Резюме

1. Представяне

Начален слайд с темата

Уважаема комисия, Уважаема г-жо Директор, уважаеми гости (ако има такива/ или поименно – ако е един конкретен гост)

Аз съм …..

Темата на моят дипломен проект е …

Всъщност, това е функционално и структурно обособена част от един по-голям проект, наречен „Електронна библиотека“. Той има за цел да създаде една обща и достъпна платформа, която да събере и предложи учебни материали в електронен вид по всички специални предмети от 8-ми до 12-ти клас по всички специалности от нашата гимназия и да даде възможност за самостоятелна проверка на усвоените знания с помощта на тестове.

Базирайки се на използваните в сайта на нашето училище - ПГЕЕ Банско, технологии, екипът ни разработи уеб приложение на Django, използващо Java Script и HTML.

Сменям слайда

1. Тестове

Тестовете са популярна форма за проверка на знания и има много приложения, които предлагат такава възможност. Google и Microsoft Teams, например, са едни от тях. Но към тях има и редица забележки, например, трудно се вграждат в други системи, което е необходимо в нашия случай.

Съставянето на функционален модул, вграден в общ проект, е предизвикателството, което аз приех. Това е задачата, чието решение се заех да намеря.

Сменям слайда

1. Разделяне на функциите

Един сайт или уеб приложение обикновено се разработва от отбор – екип, който заедно чертае целите, но разделя по някакъв критерий работата между своите членове. Най-често, един работи по функционалността - backend, а друг по визията - frontend.

В нашият случай разделението е по функционалност, в известен смисъл, и двете влязохме в ролята на full stack програмисти.

Моята задача беше да създам необходимите модели, API функции и клиентски интерфейс, които да реализират приложение за създаване, редактиране и провеждане на тестове. Надявам се, че създадох базата, която ще позволи на всеки учител, който би искал да я използва, бързо и лесно да направи тест за учениците си.

Сменям слайда

1. Реализацията

Визията на тестовата система ще се стори позната на тези, които са ползвали платформата DIKI. Това не е случаен, а е търсен ефект. Използван е точно същия визуален шаблон. Това до голяма степен важи и за функционалността на потребителския интерфейс. Например, сложността на въпросите се задава по Таксономията на Блум, с което се определят присвояваните точки за верен отговор.

Естествено, алгоритмите за генериране и оценяване на теста са съвсем различни.

Сменям слайда

1. БД

Използвам MySQL базата данни на приложението, но въпросите са съхраняват в отделени таблици и връзката с електронната библиотека е до толкова, доколкото за всеки въпрос има маркери на предмет и тема, върху които е въпроса. Това позволява да се генерират тестове по цял предмет или по конкретна тема.

Сменям слайда

1. DRF

Тук се включва фреймурка DRF - един мощен инструмент, който всъщност може да направи пълноценен приложен интерфейс с минимални усилия.

С помощта на подходящи сериализатори и съответните изгледи изградих API (апликейсън програминг интефейс), който отговаря на клиентски заявки като връща данни в JSON формат.

С това backend-а на приложението е завършен.

Сменям слайда

1. Клиентския интерфейс

Основната функционалност, като почнем от създаването на нови въпроси и редактирането им, и стигнем до самото генериране, провеждане и оценяване на тест е изведена като софтуер от страна на клиента. Използваме силата на VUE, интегрирана с DRF и Django.

Сменям слайда

1. Интеграцията

Django предоставя шаблоните и сервира на клиента JavaScript файлове, но основната работа се върши при клиента. Той изпраща запитване чрез JavaScript до сървъра и сървърът, с помощта на библиотеката DRF посредством сериализатор, връща данните. Случайния избор на въпроси и редактирането става при клиента.

Ако всяка отделна промяна се изпраща на сървъра, то той ще бъде принуден през цялото време да задържа ресурси, да поддържа сесии, да помни на кой въпрос очаква редакция и т.н. Това излишно би натоварило както комуникацията между клиента и сървъра, така и самия сървър.

Моето решение е по-елегантно и по-удобно. Функционалността е изнесена при клиента, а сървърът предоставя необходимите ресурси и данни. Такова решение също така дава възможност цялата тестова система да се прикачи почти без проблем към което и да е друго приложение.

Сменям слайда

1. VUE обекта

Реализацията на динамичен и интерактивен интерфейс е на базата на framework-а VUE. На практика това е един JavaScript обект със специфичен синтаксис.

Като всеки обект, той има данни и методи.

Чрез специфичния за VUE синтаксис обвързваме данните и методите н този обект с определени контроли от клиентския интерфейс. Промяна в данните води до промяна в контролите, а действие с контролите извиква метод на обекта.

Сменям слайда

1. Заключение

Накратко моята част от проекта ни електронна библиотека, е да демонстрирам как се вгражда функционален модул в приложение, в моят случай това е тестова система. От една страна тя е функционално интегрирана и ние сме работили в екип, за да се съгласуват нещата едно с друго, а от друга страна е алгоритмично, функционално и даже визуално независима система тестове.

Всеки, който има съответните права за достъп може да променя въпросите, а провеждането на тест е достъпно свободно. Правата на достъп се задават през административния панел на приложението.

Сменям слайда

Благодаря за вниманието!

Ако имате въпроси съм готов да отговоря.

Въпроси: