

ЕЛИЗАВЕТА ТРУШИНА

Разработчик на Python

E-MAIL: LIZATRUSHINA96@GMAIL.COM
GITHUB: [HTTPS://GITHUB.COM/PROLISBET](https://github.com/prolisbet)

ОБО МНЕ

- Окончила бакалавриат МАрхИ (Московский Архитектурный Институт) с красным дипломом по направлению «Архитектура жилых и общественных зданий»
- Более 7 лет опыта работы в сфере архитектуры, должности от помощника архитектора до ведущего архитектора
- Изучала курсы по направлению Data Science от Skillbox
- Прошла курс по программированию на Python с нуля от Зерокодер
- Прошла интенсив по анализу данных от Зерокодер
- Участвовала в командных разработках от Зерокодер
- Имею большой интерес к машинному обучению и web-разработке
- Люблю решать сложные задачи и осваивать новые технологии

НАВЫКИ

Языки программирования: Python - основной, SQL

Инструменты и технологии: Git, Repl.it, PyCharm, Jupiter Notebook, VirtualBox, Linux (Kali)

Библиотеки и фреймворки: Standard Library, NumPy, Pandas, Matplotlib, Seaborn, SciPy, Requests, Django, Flask, Bootstrap, BeautifulSoup, Sqlite3, Tkinter, Pygame, Telebot, Aiogram, Asyncio, Pytest, Unittest, Ffmpeg, EasyOCR, Googletrans, Translate, OpenCV

Модели: Whisper (транскрипция речи), CLIP (поиск изображений по текстовым запросам), GPT (написание и оптимизация промтов)

Базы данных: SQLite, MySQL

Прочие навыки: Web-разработка, разработка Telegram-ботов, работа с API, работа с HTML и CSS

ПРОЕКТЫ

Pomodoro Bot

Задачи проекта:

Разработать Telegram-бота, который будет исполнять роль таймера и помогать пользователю организовать рабочий процесс с помощью техники «помидора».

Описание проекта:

Pomodoro Bot — это ваш персональный тайм-менеджер, опирающийся на известную технику чередования периодов работы и отдыха. В наше время может быть очень непросто найти тонкую грань между прокрастинацией и выгоранием. Мой бот поможет вам эффективно организовать рабочее время в удобном для вас ритме, напоминая, когда стоит передохнуть, а когда снова взяться за работу.

ПРОЕКТЫ

Pomodoro Bot

Особенности проекта:

- Пользователь может выбрать один из трех вариантов временных интервалов по своему усмотрению и предпочтениям:
 - короткий — 12 минут работы и 3 минуты отдыха,
 - стандартный — 25 минут работы и 5 минут отдыха,
 - длинный — 45 минут работы и 15 минут отдыха.
- Информационная справка об используемой технике работы

ПРОЕКТЫ

Pomodoro Bot

```
1 import telebot
2 from telebot import types
3 import datetime
4 from datetime import timedelta
5 import time
6 import threading
7 import pytz
8
9 bot = telebot.TeleBot('6721466990:AAENDVKINfkfMZA-C005Gwkr5JuCTRkds')
10 stop_loop = {}
11 work = 0
12 rest = 0
13
14 @bot.message_handler(commands=['start'])
15 def start_message(message):
16     bot.send_message(message.chat.id, text='Привет! Я чат-бот, который поможет тебе организовать рабочий процесс!')
17     bot.send_message(message.chat.id, 'Скорее вводи команду /work и мы начнем работу...')
18
19 @bot.message_handler(commands=['work'])
20 def work_message(message):
21     chat_id = message.chat.id
22     stop_loop[chat_id] = False
23     markup = types.InlineKeyboardMarkup()
24     btn1 = types.InlineKeyboardButton(text="1", callback_data='1')
25     btn2 = types.InlineKeyboardButton(text="2", callback_data='2')
26     btn3 = types.InlineKeyboardButton(text="3", callback_data='3')
27     # Добавляем кнопки в клавиатуру
28     markup.add(btn1,btn2,btn3)
29     # Отправляем сообщение пользователю с этой клавиатурой
30     bot.send_message(message.chat.id, '*Выбери один из трех помидоров для работы:* \n'
31                     '1. 12 минут работы - 3 минуты отдыха; \n'
32                     '2. 25 минут работы - 5 минут отдыха; \n'
33                     '3. 45 минут работы - 15 минут отдыха.', parse_mode='Markdown', reply_markup=markup)
```

```
35 @bot.callback_query_handler(func=lambda call: True)
36 def query_handler(call):
37     global work, rest
38     bot.answer_callback_query(callback_query_id=call.id)
39     answer = ''
40     if call.data == '1':
41         answer = 'Вы выбрали 12 минут работы - 3 минуты отдыха'
42         work = 12
43         rest = 3
44     elif call.data == '2':
45         answer = 'Вы выбрали 25 минут работы - 5 минут отдыха'
46         work = 25
47         rest = 5
48     elif call.data == '3':
49         answer = 'Вы выбрали 45 минут работы - 15 минут отдыха'
50         work = 45
51         rest = 15
52
53     bot.send_message(call.message.chat.id, answer)
54
55 work_thread = threading.Thread(target=pomodoro_clock, args=(call.message.chat.id, work, rest))
56 work_thread.start()
57
58 def pomodoro_clock(chat_id, work, rest):
59     moscow_tz = pytz.timezone('Europe/Moscow')
60     now = datetime.datetime.now().astimezone(moscow_tz)
61     text_now = now.strftime('%H:%M')
62     bot.send_message(chat_id, f'Начинаем работать в {text_now}')
63     n = 0
64     while True:
65         work_delta = timedelta(minutes = (work+rest) * n + work)
66         rest_time = (now + work_delta).strftime('%H:%M')
67         rest_delta = timedelta(minutes = (work+rest) * n + (work+rest))
```

