* Название: сайт для создания и прохождения тестов
* Автор: Гартвих Егор
* Идея: создать сайт, на котором зарегистрированным пользователям был бы доступен функционал создания тестов, а проходить их можно было бы в свободном доступе, без регистрации.
* Реализация:  
  в проекте есть 3 модуля: admin, data и forms.

Модуль admin содержит файлы users.py и tests.py и задаёт представление этих моделей в админке(что можно редактировать, что нельзя, нужна ли авторизация для доступа к ним). Эти файлы содержат соответственно классы UsersAdminView и TestsAdminView, в которых необходимые свойства и прописаны.

Модуль data также содержит файлы users.py и tests.py с классами Tests и User соответственно. В них описываются структуры моделей тестов и пользователя в базе данных. Также, в модели пользователя есть функции set\_password, check\_password и метод \_\_repr\_\_. Они позволяются задать пароль(сгенерировать с помощью хэш-функции), проверить введенный пароль на совпадение с оригинальным, а метод repr позволяет получить красивое представление данных о пользователе.

Сама база данных содержит две таблицы – users и tests, связанные отношением one-to-many(у одного пользователя может быть много тестов, а автор у теста один). Для каждого теста в поле author\_id храним id его создателя, в поле questions – вопросы в формате “вопрос;%ответ;$очки” и в поле result результат теста в формате “от скольки очков-до скольки очков;&результат”

Модуль forms тоже содержит файлы users.py и tests.py, в которых есть классы форм.

По файлу forms/tests.py: TestId и TestNameSubmit – формы для поиска теста по его номеру и по названию соответстенно. TestForm – форма для создания теста, Answers – для добавления вопрос и ответов при создании. TestAnswers – форма для выбора ответов при прохождении теста. AccountSubmit – попытался сделать поиск аккаунтов, но не успел.

В файле forms/users.py классы RegisterForm и LoginForm, отвечающие за регистрацию и вход.

* Функции файла main.py – основного файла с кодом.

Рассмотрим нестандартные функции:

create\_test , отвечающая за эндпоинт /tests/create – функция, обрабатывающая форму для создания теста, проверяющая валидность данных и сохраняющая вопросы и результаты в нужном для базе данных виде(описан выше). Также обрабатывает сигналы «подформ», то есть добавление/удаления вопроса или результата.

Функции 404 и 401 отвечают за обработку ошибок с соответствующими кодами(«Страница не найдена» и «необходима авторизация»).

Функция “run\_news” отвечает за эндпоинт /tests/{test\_id} и позволяет пройти тест. После каждого ответа на вопрос она запоминает количество очков и номер текущего вопроса(очки получает из базы данных). В конце прохождения перекидывает на страницу tests/{test\_id}/finish/{score}

Функция finish\_test отвечает за вышеупомянутый эндпоинт и показывает пользователю результат, соответствующий количеству набранных за проходимый тест очков.

Функция view\_test ответ за эндпоинт /tests/view/{test\_id}. Позволяет посмотреть основную информацию о тесте (название, описание, автор, дата создания) и запустить его прохождение при нажатии на нужную кнопку на странице. Если запрашиваемого теста не существует, функция корректно обработает этот случай.

Функции search\_by\_id и search\_by\_name отвечают за поиск теста по номеру или названию. При нахождении нужного они перекинут пользователя на страницу такого теста, в обратном случае – корректно покажут, что такого теста не существует.

Функция view\_user позволяет посмотреть информацию о пользователе (его имя и детальную информацию, если указана), функция profile доступна только самому пользователю и позволяет также увидеть приватную информацию(например, email).

Используемые технологии:

Flask 3.1.0

Flask-Login 0.6.3

Flask-Admin 1.6.1

Flask-WTF 1.2.2

Jinja2 3.1.6

SQLAlchemy 2.0.40

WTForms 3.2.1

Скриншоты:

