak ditemukan!'

});

} else {

res.status(200).json({

status: 'Success',

requestedPlaylists: requestedPlaylists.map(o => ({

id\_playlist: o.id\_playlist,

judul\_playlist: o.judul\_playlist,

collection\_lagu: o.collection\_lagu

}))

});

}

});

//7

app.put('/updatePlaylist/:id\_playlist', (req, res) => {

let playlist= playlists.find(o => o.id\_playlist === parseInt(req.params.id\_playlist));

if (!playlist) {

res.status(400).json({

status: 'Fail',

message: 'ID playlist tidak ditemukan!'

});

} else {

playlist.judul\_playlist= req.body.judul\_playlist;

res.status(200).json({

status: 'Success',

message: 'Update playlist berhasil!',

updatedPlaylist: playlist

});

}

});