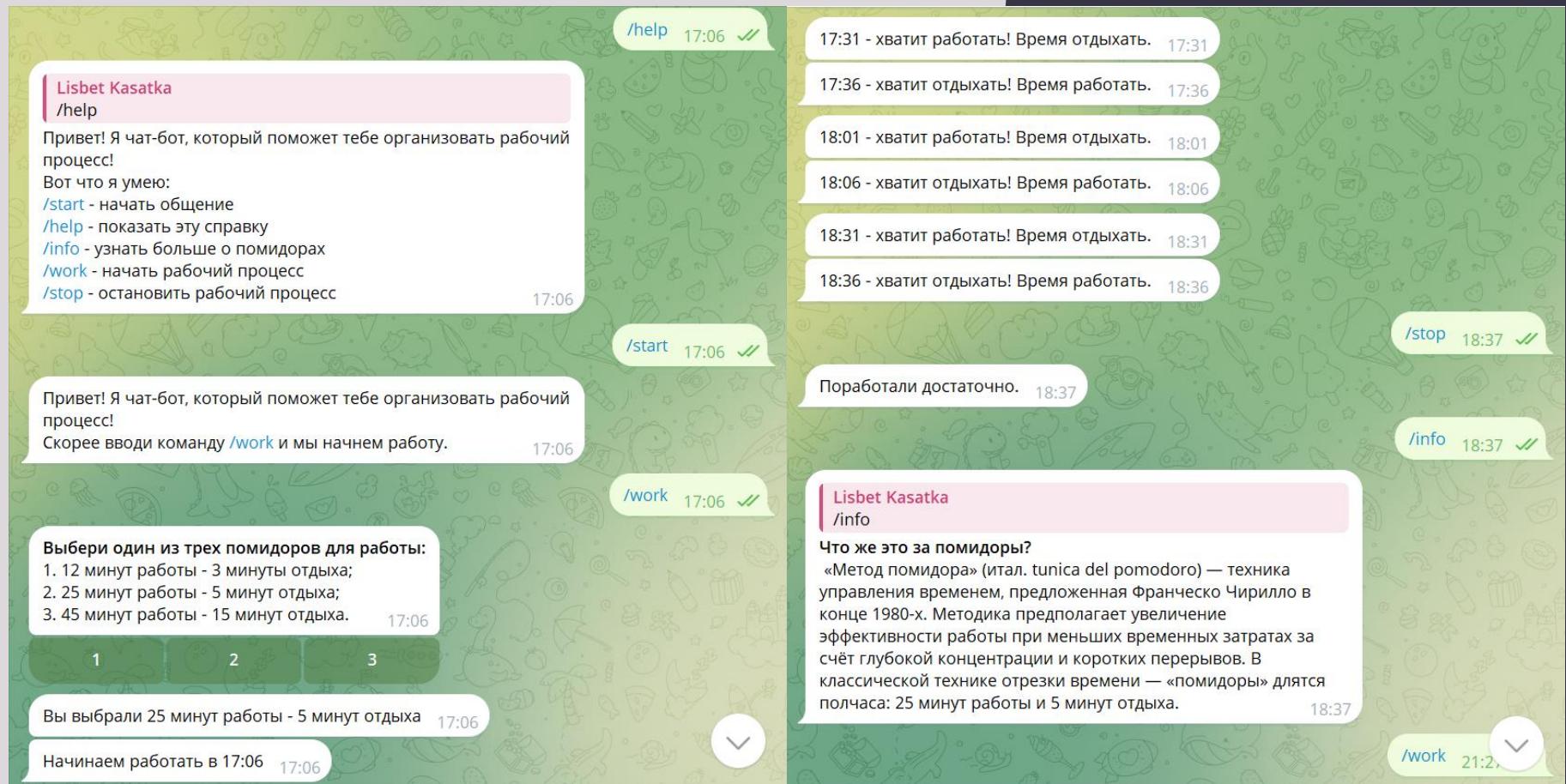
ПРОЕКТЫ

Pomodoro Bot

```
68     work_time = (now + rest_delta).strftime('%H:%M')
69     new_now = datetime.datetime.now().astimezone(moscow_tz
70                           ).strftime('%H:%M')
71     if new_now == rest_time:
72         bot.send_message(chat_id, f'{rest_time} - хватит работать! Время отдохнуть.')
73         custom_sleep(chat_id, 60)
74     elif new_now == work_time:
75         bot.send_message(chat_id, f'{work_time} - хватит отдохнуть! Время работать.')
76         n += 1
77         custom_sleep(chat_id, 60)
78     if stop_loop.get(chat_id):
79         break
80
81 def custom_sleep(chat_id, sec):
82     start = time.time()
83     while time.time() - start < sec:
84         if stop_loop.get(chat_id):
85             break
86         pass # Активное ожидание
87
88 @bot.message_handler(commands=['info'])
89 def info_message(message):
90     info_text = ('«Метод помидора» (итал. tunica del pomodoro) – техника управления временем, '
91                 'предложенная Франческо Чирилло в конце 1980-х. Методика предполагает увеличение '
92                 'эффективности работы при меньших временных затратах за счёт глубокой концентрации '
93                 'и коротких перерывов. В классической технике отрезки времени – «помидоры» делятся на часы: '
94                 '25 минут работы и 5 минут отдыха.')
95     bot.reply_to(message, f'*Что же это за помидоры?* \n {info_text}', parse_mode='Markdown')
96
97 @bot.message_handler(commands=['help'])
98 def help_message(message):
99     help_text = ('Привет! Я чат-бот, который поможет тебе организовать рабочий процесс! \n'
100                'Вот что я умею: \n')
101
102
103
104
105
106
107
108 @bot.message_handler(commands=['stop'])
109 def stop_loop_handler(message):
110     chat_id = message.chat.id
111     # Остановить цикл для данного пользователя
112     stop_loop[chat_id] = True
113     bot.send_message(chat_id, "Поработали достаточно.")
114
115 bot.polling(none_stop=True)
```

ПРОЕКТЫ

Pomodoro Bot



ПРОЕКТЫ

Тетрис

Задачи проекта:

С помощью библиотеки `pygame` разработать приложение, имитирующее знаменитую аркадную игру.

