const TourModel = require('./../models/tourModel');

const APIFeatures = require('./../utils/apiFeatures');

//middleware

// exports.checkID = (req,res,next, val)=>{

// console.log(`Tour ID is: ${val}`);

// const id = req.params.id \* 1;

// const tour = tours.find(el => el.id === id)

// if(id > tours.length || !tour){

// // if(req.params.id \* 1 > tours.length){

// return res.status(404).json({

// status:'fail',

// message:'Invalid ID '

// })

// }

// next();

// }

// exports.checkBody = (req, res, next) =>{

// if(!req.body.name || !req.body.price){

// return res.status(400).json({

// status:'Fail',

// message: 'Missing Name or Price'

// })

// }

// next();

// }

exports.aliasTopTours = (req, res, next)=>{

req.query.limit = '5';

req.query.sort = '-ratingsAverage,price';

req.query.fields = 'name,price,ratingsAverage,summary,difficulty';

next();

};

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

try {

//EXECUTE THE QUERY

const features = new APIFeatures(TourModel.find(),req.query).filter().sort().limitFields().paginate();

const tours = await features.query;

//SEND RESPONSE

res.status(200).json({

status:'success',

results:tours.length,

data:{

tours

}

})

} catch (error) {

res.status(404).json({

status: 'Failed',

message: error.message

}) } }

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

try {

const tour = await TourModel.findById(req.params.id);

res.status(200).json({

status:'success',

data:{

tour

}

})

} catch (error) {

res.status(404).json({

status:'Fail',

message:error

})

}

}

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

try {

const newTour = await TourModel.create(req.body);

res.status(201).json({

status: 'success',

data:{

tour: newTour

}

})

} catch (error) {

res.status(400).json({

status:'Fail',

message:error.message

})

}

}

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

try {

const tour = await TourModel.findByIdAndUpdate(req.params.id, req.body,{

new:true,

runValidators:true

})

res.status(200).json({

status:'success',

data:{

tour

}

});

} catch (error) {

res.status(404).json({

status:'Fail',

message: error.message

}) } }

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

res.status(204).json({

status:'success deleted',

data:null

});

}

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

try {

const stats = await TourModel.aggregate([

{

$match:{

ratingsAverage: { $gte:4.5 }

}

},

{

$group:{

\_id:{ $toUpper: '$difficulty'},

// \_id:'$ratingsAverage',

num:{$sum:1},

numRatings:{$sum:'$ratingsQuantity'},

avgRating:{ $avg:'$ratingsAverage'},

avgPrice: {$avg: '$price'},

minPrice:{$min:'$price'},

maxPrice:{$max:'$price'},

}

},

{

$sort:{avgPrice:1}

}

// {

// // $match:{\_id: { $ne:'EASY'}} // can double the aggregation

// }

]);

res.status(200).json({

status:'success',

data:{

stats

}

});

} catch (error) {

res.status(404).json({

status:'Failed',

message:error.message

})

}

}

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

try {

const year = req.params.year \* 1; // 2021

const plan = await TourModel.aggregate([

{

$unwind: '$startDates'

},

{

$match:{

startDates:{

$gte: new Date(`${year}-01-01`),

$lte: new Date(`${year}-12-31`)

}

}

},

{

$group:{

\_id:{$month: '$startDates'},

numTourStarts: {$sum: 1},

tours:{$push: '$name'}

}

},

{

$addFields:{month: '$\_id'}

},

{

$project:{

\_id:0

}

},

{

$sort:{

numTourStarts:-1

}

},

{

$limit:12

}

]);

res.status(200).json({

status:'success',

result:plan.length,

data:{

plan

}

});

} catch (error) {

res.status(400).json({

status:'Failed',

message: error.message

})

}

}