***Source Code***

1. **Autentikasi *Login***

const { Users } = require("../../models");

const Joi = require("joi");

const bcrypt = require("bcrypt");

const jwt = require("jsonwebtoken");

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

const schema = Joi.object({

username: Joi.string().required(),

password: Joi.string().min(6).required(),

});

const { error } = schema.validate(req.body);

if (error)

return res.status(400).send({

error: {

message: error.details[0].message,

},

});

try {

const userExist = await Users.findOne({

where: {

username: req.body.username,

},

attributes: {

exclude: ["createdAt", "updatedAt"],

},

});

if (!userExist) {

return res.status(400).send({

status: "failed",

message: "unregistered account",

});

}

const isValid = await bcrypt.compare(req.body.password, userExist.password);

if (!isValid) {

return res.status(400).send({

status: "failed",

message: "password invalid",

});

}

const token = jwt.sign({ id: userExist.id }, process.env.SECRET\_KEY);

return res.status(200).send({

status: "success",

message: "Login Success",

data: {

user: {

id: userExist.id,

username: userExist.username,

level: userExist.level,

token,

},

},

});

} catch (error) {

console.log(error);

return res.status(500).send({

status: "failed",

message: "Server Error",

});

}

};

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

try {

const id = req.user.id;

console.log("i", id);

const dataUser = await Users.findOne({

where: {

id,

},

attributes: {

exclude: ["createdAt", "updatedAt", "password"],

},

});

if (!dataUser) {

return res.status(404).send({

status: "Failed!! error data user",

});

}

res.send({

status: "success",

});

} catch (error) {

res.status(500).send({

status: "failed",

message: "Server Error",

});

}

};

1. **Berita**

const { response } = require("express");

const { Berita } = require("../../models");

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

try {

const image = req.file ? req.file.filename : null;

const { tittle, content\_berita, tgl } = req.body;

if (!tittle || !content\_berita || !tgl || !image) {

return res.status(400).json({ message: "Semua field harus di isi" });

}

const berita = await Berita.create({

tittle,

content\_berita,

tgl,

image,

});

res.status(201).json({

message: "Data Berita berhasil ditambahkan",

data: berita,

});

} catch (error) {

console.log(error);

res.status(500).json({

message: "Data berita gagal ditambahkan karena kesalahan dari Server",

error: error.message,

});

}

};

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

try {

const { id } = req.params;

const { tittle, content\_berita, tgl } = req.body;

const berita = Berita.findByPk(id);

if (!berita) {

return res.status(400).json({ message: "Id Tidak ditemukan" });

}

let image = req.file;

if (req.file) {

image = req.file.filename;

}

const dataUpdate = await Berita.update(

{

tittle: tittle,

content\_berita: content\_berita,

tgl: tgl,

image: image,

},

{

where: { id: id },

}

);

res.status(201).json({

message: "Data Berita Berhasil update",

data: dataUpdate,

});

} catch (error) {

console.log(error);

res.status(500).json({

message: "Data berita gagal diupdate karena kesalahan dari Server",

error: error.message,

});

}

};

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

try {

const { id } = req.params;

const berita = await Berita.findByPk(id);

if (!berita || berita.length === 0 || JSON.stringify(berita) === "[]") {

return res.status(201).json({

success: true,

message: `data dengan id ${id} tidak ada`,

berita: [],

});

}

const newBerita = berita;

await newBerita.destroy();

return res.status(200).json({ message: "Data Berita berhasil dihapus" });

} catch (error) {

res.status(500).json({ message: "Kesalahan pada server" });

}

};

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

try {

const berita = await Berita.findAll();

if (!berita || berita.length === 0 || JSON.stringify(berita) === "[]") {

return res.status(200).json({

success: true,

message: "Data Berita Belum Tersedia",

data: [],

});

}

return res.status(200).json({

success: true,

message: "Data Berita berhasil di get",

data: berita,

});

} catch (error) {

console.log(error);

return res.status(500).json({

message: "Error saat mencari data pada server",

});

}

};

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

try {

const { id } = req.params;

const berita = await Berita.findByPk(id);

if (!berita || berita.length === 0 || JSON.stringify(berita) === "[]") {

return res.status(200).json({

success: true,

message: `Data berita by id ${id} = ${berita}`,

data: [],

});