Описание проекта:

Тетрис — это аркадная игра головоломка, в которой блоки падают и должны быть помещены в пустые места поля. Нужно установить блоки так, чтобы сформировать целую линию блоков и уничтожить их. Игра заканчивается, когда новая фигурка не может поместиться на поле.

ПРОЕКТЫ

Тетрис

Особенности проекта:

- Блокам случайно присваиваются разные цвета
- В полёте игрок может поворачивать фигурку на 90° и двигать её по горизонтали
- Можно «сбрасывать» фигурку, то есть ускорять её падение, когда уже решено, куда фигурка должна упасть

ПРОЕКТЫ

Тетрис

The screenshot shows a code editor with three tabs open:

- tetris.py**: Contains the main game loop and initialization code. It imports pygame, sys, random, and time. It sets up the screen, initializes pygame, defines colors (BLACK, WHITE, BLUE, RED, GREEN, YELLOW, VIOLET, ROSE, ORANGE), and sets up tetromino shapes.
- tris.py**: Contains the logic for the Tetris pieces. It defines a `Piece` class that takes `x`, `y`, and `shape` as parameters. It has methods for `__init__` and `rotate`. The `rotate` method rotates the piece by 90 degrees clockwise. It also contains comments about rotation logic and usage counts.
- etris.py**: Contains the logic for the board. It defines a `Board` class that takes width and height. It has methods for `__init__`, `add_piece`, `check_lines`, and `draw_piece`. The `add_piece` method adds a piece to the grid. The `check_lines` method removes full rows from the grid. The `draw_piece` method draws the piece on the screen.

ПРОЕКТЫ

Тетрис

```
py x
def draw_piece(screen, piece):
    for i, row in enumerate(piece.shape):
        for j, cell in enumerate(row):
            if cell:
                pygame.draw.rect(screen, COLORS[piece.index],
                                 rect: ((piece.x + j) * 30, (piece.y + i) * 30, 30, 30))
                pygame.draw.rect(screen, BLACK, rect: ((piece.x + j) * 30, (piece.y + i) * 30, 30, 30))

# Кнопки
game_over_button = pygame.Rect(125, 200, 180, 50)
new_game_button = pygame.Rect(125, 375, 180, 50)

font = pygame.font.Font(name: None, size: 36)

2 usages  prolibset
def draw_button(button, text, color=WHITE):
    pygame.draw.rect(screen, color, button)
    pygame.draw.rect(screen, BLACK, button, width: 1)
    text_render = font.render(text, antialias: True, BLACK)
    text_rect = text_render.get_rect(center=button.center)
    screen.blit(text_render, text_rect)

py x
def game_over():
    draw_button(game_over_button, text: "Game Over", WHITE)
    draw_button(new_game_button, text: "New Game", WHITE)
    pygame.display.flip()
    waiting_for_input = True
    while waiting_for_input:
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                pygame.quit()
                sys.exit()
            if event.type == pygame.MOUSEBUTTONDOWN and event.button == 1:
                if new_game_button.collidepoint(event.pos):
                    waiting_for_input = False
    return True

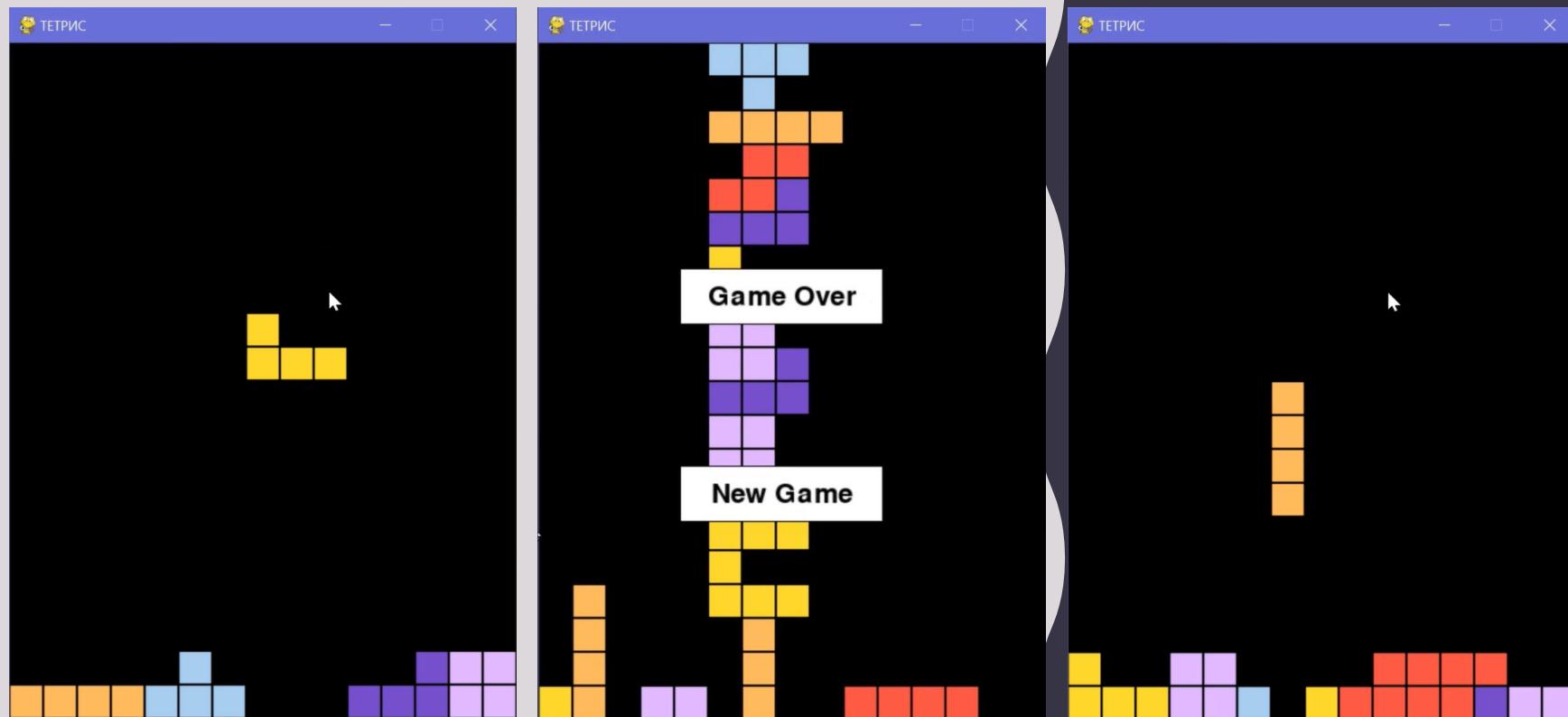
py x
while run:
    fall_time += clock.get_rawtime()
    clock.tick(fps)

    # Обработка событий
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            run = False

    # Управление фигуры клавишами:
    keys = pygame.key.get_pressed()
    if keys[pygame.K_LEFT]:
        current_piece.x -= 1
        if current_piece.x < 0 or check_collision(current_piece, board.grid):
            current_piece.x += 1
        time.sleep(0.1)
    if keys[pygame.K_RIGHT]:
        current_piece.x += 1
        if check_collision(current_piece, board.grid):
            current_piece.x -= 1
        time.sleep(0.1)
    if keys[pygame.K_UP]:
        current_piece.rotate()
        if check_collision(current_piece, board.grid):
            for _ in range(3): # Вращаем обратно
```

