ro'yxatdan o'tgan foydalanuvchi testni bir marta testni boshlash tugmasini bosganda test boshlani va qayta testni boshlash tugmasini bosganda message:"siz testni allaqachon boshlagansiz" deb testni qolganjoyidan davom etib ketsin

const getTestBoshlash = async (req, res, next) => {

try {

const { testId } = req.params;

// Testni bazadan topamiz

const test = await testModel.findById(testId);

if (!test) {

return next(new ErrorResponse("Test topilmadi", 404));

}

// Foydalanuvchi uchun natijalarni topamiz

const existingResult = await UserTestResult.findOne({

userId: req.user.\_id,

testId,

});

if (existingResult) {

// Agar foydalanuvchi allaqachon testni boshlagan bo'lsa

if (Date.now() > existingResult.endTime) {

return res.status(400).json({

message: "Siz testni allaqachon yakunlagansiz",

});

}

// Test davom ettiriladi

return res.status(200).json({

message: "Siz testni allaqachon boshlagansiz",

test: {

title: test.title,

duration: test.duration,

startTime: existingResult.startTime,

endTime: existingResult.endTime,

questions: test.questions.map((q) => ({

question: q.question,

options: q.options,

questionId: q.\_id,

})),

},

});

}

// Agar foydalanuvchi yangi test boshlayotgan bo'lsa

const startTime = Date.now();

const endTime = startTime + test.duration \* 60000;

// Yangi foydalanuvchi natijasini saqlaymiz

const result = new UserTestResult({

userId: req.user.\_id,

testId,

startTime,

endTime,

correctAnswers: 0,

totalQuestions: test.questions.length,

});

await result.save();

// Yangi test uchun ma'lumotlarni qaytaramiz

res.status(200).json({

message: "Test boshlandi",

test: {

title: test.title,

duration: test.duration,

startTime,

endTime,

questions: test.questions.map((q) => ({

question: q.question,

options: q.options,

questionId: q.\_id,

})),

},

});

} catch (error) {

next(new ErrorResponse(error.message, 500));

}

};