**OB1 Проект самообучения**

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

**Задание**

Реализовать функционал самообучения для студентов. Для этого необходимо создать платформу, которая работает только с авторизованными пользователями. На платформе необходимо предусмотреть функционал разделов и материалов. Для каждого материала можно добавить тесты. Управление всеми сущностями необходимо реализовать через стандартный Django admin. Проверка ответа на тест осуществляется с помощью отдельного запроса на бэкенд. Реализовать либо Rest API, либо SSR с использованием Bootstrap. Для реализации проекта использовать фреймворк Django.

**Функции системы:**

* регистрация и авторизация пользователей
* добавление, редактирование и просмотр справочников и обучающих материалов (пользователей, предметов, их разделов, url-ссылок на дополнительные материалы, вопросов к разделам для тестирования)
* поиск обучающих материалов по параметрам
* предоставление (или запрет) обучающимся доступа к материалам раздела для изучения и тестирования
* просмотр обучающих материалов
* генерация тестов
* прохождение студентами тестирования
* оценка результатов тестирования

**Разрешения**

Распределение доступа между пользователями показано на диаграмме вариантов использования (рис. 1).

Все пользователи могут регистрироваться и авторизоваться в приложении.

Также есть 3 категории пользователей с различными правами.

Администратор имеет доступ ко всему функционалу системы. Он может редактировать все данные, просматривать списки зарегистрированных пользователей, получать информацию по всем предметам и разделам.

Автор предмета – создает, редактирует и удаляет данные по предметам и их разделам, заполняет вопросы для тестирования по разделам, прикрепляет ссылки на дополнительные материалы для раздела, записывает студентов на свои предметы, также может получать списки предметов под его авторством и просматривать результаты тестирования студентов по своим предметам.

Студент – может получить список всех предметов без детализации информации, просматривать детальную информацию по предметам и разделам, к которым ему предоставлен доступ. Студент может пройти промежуточное (по отдельному разделу) и итоговое (по всему курсу) тестирование и посмотреть результаты своего тестирования.



**Рис. 1 Диаграмма вариантов использования**

**Архитектура приложения**

Программа представляет собой API-приложение. API-запросы пользователей и ответы на них передаются в json-формате. Для реализации программы используются DRF и СУБД Postgresql. Также можно работать со всеми моделями через панель администратора Django (*только там сейчас не очень удобный интерфейс для некоторых объектов и не учитываются права пользователей*). Документация по программе формируется автоматически с помощью Swagger.

Структура моделей базы данных приведена на рисунках 2-3.

Функционал системы распределен между тремя приложениями:

* Users – регистрация, авторизация пользователей, вывод списка пользователей
* Study – выполнение операций CRUD для справочников предметов, разделов и дополнительных материалов
* Tests\_study – генерация тестов, прохождение тестирования, оценка и вывод результатов тестирования.

Пользователи регистрируются по электронной почте. Также можно указать фамилию, имя пользователя, прикрепить аватар. *Верификации почты на данный момент не предусмотрена.* Для авторизации пользователей используется JWT (JSON Web Token).

Предмет состоит из разделов, к разделу могут быть прикреплены url-ссылки на дополнительные материалы (предусмотреть возможность прикрепления материалов и для всего курса? Разные виды материалов?). Все данные может создавать, редактировать, удалять автор предмета. Также возможен поиск по предметам и их разделам с применением параметров api-запроса для администратора, автора курса и подписанных студентов. Для поиска используются библиотеки SearchFilter, OrderingFilter, DjangoFilterBackend.

Для того, чтобы студент мог просматривать материалы предмета, автор добавляет ссылку на него в таблицу AccessSubjectGroup. Для запрета доступа автор удаляет из этой таблицы соответствующую запись.

Для каждого раздела можно добавить вопросы с вариантами ответов для тестирования. Количество вопросов, включаемых в тест из каждого раздела, настраиваются автором предмета (если вопросов будет недостаточно, то будут добавлены все имеющиеся вопросы раздела). Вопросы различаются по сложности (от 1 до 5, по умолчанию 1). В программе используются только вопросы с альтернативными ответами (можно выбрать только один вариант ответа из предложенных).

Тесты генерируются каждый раз заново из заполненных автором предмета вопросов. При генерации теста вопросы и ответы на них перемешиваются случайным образом. Тесты могут быть двух типов – промежуточные (по отдельному разделу) и итоговые (по всему предмету).

**Структура базы данных**



**Рис.2 Логический уровень**



**Рис.3 Физический уровень**

**Варианты доработки программы:**

Для пользователей можно:

- добавить верификацию почты

- добавить возможность выводить все предметы под авторством пользователя по его id

Для тестов можно:

- добавить количество попыток тестирования,

- указать будет ли случайно изменяться порядок ответов

- учесть продолжительность тестирования

- предусмотреть нормализацию значений для расчета баллов (приведение к фиксированному диапазону значений)

- предусмотреть разные типы вопросов (с несколькими вариантами ответов, с ответами в произвольной форме и т.д.)

Для предметов, разделов и материалов можно:

- предоставить возможность передачи прав автора на предмет

- предусмотреть возможность добавления картинок

- разные виды материалов (книги, сайты, видео и т.д.)

- ограничить период доступа к материалам

- предоставить возможность пользователям оценивать предметы, разделы и дополнительные материалы

- дать возможность оставлять комментарии и вопросы к разделам

Для обучения можно:

- отслеживать прогресс обучения

- запрещать доступ к следующим разделам и тестированию, пока текущий материал не будет изучен