ПРОЕКТЫ

Тетрис



ПРОЕКТЫ

Поттериана

Задачи проекта:

Разработать сайт для фанатов саги о Гарри Поттере и для тех, кто только знакомится с «Поттерианой».

Описание проекта:

Поттериана — это сайт, где можно узнать, что же такое этот «Гарри Поттер», вспомнить содержание каждой из семи книг, а также познакомиться с главными персонажами.

ПРОЕКТЫ

Поттериана

Особенности проекта:

- Интерактивный пользовательский интерфейс, включающий элементы перелистывания, выпадающие элементы и удобные кнопки, для динамичного и увлекательного взаимодействия с контентом
- Адаптированность структуры сайта для мобильных устройств
- Лаконичный и приятный для восприятия дизайн

ПРОЕКТЫ

Поттериана

```
1  <!doctype html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport"
6          content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0,
7      <meta http-equiv="X-UA-Compatible" content="ie=edge">
8      <title>Potteniana</title>
9      <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel='10
10     <style>
11         .carousel-inner {
12             display: flex;
13             align-items: center;
14             height: 600px; /* Установите фиксированную высоту для карусели */
15         }
16         .carousel-item img {
17             max-height: 100%;
18             height: auto;
19             margin: auto;
20         }
21         .carousel-control-prev-icon,
22         .carousel-control-next-icon {
23             background-color: black; /* Установите цвет иконок */
24             border-radius: 50%; /* Сделать иконки круглыми */
25             padding: 10px; /* Добавить внутренний отступ */
52     <body>
53
54         <div class="container-fluid bg-body-secondary fixed-top d-flex align-items-center" style="height: 60px">
55             <span class="navbar-brand mt-3 h1 text-dark fs-1" style="font-family: 'Century Gothic'>
56                 
57                 Поттериана
58             </span>
59         </div>
60
61         <!-- Navigation bar -->
62         <nav class="navbar fixed-top bg-white" style="margin-top: 100px;">
63             <div class="container-fluid">
64                 <div class="row w-100">
65                     <div class="col-md-8 d-flex align-items-center">
66                         <ul class="nav nav-tabs me-auto" id="myTab" role="tablist">
67                             <li class="nav-item" role="presentation">
68                                 <a class="nav-link active" id="home-tab" data-bs-toggle="tab" href="#home">Home</a>
69                             </li>
70                             <li class="nav-item" role="presentation">
71                                 <a class="nav-link" id="history-tab" data-bs-toggle="tab" href="#history">History</a>
72                             </li>
73                             <li class="nav-item" role="presentation">
74                                 <a class="nav-link" id="characters-tab" data-bs-toggle="tab" href="#characters">Characters</a>
75                             </li>
76                         </ul>
77                     </div>
78                 </div>
79             </div>
80         </nav>
81
82         <div class="container">
83             <div class="row w-100">
84                 <div class="col-md-8 d-flex align-items-center">
85                     <div class="carousel slide" data-bs-ride="carousel" data-bs-interval="3000">
86                         <div class="carousel-inner">
87                             <div class="carousel-item active">
88                                 
89                             </div>
90                             <div class="carousel-item">
91                                 
92                             </div>
93                             <div class="carousel-item">
94                                 
95                             </div>
96                         </div>
97                         <div class="carousel-control w-100" style="text-align: center; margin-top: -10px">
98                             <span class="carousel-control-prev" data-bs-target="#myCarousel" data-bs-slide="prev"></span>
99                             <span class="carousel-control-next" data-bs-target="#myCarousel" data-bs-slide="next"></span>
100                        </div>
101                    </div>
102                </div>
103            </div>
104        </div>
105    </body>
```

html > body > nav.navbar.fixed-top.bg-white > div.container-fluid > div.row.w-100 > div.col-md-8.d-flex.align-items-cent

