МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

«САМАРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ имени академика С.П.КОРОЛЕВА»

ИНСТИТУТ ИНФОРМАТИКИ И КИБЕРНЕТИКИ

КАФЕДРА ТЕХНИЧЕСКОЙ КИБЕРНЕТИКИ

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №1

Тема: «Разработка базы данных»

Дисциплина: «Технологии сетевого программирования»

Выполнил: Якухин И.В., гр.6303

Проверил: Кашапов А.И.

Самара 2025

СОДЕРЖАНИЕ

[Введение 3](#_Toc199543999)

[1 Описание базы данных 4](#_Toc199544000)

[1.1 Колода (Deck) 4](#_Toc199544001)

[1.2 Карта (Card) 4](#_Toc199544002)

[1.3 Пользователь (User) 5](#_Toc199544003)

[1.4 Прогресс пользователя (User\_progress) 5](#_Toc199544004)

[2 Скрипты для заполнения таблиц 7](#_Toc199544005)

[Приложение А 8](#_Toc199544006)

[Ссылка на GitHub с проектом 8](#_Toc199544007)

[Приложение Б 9](#_Toc199544008)

[Скрипт для заполнения таблицы Cards “fill\_cards.py” 9](#_Toc199544009)

[Скрипт для заполнения таблицы Users “fill\_users.py” 9](#_Toc199544010)

[Скрипт для заполнения таблицы User\_progress “fill\_progress.py” 9](#_Toc199544011)

Введение

Cardiogram – приложение с увлекательной игрой для запоминания английских слов с помощью карточек. Стек технологий приложения: Python, Django, React, PostgreSQL. Все компоненты конечной системы контенеризованы с помощью Docker для обеспечения изоляции окружения, удобства развертывания и масштабирования.

Учебный проект разработан с целью демонстрации навыков работы с базой данных и инструментами серверной и клиентской разработки, проектирования структуры приложений и API, контейнеризации.

Ссылка на GitHub страницу со всеми файлами проекта можно найти в приложении А.

1. Описание базы данных

Основу базы данных составляют 4 таблицы: Users, Cards, Decks, User\_progress. Каждая таблица этой базы данных является представлением определенных сущностей, связанных между собой отношениями. Ниже приведено более подробное описание каждой сущности и связей между ними.

* 1. Колода (Deck)

Описывает объединение карточек, связанных с одной общей тематикой.

Поля:

* Id – уникальный идентификатор записи;
* Name – название колоды;
* Description – описание тематики колоды;
* Cover – ссылка на медиафайл, содержащий изображение обложки колоды.

Сущность связана с сущностью Карта (Card) отношением один ко многим. Одной колоде может принадлежать несколько карточек.

* 1. Карта (Card)

Описывает карточку с иностранным словом и его переводом.

Поля:

* Id – уникальный идентификатор записи;
* Deck – идентификатор колоды, к которой относится карточка;
* Front\_text – иностранное слово, которое будет отображаться на передней стороне карточки;
* Back\_text – перевод слова, который будет отображаться на обратной стороне карточки;
* Example\_usage – пример употребления иностранного слова в контексте с переводом.

Сущность связана с сущностью Колода (Deck) отношением многие к одному через поле deck. Несколько карточек могут принадлежать одной колоде.

Также связана с сущностью Прогресс пользователя (User\_progress) отношением один ко многим.

* 1. Пользователь (User)

Содержит данные пользователя, взаимодействующего с приложением.

Поля:

* Id – уникальный идентификатор записи;
* Username – уникальное пользовательское имя, которое используется для аутентификации в приложении;
* First\_name – имя пользователя;
* Last\_name – фамилия пользователя;
* Email – электронная почта пользователя;
* Password – пароль от аккаунта пользователя.

Пользователь связан с сущностью Карта (Card) отношением многие ко многим через таблицу User\_progress. Один пользователь может изучать несколько карточек, и одна карточка может изучаться многими пользователями.

Пользователь также связан с сущностью Прогресс пользователя (User\_progress) отношением один ко многим. У одного пользователя может быть несколько записей о прогрессе.

* 1. Прогресс пользователя (User\_progress)

Содержит записи о прогрессе пользователей по определенным карточкам.

Поля:

* Id – уникальный идентификатор записи;
* User – идентификатор пользователя, которому принадлежит запись о прогрессе;
* Card – идентификатор карты, прогресс изучения которой засчитывается пользователю;
* Attempts – общее число попыток пользователя отгадать карточку;
* Successful\_attempts – число успешных попыток пользователя отгадать карточку.

Как было указано выше таблица User\_progress помогает осуществлять связь многие ко многим между сущностями Карта (Card) и Пользователь (User).

1. Скрипты для заполнения таблиц

Для заполнения таблиц данными в основном модуле приложения (cardiogram) была создана директория “management/commands”, в которую были помещены скрипты, создающие записи таблиц из текстовых файлов с данными. Код трёх скриптов приведен в приложении Б. Для их запуска была использована команда “python manage.py <название скрипта>”. Таким образом были заполнены таблицы Users, Cards, User\_progress. На рисунке 1 приведен скриншот admin-панели Django с отрытой таблицей, наполненной данными.

