**Тестовое задание на Python Backend Developer**

Необходимо реализовать небольшое Django-приложение для библиотеки, которое позволяет читателям выбрать интересующую книгу из каталога библиотеки, взять ее почитать, а потом вернуть.

***Должны быть реализованы, как минимум, 2 сущности:***

1. **Пользователи**. 2 роли пользователей: Библиотекарь и Читатель
   1. Минимальные обязательные атрибуты роли Библиотекарь: username, password, табельный номер
   2. Минимальные обязательные атрибуты роли Читатель: username, password, имя, фамилия, адрес проживания (просто текстовое поле)
2. **Книги.** Минимальный набор полей: название, автор, жанр.

Структура таблиц базы данных, конечный список моделей и связи – на усмотрение исполнителя.

***Функционал приложения:***

*Вэб-интерфейс на Django-templates:*

Веб-интерфейс должен быть простым, желательно использование bootstrap. Должна быть панель навигации для перемещения между страницами, кнопка для работы с сессиями аутентификации (войти/выйти).

1. Читатель:
   1. Страница регистрации / входа – по username + password.
   2. Главная страница – Каталог книг. Представляет из себя список/таблицу всех книг в каталоге библиотеки. Книги должны быть отсортированы по названию, на странице видны название, автор и жанр. В строке с книгой должна быть кнопка, которая позволит взять книгу почитать или вернуть ее, если Читатель уже ранее взял ее.
   3. Вторая страница – Мои книги. Список книг (название, дата получения книги и количество дней, сколько книга на руках), которые сейчас находятся на руках у Читателя (допускаем, что при нажатии кнопки «Взять» книга сразу попадает к Читателю). Сортировка по названию книги.
   4. \* Доп.задание – на главной странице рядом с названием добавить кнопку/значок «Копировать», чтобы можно было копировать в буфер обмена название книги.
2. Библиотекарь:
   1. Страница регистрации / входа – по username + password.
   2. Главная страница – Список должников: username, имя, фамилия, адрес, название, дата получения книги и количество дней, сколько книга на руках у должника;

*API с использованием Django REST Framework:*

1. Читатель:
   1. Любая аутентификация по токенам, включая JWT (можно использовать любые библиотеки)
   2. Получение списка книг (название, автор, жанр)
   3. Взять книгу
   4. Вернуть книгу
   5. Получить список книг на руках (название, дата, когда взял, количество дней, сколько книга на руках у должника)

\* Доп.задание – подключить api-документацию с использованием drf-spectacular или drf-yasg

*Админ-панель:*

В админ-панели должны быть зарегистрированы основные сущности, чтобы можно:

1. Просматривать, создавать, изменять, удалять пользователей
2. Просматривать, создавать, изменять, удалять книги
3. Просматривать, создавать, изменять, удалять списки всех, кто когда-либо брал книги в библиотеке, с возможностью фильтровать по тем, кто еще не вернул книгу
4. \* Доп.задание - для фиксации истории изменений в моделях книг необходимо применить библиотеку Django Simple History.

***Полезные ссылки:***

* Django - <https://www.djangoproject.com/>
* Django REST Framework - <https://www.django-rest-framework.org/>
* Django simple history - <https://django-simple-history.readthedocs.io/>
* drf-spectacular - <https://drf-spectacular.readthedocs.io/>
* drf-yasg - <https://drf-yasg.readthedocs.io/>

***Сдача задания:***

* Проект должен быть размещен на Github
* Подготовить readme с инструкциями по запуску проекта.