ПРОЕКТЫ

Поттериана

```
<div class="tab-pane fade" id="history" role="tabpanel" aria-labelledby="history-tab">
  <div class="accordion" id="accordionExample">
    <div class="accordion-item">
      <h2 class="accordion-header">
        <button class="accordion-button" type="button" data-bs-toggle="collapse" data-bs-target="#collapseOne" aria-expanded="false" aria-controls="collapseOne">
          Философский камень
        </button>
      </h2>
      <div id="collapseOne" class="accordion-collapse collapse show" data-bs-parent="#accordionExample">
        <div class="accordion-body">
          Родителей Гарри Поттера убил Волан-де-Морт. Мальчика отдали на воспитание к племяннику Альбусу Дамблдору, который относился к мальчику плохо. Когда Гарри повзрослел, он начал получать письма на имя мальчика. Дядя увёз семью в лачугу посреди моря.
          <br>В их убежище вломился великан и заявил, что Гарри – волшебник. Гарри был принят в школу Хогвартса. Вместе с новым знакомым, Гарри купил всё необходимое для учебы в школе. В пути в школу Гарри встретил Северуса Снейпа, который стал его первым учителем.
          <br>С момента поступления Поттера в школу происходило много интересных событий. Однажды Гарри и его друзья встретили огромного тролля, который напал на подругу Гарри. Ему и другим ученикам пришлось сражаться с троллем.
          <br>В промежутке между происходящими событиями, Гарри с друзьями встретили великанов, находившихся в лесу. Великаны были очень голодны и начали разгрызать деревья. Гарри и его друзья помогли им найти пищу.
          <br>Гарри и его друзья также помогали ученикам из других семей. Однажды они помогли ученику из Слизерина, который случайно обнаружил Гарри в лесу. Ученик был очень благодарен Гарри и его друзьям.
          <br>Гарри и его друзья также помогали ученикам из других семей. Однажды они помогли ученику из Слизерина, который случайно обнаружил Гарри в лесу. Ученик был очень благодарен Гарри и его друзьям.
```

ПРОЕКТЫ

Поттериана

Поттериана

Главная История Герои Поиск Поиск



«Гарри Поттер» (англ. «Harry Potter») — серия романов, написанная британской писательницей Дж. К. Роулинг. Книги представляют собой хронику приключений юного волшебника Гарри Поттера, а также его друзей Рона Уизли и Гермионы Грейнджа, обучающихся в школе чародейства и волшебства Хогвартс. Основной сюжет посвящён противостоянию Гарри и тёмного волшебника по имени лорд Волан-де-Морт, в чьи цели входит обретение бессмертия и порабощение магического мира.

Поттериана

Главная История Герои Поиск Поиск

- Философский камень
- Тайная комната
- Узник Азкабана
- Кубок огня

Семья Уизли забрала Гарри от его родственников, и они все вместе отправились на Чемпионат мира по квиддичу. После матча тёмные маги напали на волшебников, ночующих в лагере. К тому же некто выпустил в небо Чёрную метку лорда Волан-де-Морта. После случившегося Гарри и его друзья приехали в Хогвартс. Директор объявила всем, что в этом году состоится Турнир Трёх Волшебников — испытание для трёх учащихся старше семнадцати лет. Также в школу прибыл новый учитель, а потом ученики двух других школ волшебства для участия в соревновании.

В день выбора участников Турнира специальный Кубок выбросил четвёртое имя — Гарри Поттера.

Мальчик не без труда справился с первым испытанием — сражение с драконом. А потом и со вторым — заплыл в озеро для спасения друга.

Третьим заданием стал лабиринт. Необходимо было пройти его и завладеть Кубком. Гарри и ещё один чемпион вдвоём добежали до приза и вместе коснулись его. Кубок оказался порталом, который перенёс их на кладбище. Здесь Поттер встретился с Волан-де-Мортом. Друга Гарри тут же убили, а его самого заставили смотреть, как Лорд принимает свой настоящий облик.

Далее последовал бой между Волан-де-Мортом и Поттером. Гарри помогли духи людей,

Поттериана

Главная История Герои Поиск Поиск



Гарри Джеймс Поттер
Главный герой Поттериана, одноклассник и лучший друг Рона Уизли и Гермионы Грейнджа, член Золотого Трио. Самый знаменитый студент Хогвартса за последние сто лет. Первый волшебник, которому удалось противостоять смертельному проклятию «Авада Кедавра», благодаря чему он стал знаменитым и получил прозвище «Мальчик, Который Выйти».



Гермиона Джин Грейнджа
Одна из главных героинь Поттериана, подруга и однокурсница Гарри Поттера и Рона Уизли, член Золотого Трио. Единственная дочь мистера Грейнджа и его жены. Играет важную роль во всех событиях, которые происходят в жизни Гарри.

Внешний вид Характер

ПРОЕКТЫ

Today Finance Bot

Задачи проекта:

Разработать бот в телеграм, который будет выдавать актуальные новости международного рынка ценных бумаг.

Описание проекта:

Today Finance Bot — это телеграм-бот, который с помощью API дает пользователям доступ к показателям международных компаний и последним новостям мира инвестиций. Информация будет полезна как инвесторам, так и людям, регулярно интересующимся рынком ценных бумаг.

ПРОЕКТЫ

Today Finance Bot

Особенности проекта:

- Доступ к актуальной информации рынка ценных бумаг через API
- Возможность получения автоматически последних пяти новостей или конкретного их количества с помощью специальной команды
- Получение последних данных и показателей по компании по тикеру – краткому названию акции, облигации или другой ценной бумаги на бирже