}

const newBeritaId = berita;

return res.status(200).json({

success: true,

data: newBeritaId,

message: `Data berita by id = ${id}`,

});

} catch (error) {

return res.status(500).json({

success: false,

message: `error get data by id ${id} karena masalah pada server`,

});

}

};

1. **Profil**

const e = require("express");

const { Profil } = require("../../models");

const path = require("path");

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

try {

const image = req.file ? req.file.filename : null;

const { tittle\_profil, content\_profil, tgl } = req.body;

if (!tittle\_profil || !content\_profil || !tgl || !image) {

return res.status(400).json({ message: "Semua field harus di isi" });

}

const profil = await Profil.create(

{

tittle\_profil,

content\_profil,

tgl,

image,

},

{ returning: true }

);

res.status(201).json({

message: ` Data Profil berhasil ditambahkan`,

data: profil,

});

} catch (error) {

console.log(error);

res.status(500).json({ message: "Server error", error: error.message });

}

};

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

try {

const { id } = req.params;

const { tittle\_profil, content\_profil, tgl } = req.body;

const profil = await Profil.findByPk(id);

if (!profil) {

return res.status(404).json({ message: "Profil tidak ditemukan" });

}

let image = profil.image;

if (req.file) {

image = req.file.filename;

}

await Profil.update(

{

tittle\_profil,

content\_profil,

tgl,

image,

},

{

where: { id },

}

);

res.status(200).json({ message: "Profil berhasil diperbarui" });

} catch (error) {

console.error(error);

res

.status(500)

.json({ message: "Terjadi kesalahan pada server saat update berita " });

}

};

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

try {

const { id } = req.params;

const profil = await Profil.findByPk(id);

if (!profil || profil.length === 0 || JSON.stringify(profil) === "[]") {

return res.status(201).json({

success: true,

message: `Profil tidak ditemukan ${id}`,

profil: [],

});

}

const newDeleteProfil = profil;

await newDeleteProfil.destroy();

return res.status(200).json({ message: "Data Berita berhasil dihapus" });

} catch (error) {

res.status(500).json({ message: "Kesalahan pada server" });

}

};

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

try {

const profil = await Profil.findAll();

if (!profil || profil.length === 0 || JSON.stringify(profil) === "[]") {

return res.status(200).json({

success: true,

message: "Data Profil tidak ditemukan",

data: [],

});

}

res.status(200).json({

success: true,

message: "Data profil berhasil ditemukan!",

data: profil,

});

} catch (error) {

console.log(error);

res.status(500).json({ message: "data tidak berhasil dicari" });

}

};

1. **Pelapor**

const { Pelapor } = require("../../models");

const { path } = require("path");

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

try {

const {

nama\_pelapor,

jenis\_identitas,

no\_identitas,

alamat\_pelapor,

no\_hp\_pelapor,

email,

unit\_kerja,

kategori\_pelapor,

status\_pelapor,

tgl\_peristiwa,

lokasi\_peristiwa,

kronologi\_peristiwa,

nama\_terlapor,

no\_hp\_terlapor,

status\_terlapor,

} = req.body;

const userId = req.user ? req.user.id : null;

// Mengambil file yang diupload

const file\_identitas = req.files["file\_identitas"]

? req.files["file\_identitas"][0].path

: null;

const bukti\_peristiwa = req.files["bukti\_peristiwa"]

? req.files["bukti\_peristiwa"][0].path

: null;

// Menyimpan data ke database

const newPelapor = await Pelapor.create({

nama\_pelapor,

jenis\_identitas,

no\_identitas,

file\_identitas,

alamat\_pelapor,

no\_hp\_pelapor,

email,

unit\_kerja,

kategori\_pelapor,

status\_pelapor,

tgl\_peristiwa,

lokasi\_peristiwa,

kronologi\_peristiwa,

bukti\_peristiwa,

nama\_terlapor,

no\_hp\_terlapor,

status\_terlapor,

created\_by: userId,

});

res.status(201).json({

message: "Data Pelapor Berhasil ditambahkan",

data: newPelapor,

});

} catch (error) {

console.log(error);

res.status(500).json({

message: "Data Pelapor gagal ditambahkan karena masalah pada server",

});

}

};

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

try {

const { id } = req.params;

const pelapor = Pelapor.findByPk(id);

if (!pelapor) {

res.status(400).json({

message: "data tidak id ditemukan",

});

}

if (

!req.files ||

!req.files["file\_identitas"] ||

!req.files["bukti\_peristiwa"]

) {

return res.status(400).json({ message: "File tidak ditemukan!" });

}

// const fileIdentitas = req.files["file\_identitas"][0].path;

// const buktiPeristiwa = req.files["bukti\_peristiwa"][0].path;

const fileIdentitas =

req.files && req.files["file\_identitas"]

? req.files["file\_identitas"][0].path

: pelapor.file\_identitas;

const buktiPeristiwa =

req.files && req.files["bukti\_peristiwa"]

? req.files["bukti\_peristiwa"][0].path

: pelapor.bukti\_peristiwa;

// const file\_identitas = req.files["file\_identitas"]

// ? req.files["file\_identitas"][0].path

// : pelapor.file\_identitas;

// const bukti\_peristiwa = req.files["bukti\_peristiwa"]

// ? req.files["bukti\_peristiwa"][0].path

// : pelapor.bukti\_peristiwa;

const datauntukUpdate = {

nama\_pelapor: req.body.nama\_pelapor || pelapor.nama\_pelapor,

jenis\_identitas: req.body.jenis\_identitas || pelapor.jenis\_identitas,

no\_identitas: req.body.no\_identitas || pelapor.no\_identitas,

file\_identitas: fileIdentitas,

alamat\_pelapor: req.body.alamat\_pelapor || pelapor.alamat\_pelapor,

no\_hp\_pelapor: req.body.no\_hp\_pelapor || pelapor.no\_hp\_pelapor,

email: req.body.email || pelapor.email,

unit\_kerja: req.body.unit\_kerja || pelapor.unit\_kerja,

kategori\_pelapor: req.body.kategori\_pelapor || pelapor.kategori\_pelapor,

status\_pelapor: req.body.status\_pelapor || pelapor.status\_pelapor,

tgl\_peristiwa: req.body.tgl\_peristiwa || pelapor.tgl\_peristiwa,

lokasi\_peristiwa: req.body.lokasi\_peristiwa || pelapor.lokasi\_peristiwa,

bukti\_peristiwa: buktiPeristiwa,

kronologi\_peristiwa:

req.body.kronologi\_peristiwa || pelapor.kronologi\_peristiwa,

};

const dataPelapor = await Pelapor.update(datauntukUpdate, {

where: { id: req.params.id },

});

res.status(200).json({

message: "Data berhasil diupdate",

data: dataPelapor,

});

} catch (error) {

console.log(error);

res.status(500).json({

message: "Data gagal diupdate karena kesalahan dari server",

});

}

};

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

try {

const pelapor = await Pelapor.findAll();

if (!pelapor || pelapor.length === 0 || JSON.stringify(pelapor) === "[]") {

return res.status(200).json({

success: true,

message: "Data pelapor tidak ada",

data: [],

});

}

res.status(200).json({

success: true,

message: "Data terlapor berhasil ditemukan!",

data: pelapor,

});

} catch (error) {

console.log(error);

res.status(500).json({ message: "data tida berhasil dicari" });

}

};

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

try {

const { id } = req.params;

const pelapor = await Pelapor.findByPk(id);

if (!pelapor) {

return res.status(200).json({

success: true,

message: `get pelapor by id = ${id}`,

data: [],

});

}

// const newPelapor = pelapor;

return res.status(200).json({

success: true,

data: pelapor,

message: `data pelapor by id = ${id}`,

});

} catch (error) {

return res.status(500).json({

success: false,

message: `error get pelapor by ${id} karena masalah pada server`,

});

}

};

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

try {

const { id } = req.params;

// Cari data yang akan dihapus

const pelapor = await Pelapor.findByPk(id);

if (!pelapor) {

return res.status(404).json({ message: "Data Pelapor tidak ditemukan" });

}

// Hapus data

await pelapor.destroy();

res.status(200).json({ message: "Data Pelapor berhasil dihapus" });

} catch (error) {

console.log(error);

res.status(500).json({ message: "Kesalahan pada server" });

}

};