ПРИЛОЖЕНИЕ ВИсходный текст модуляЛистов

Cсылка на местоположение полного комплекта исходных модулей программы: [**https://github.com/lequangnhueNdx86/EXAM.git**](https://github.com/lequangnhueNdx86/EXAM.git)

Некоторые тексты кодов (в файле ExamService.java):

**public** ExamGenerateResponse generateExam(ExamRequest examRequest) {

Exam exam = mapper.map(examRequest, Exam.**class**);

Lesson lesson = lessonRepository.findByIdAndRemovedFalse(examRequest.getLesson()).get();

// generate exam

// totalQuestion

Integer totalQuestion = getTotalQuestion(lesson);

// totalTime

Integer timeExam = getTotalTimeExam(lesson);

// total score

Integer totalScore = getTotalScore(lesson);

// random question & set answer false

List<Answer> listAnswer = randomQuestionAndSetAnswerFalse(lesson, exam);

exam.setTotalQuestion(totalQuestion);

exam.setCorrectQuestion(0);

exam.setTimeExam(timeExam);

exam.setTimeComplete(0.0);

exam.setTotalScore(totalScore);

exam.setScore(0);

exam.setResult(Constants.***FAIL***);

exam.setLesson(lesson);

exam.setListAnswer(listAnswer);

exam = examRepository.saveAndFlush(exam);

**return** mapExamToExamGenerateResponse(exam);

}

**public** ListExamGetResponse getAllExam() {

List<Exam> listExam = examRepository.findByOrderByCreateTimeDesc();

List<ExamGetResponse> list = **new** ArrayList<>();

listExam.forEach(exam -> {

**if** (exam.getLesson().getRemoved() == **false**) {

list.add(mapExamToExamGetResponse(exam, mapLessonToLessonGetResponse(exam.getLesson())));

}

});

**return** **new** ListExamGetResponse(list, Integer.*valueOf*(list.size()), **new** LessonGetResponse());

}

**public** ListExamGetResponse getAllExamByLesson(Integer lessonId) {

// get list exam by lesson id from db

List<Exam> listExam = examRepository.findAllByLessonId(lessonId);

// get lesson from db

Lesson lesson = lessonRepository.findByIdAndRemovedFalse(lessonId).get();

// map lesson -> lesson get response

LessonGetResponse lessonGetResponse = mapLessonToLessonGetResponse(lesson);

List<ExamGetResponse> listExamGetResponses = **new** ArrayList<>();

// each exam, map exam -> exam get response

listExam.forEach(exam -> {

listExamGetResponses.add(mapExamToExamGetResponse(exam, lessonGetResponse));

});

**return** **new** ListExamGetResponse(listExamGetResponses, listExamGetResponses.size(), lessonGetResponse);

}

**public** ExamDetailGetResponse getExamById(Integer examId) {

Exam exam = examRepository.findById(examId).get();

// map exam -> exam detail by name & type

ExamDetailGetResponse examDetailGetResponse = mapper.map(exam, ExamDetailGetResponse.**class**);

examDetailGetResponse.setLesson(mapLessonToLessonGetResponse(exam.getLesson()));

List<Answer> listAnswer = exam.getListAnswer();

List<AnswerGetResponse> answerGetResponses = **new** ArrayList<>();

listAnswer.forEach(answer -> {

answerGetResponses.add(mapAnswerToAnswerGetResponse(answer));

});

examDetailGetResponse.setListAnswer(answerGetResponses);

**return** examDetailGetResponse;

}