ПРОЕКТЫ

Today Finance Bot

```
s_news.py
import aiohttp
from aiogram import Bot, Dispatcher, F
from aiogram.filters import CommandStart, Command, CommandObject
from aiogram.types import Message
import requests
import logging
from googletrans import Translator

from config import TOKEN, ALPHA_VANTAGE_API_KEY

bot = Bot(token=TOKEN)
dp = Dispatcher()
translator = Translator()

logging.basicConfig(level=logging.INFO)

3 usages  prolijst
def translate(text):
    try:
        translated = translator.translate(text, dest='ru', src='en')
        return translated.text
    except Exception as e:
        logging.error(f"Ошибка при получении данных: {e}")
        return "Произошла ошибка при переводе текста."
end_company_overview()

ks_news.py
1 usage  prolijst
def get_market_news(number):
    url = f'https://www.alphavantage.co/query?function=NEWS_SENTIMENT&apikey={ALPHA_VANTAGE_API_KEY}'
    try:
        number = int(number)
    except ValueError:
        number = 5
    try:
        response = requests.get(url)
        response.raise_for_status()
        data = response.json()
        if 'feed' in data:
            news_items = data['feed'][:number] # Количество новостей в запросе
            news_list = []
            for item in news_items:
                title = item.get('title', 'No title')
                summary = item.get('summary', 'No summary')
                url = item.get('url', 'No URL')
                news_list.append(f'{translate(title)}\n{translate(summary)}\nПодробнее: {url}\n')
            stocks_news = "\n\n".join(news_list)
            return stocks_news
        else:
            return "Не удалось получить данные о рынке. Пожалуйста, попробуйте позже."
    except Exception as e:
        logging.error(f"Ошибка при получении данных: {e}")
        return "Произошла ошибка при получении данных."
end_company_overview()

cs_news.py
1 usage  prolijst
def get_company_overview(symbol):
    url = f'https://www.alphavantage.co/query?function=OVERVIEW&symbol={symbol}&apikey={ALPHA_VANTAGE_API_KEY}'
    try:
        response = requests.get(url)
        response.raise_for_status()
        data = response.json()
        if data:
            name = data.get('Name', 'No Name')
            description = data.get('Description', 'No Description')
            market_cap = data.get('MarketCapitalization', 'No Market Cap')
            pe_ratio = data.get('PERatio', 'No PE Ratio')
            return f"Компания: {name}\nОписание: {translate(description)}\n\n" \
                   f"Рыночная капитализация: {market_cap}\nP/E Ratio: {pe_ratio}"
        else:
            return "Не удалось получить данные о компании. Пожалуйста, попробуйте позже."
    except Exception as e:
        logging.error(f"Ошибка при получении данных: {e}")
        return "Произошла ошибка при запросе данных о компании."
end_company_overview()

@dp.message(CommandStart())
async def start(message: Message):
    await message.answer("Привет! Я бот, который может предоставить информацию о рынке ценных бумаг.\n\nЧтобы начать, выберите один из вариантов ниже: /news или /company_overview")
```

ПРОЕКТЫ

Today Finance Bot

Привет! Я бот, который может предоставить информацию о рынке ценных бумаг.
Используй команду `/market` с числом N, чтобы получить N последних новостей.
Используй команду `/company` с символом акции, чтобы получить данные о компании.

Компания: Apple Inc
Описание: Apple Inc. - американская многонациональная технологическая компания, которая специализируется на потребительской электронике, компьютерном программном обеспечении и онлайн-услугах. Apple является крупнейшей в мире технологической компанией по выручке (на общую сумму 274,5 млрд долларов в 2020 году) и с января 2021 года, самая ценная компания в мире. По состоянию на 2021 год Apple является четвертым по величине поставщиком ПК в мире по продажам блоков и четвертым по величине производителем смартфонов. Это одна из крупнейших американских компаний по информационным технологиям, наряду с Amazon, Google, Microsoft и Facebook.
Рыночная капитализация: 3469575717000
P/E Ratio: 34.73

`/start` 0:50 ✓
`/company AAPL` 0:50 ✓

Что случилось с Tesla Stock сегодня? Тесла (NASDAQ: TSLA)
Акции TESLA, Inc. TSLA поднялись выше в понедельник, так как инвесторы ожидают показателей доставки компании в третьем квартале и выпуска ее роботакси.
Подробнее:
<https://www.benzinga.com/news/24/09/40992274/what-happened-with-tesla-stock-today>

Conference Conference Conference Conference Conference - Oceanfirst Financial (NASDAQ: OCFC) планирует доходы.
Red Bank, N.J., 23 сентября 2024 г. (Globe Newswire) - Oceanfirst Financial Corp. OCFC, холдинговая компания для Bank Oceanfirst Bank, объявила, что выпустит свой выпуск прибыли за квартал, закончившийся 30 сентября 2024 года в четверг, октябрь 17, 2024 после закрытия рынка.
Подробнее:
<https://www.benzinga.com/pressreleases/24/09/g40992162/oceanfirst-financial-corp-schedules-earnings-conference-call>

Вот сколько вы бы сделали владеющим акциями подшипников RBC за последние 15 лет - подшипники RBC (NYSE: RBC)
RBC Bearings RBC превзошел рынок за последние 15 лет на 6,54% на годовой основе, обеспечивая среднегодовой доход в 18,4%. В настоящее время рыночная капитализация RBC имеет рыночную капитализацию в 8,67 млрд долларов.
Подробнее:
<https://www.benzinga.com/insights/news/24/09/40991978/heres-how-much-you-would-have-made-owning-rbc-bearings-stock-in-the-last-15-years>

`/market 3` 0:54 ✓
`/company MSFT` 0:51 ✓

Рыночные киты и их недавние ставки на варианты VST - Vistra (NYSE: VST)
Финансовые гиганты сделали заметный бычий шаг на Виштрэ. Наш анализ истории вариантов для Vistra VST выявил 96 необычных сделок. Задумываясь над деталями, мы обнаружили, что 41% трейдеров были бычьи, а 40% показали медвежьими тенденциями.
Подробнее:
<https://www.benzinga.com/insights/options/24/09/40991659/markt-whales-and-their-recent-bets-on-vst-options>

Рыночные киты и их недавние ставки на варианты ZIM - интегрированная доставка Zim (NYSE: ZIM)
Инвесторы с большим количеством денег, чтобы потратить, заняли оптимистическую позицию на Zim Integrated Shipping Zim. Мы заметили это сегодня, когда сделки появились в общедоступной истории вариантов, которые мы отслеживаем здесь, в Бензинга. Независимо от того, являются ли это учреждениями или просто богатыми людьми, мы не знаем.
Подробнее:
<https://www.benzinga.com/insights/options/24/09/40991658/markt-whales-and-their-recent-bets-on-zim-options>

Глядя на недавнее необычное занятие Penn Entertainment - Penn Entertainment (NASDAQ: Penn)
Инвесторы с большим количеством денег, чтобы потратить, заняли медвежью позицию на Penn Entertainment Penn. Мы заметили это сегодня, когда сделки появились в общедоступной истории вариантов, которые мы отслеживаем здесь, в Бензинга. Независимо от того, являются ли это учреждениями

`/market` 0:30 ✓

ПРОЕКТЫ

Magic Tricks Shop

Задачи проекта:

Разработать веб-приложение для управления магазином товаров для фокусников и создать телеграм-бота для взаимодействия с магазином.