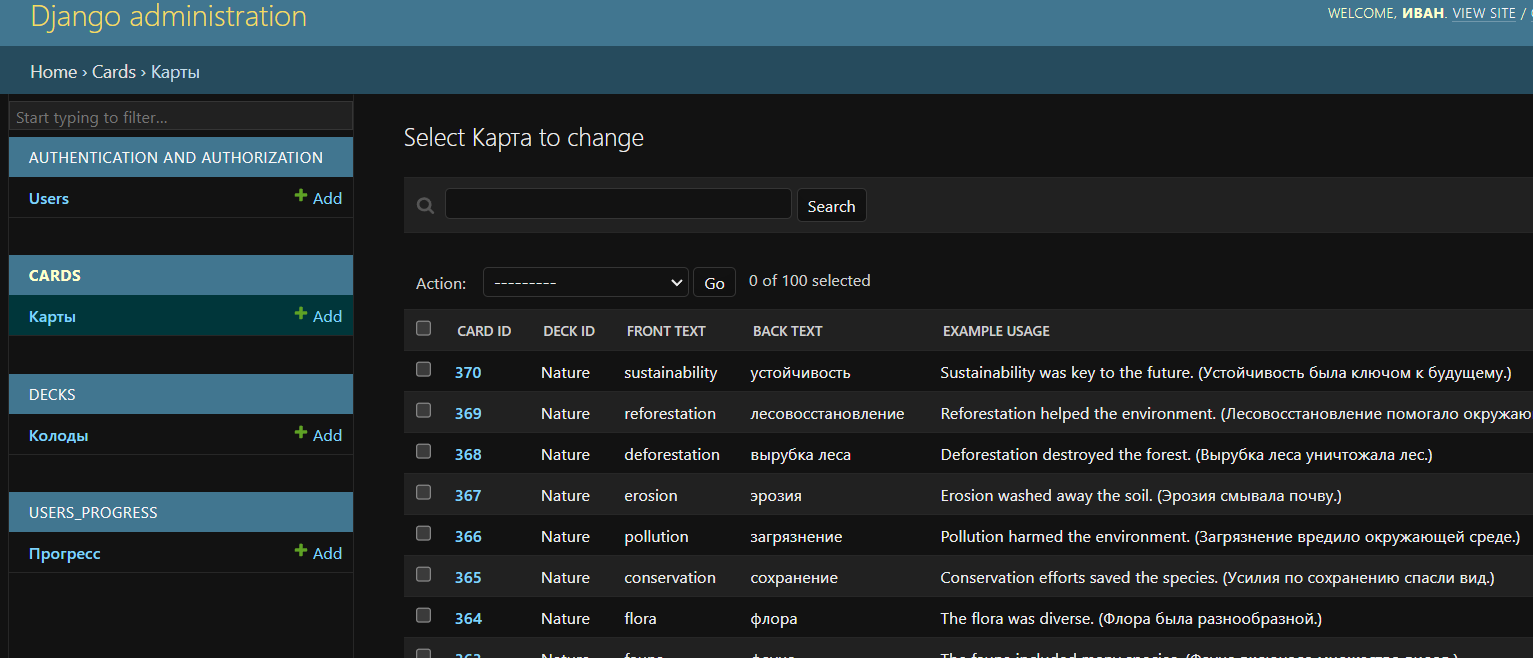


Рисунок 1 – Скриншот admin-панели Django с таблицей Cards

Приложение А

Ссылка на GitHub с проектом

https://github.com/BruhMano/CardioGram.git

Приложение Б

Скрипт для заполнения таблицы Cards “fill\_cards.py”

from cards.models import Card

from decks.models import Deck

from django.core.management.base import BaseCommand

class Command(BaseCommand):

help = 'Заполняет базу данных тестовыми карточками'

def handle(self, \*args, \*\*options):

eng\_file = open('data/nature\_eng.txt', 'r')

ru\_file = open('data/nature\_ru.txt', 'r', encoding='utf-8')

for word in eng\_file:

word = word.strip()

data = ru\_file.readline().rstrip().split(' - ')

card = Card(deck = Deck.objects.get(name = 'Nature'),front\_text = word,

back\_text = data[0], example\_usage = data[1])

card.save()

Скрипт для заполнения таблицы Users “fill\_users.py”

from django.contrib.auth import get\_user\_model

from django.core.management.base import BaseCommand

class Command(BaseCommand):

help = 'Заполняет базу данных тестовыми пользователями'

def handle(self, \*args, \*\*options):

data\_file = open('data/users.txt', 'r', encoding = 'utf-8')

user\_model = get\_user\_model()

for line in data\_file:

data = line.strip().split(' ')

user = user\_model(username = data[0], first\_name = data[1],

last\_name = data[2], email = data[3])

user.set\_password(data[4])

user.save()

Скрипт для заполнения таблицы User\_progress “fill\_progress.py”

from cards.models import Card

from users\_progress.models import Progress

from django.contrib.auth import get\_user\_model

from django.core.management.base import BaseCommand

class Command(BaseCommand):

help = 'Заполняет базу данных тестовыми данными о прогрессе пользователя'

def handle(self, \*args, \*\*options):

data\_file = open('data/progress.txt', 'r')

for line in data\_file:

data = line.strip().split(' ')

card = Card.objects.filter(front\_text=data[1]).first()

print(card)

progress = Progress(user = get\_user\_model().objects.get(username = data[0]),

card = card,

attempts = data[3],successful\_attempts = data[2])

progress.save()