Описание проекта:

Magic Tricks Shop — это комплексное решение для магазина товаров для фокусников, включающее веб-приложение на Django и телеграм-бота на Aiogram. Веб-приложение предоставляет пользователям возможность регистрироваться, авторизоваться, управлять своими заказами и оставлять отзывы, а также включает в себя панель аналитики для администраторов. Оно включает в себя три модуля: для работы с товарами, для управления заказами и для обслуживания пользователей. Телеграм-бот обеспечивает удобный доступ к функциям магазина, включая верификацию пользователей и просмотр аналитики, что позволяет гибко управлять магазином из любого места.

ПРОЕКТЫ

Magic Tricks Shop

Особенности проекта:

- Веб-приложение оптимизировано для работы на различных устройствах, включая смартфоны и планшеты, что позволяет пользователям комфортно использовать его в любом месте
- Администраторы имеют доступ к подробной аналитической информации, что помогает в принятии обоснованных бизнес-решений
- Веб-приложение обладает привлекательным и интуитивно понятным дизайном, который улучшает пользовательский опыт и делает навигацию по сайту удобной

ПРОЕКТЫ

Magic Tricks Shop

ПРОЕКТЫ

Magic Tricks Shop

The screenshot shows a code editor with three tabs open, all corresponding to the file `shopOrders\views.py`:

- `shopGoods\views.py`: Contains logic for adding items to a cart and removing them.
- `shopOrders\views.py`: Contains logic for placing orders and managing order history.
- `shopOrders\views.py`: Another instance of the same file, likely a backup or a different view of the same code.

The code in these files is as follows:

```
shopGoods\views.py
def add_to_cart(request):
    if request.method == 'POST':
        product_id = request.POST.get('product_id')
        cart = request.session.get('cart', {})

        if product_id in cart:
            cart[product_id] += 1
        else:
            cart[product_id] = 1

        request.session['cart'] = cart
        messages.success(request, message='Товар добавлен в корзину')

    return redirect('shopUsers:cart')

def remove_from_cart(request, product_id):
    cart = request.session.get('cart', {})

    # Прорабазуем product_id в строку, поскольку клик
    str_product_id = str(product_id)

    if str_product_id in cart:
        del cart[str_product_id]

    return redirect('shopUsers:cart')
```

```
shopOrders\views.py
def order(request):
    if not request.session.get('user_id'):
        return redirect('shopUsers:login') # Проверка, что пользователь вошел

    # Создаем список продуктов с количеством
    products_with_quantity = []

    # Получаем товары из корзины
    cart_items = request.session.get('cart', {})
    products = Product.objects.filter(id__in=cart_items.keys())

    # Перебираем продукты и добавляем их в products_with_quantity
    for product in products:
        quantity = cart_items.get(str(product.id), 0)
        products_with_quantity.append({
            'product': product,
            'quantity': quantity,
        })

    if request.method == 'POST':
        order_form = OrderForm(request.POST)
        if order_form.is_valid():
            order = order_form.save(commit=False)
            order.user = User.objects.get(id=request.session['user_id'])

            context = {
                'orders': orders
            }

            return render(request, template_name='shopOrders/order_confirmation.html', context)
```

The project structure on the left includes `MagicTricksShop_Project`, `django_project`, `MagicTricksShop`, `shopGoods`, `shopOrders`, `shopUsers`, `db.sqlite3`, `manage.py`, `main_dj.py`, and `telegram_bot`.

ПРОЕКТЫ

Magic Tricks Shop

The screenshot shows a code editor interface with multiple tabs open, displaying Python code for a Django project named 'MagicTricksShop_Project'. The project structure is visible in the left sidebar, showing nested folders like 'MagicTricksShop' and 'shopUsers'.

The main code editor area displays the content of `shopUsers/views.py`. The code handles user registration, login, cabinet access, profile editing, and logout. It uses Django's User model and form classes, along with messages and redirects. A separate block of code for a 'cart' view is also shown, which filters products by their ID and calculates total price.

```
File Edit View Navigate Code Refactor Run Tools Git Window Help MagicTricksShop_Project - shopUser... Run Tools Git Window Help MagicTricksShop_Project - shopUsers\views.py Run Tools Git Window Help MagicTricksShop_Project - shopUsers\views.py Project Pull Requests Bookmarks Structure Storm Sections Git Python Packages TODO Problems Terminal Services Storm Sections Python Console Problems Terminal Services Storm Sections 1:1 CRLFs 1:1 CRLF
```

```
def register(request):
    error = ''
    if request.method == 'POST':
        form = UserForm(request.POST) # Сюда сохраняется информация о пользователе
        if form.is_valid():
            form.save()
            return redirect('shopUsers:login')
        else:
            error = "Данные были заполнены некорректно"
    form = UserForm()
    return render(request, template_name='shopUsers/register.html')

def login(request):
    if request.method == 'POST':
        name = request.POST.get('name')
        phone = request.POST.get('phone')

        # Проверка наличия пользователя в базе данных
        try:
            user = User.objects.get(name=name, phone=phone)
            # Успешный вход: сохраним id пользователя в сессии
            request.session['user_id'] = user.id
            return redirect('shopUsers:cabinet')
        except User.DoesNotExist:
            messages.error(request, message='Пользователь не найден.')
            return redirect('shopUsers:login')

    if request.method == 'POST':
        form = UserForm(request.POST, instance=user)
        if form.is_valid():
            form.save()
            messages.success(request, message='Ваш профиль был успешно обновлен!')
            return redirect('shopUsers:cabinet')
        else:
```

```
def cabinet(request):
    return render(request, template_name='shopUsers/cabinet.html')

def edit_profile(request):
    user_id = request.session.get('user_id')
    if not user_id:
        messages.error(request, message='Вам необходимо войти в систему для редактирования профиля')
        return redirect('shopUsers:login')

    try:
        user = User.objects.get(id=user_id)
    except User.DoesNotExist:
        messages.error(request, message='Пользователь не найден.')
        return redirect('shopUsers:login')

    if request.method == 'POST':
        form = UserForm(request.POST, instance=user)
        if form.is_valid():
            form.save()
            messages.success(request, message='Ваш профиль был успешно обновлен!')
            return redirect('shopUsers:cabinet')
        else:
```

```
def logout(request):
    auth_logout(request) # Завершаем сеанс пользователя
    request.session.flush()
    return render(request, template_name='shopUsers/logout_success.html')

def cart(request):
    cart_items = request.session.get('cart', {})
    products = Product.objects.filter(id__in=cart_items.keys())

    # Создаем список продуктов с количеством
    products_with_quantity = []
    for product in products:
        quantity = cart_items.get(str(product.id), 0)
        products_with_quantity.append({
            'product': product,
            'quantity': quantity,
        })

    total_price = sum(item['quantity'] * item['product'].price for item in products_with_quantity)

    context = {
        'products_with_quantity': products_with_quantity,
        'total_price': total_price,
```

ПРОЕКТЫ

Magic Tricks Shop

The screenshot shows the PyCharm IDE interface with two code editors open. The left editor contains `main_tb.py` and the right editor contains `database.py`. The project structure on the left shows the `MagicTricksShop_Project` folder, which includes a `django_project`, a `telegram_bot` directory containing `venv`, `config.py`, `database.py`, `keyboards.py`, `main_tb.py`, `test_bot.py`, `.gitattributes`, `.gitignore`, `LICENSE`, `README.md`, and `requirements.txt`. The `External Libraries` and `Scratches and Consoles` sections are also visible.

`main_tb.py` code:

```
80     @dp.message(lambda message: ':' in message.text)
81     async def verify_admin(message: Message, state: FSMContext):
82         username, password = message.text.split(':', maxsplit=1)
83         admin = get_admin_by_credentials(username, password)
84         if admin:
85             await state.update_data(username=admin[4]) # Сохраняем имя администратора в состояние
86             await message.answer(text=f"Добро пожаловать, {admin[4]}!", reply_markup=admin_keyboard())
87         else:
88             await message.answer("Неверные учетные данные. Попробуйте снова.")
89
90     @dp.message(F.text == 'Мои заказы', StateFilter(UserState.verified))
91     async def show_user_orders(message: Message, state: FSMContext):
92         data = await state.get_data()
93         user_id = data.get("user_id")
94         logging.info(f"User ID from state: {user_id}")
95         if user_id:
96             orders = get_orders_by_user(user_id)
97             if orders:
98                 for order in orders:
99                     order_details = get_order_details(order[0])
100                    details = "\n".join([f"{item[1]} x {item[0]} - {item[2]} ₽" for item in order_details])
101                    await message.answer(f"Заказ #{order[0]}:\n{details}\nСтатус: {order[1]}")
102            else:
103                await message.answer("У вас нет заказов.")
```

`database.py` code:

```
29     2 usages ▾ browser
30     def get_admin_by_credentials(username, password):
31         conn = sqlite3.connect(DB_PATH)
32         cursor = conn.cursor()
33         # Получаем пароль из базы данных
34         cursor.execute(sql: "SELECT * FROM auth_user WHERE username=?", parameters: (username,))
35         admin = cursor.fetchone()
36         conn.close()
37
38         if admin:
39             # Предполагается, что хеш пароля находится в столбце с индексом 1 (замените на нужный индекс)
40             hashed_password = admin[1]
41             if check_password(password, hashed_password):
42                 return admin
43             return None
44
45     2 usages ▾ prolisbet
46     def get_orders_by_user(user_id):
47         conn = sqlite3.connect(DB_PATH)
48         cursor = conn.cursor()
49         cursor.execute(sql: "SELECT * FROM shopOrders_order WHERE user_id=?", parameters: (user_id,))
50         orders = cursor.fetchall()
51         conn.close()
52         return orders
```

ПРОЕКТЫ

Magic Tricks Shop

127.0.0.1:8000 127.0.0.1:8000/catal... 127.0.0.1:8000/u...

Яндекс Pinterest (650) Входящие - li... Яндекс Pinterest (650) Входящие - li... Яндекс Pinterest (650) Входящие - li...

Magic Tricks Shop Главная страница Каталог товаров Войти Корзина Главная страница Каталог товаров Войти Корзина Главная страница Каталог товаров Войти Корзина

Добро пожаловать в Magic Tricks Shop!

Погрузитесь в захватывающий мир иллюзий и волшебства вместе с нами! Magic Tricks Shop — это ваше уникальное пространство, где каждый, от начинающего до профессионального иллюзиониста, найдет всё необходимое для создания незабываемых магических шоу.

Следите за нашими обновлениями, акциями и специальными предложениями. Подписывайтесь на нашу рассылку и соцсети, чтобы всегда быть в курсе новинок и событий мира магии.

Magic Tricks Shop — откройте для себя мир волшебства уже сегодня!

Каталог магических товаров

Веревка Волшебника
Комплект веревок, которые кажутся обычными, но обладают магическими свойствами.
899,00 руб.
★★★★★(1 отзыв)
[Заказать](#)

Карты Иллюзиониста
Колода карт с уникальными свойствами для выполнения невероятных трюков.
1199,00 руб.
★★★★★(1 отзыв)
[Заказать](#)

Книга Чудес
Обучающее руководство с секретами великих фокусников.
799,00 руб.
★★★★★(1 отзыв)
[Заказать](#)

Регистрация магического адепта

Имя Имя

Email Email

Телефон (с кодом страны) Телефон

Адрес Адрес

[Погрузиться в магию](#) [Вы уже магический адепт?](#)

Magic Tricks Shop © 2024 | Все права защищены. | Политика конфиденциальности | Свяжитесь с нами | Следуйте за нами: [Социальные сети]

ПРОЕКТЫ

Magic Tricks Shop

127.0.0.1:8000/

127.0.0.1:8000/

127.0.0.1:8000/

Яндекс Pinterest (650) Входящие - li... Входящие (10) - liz...

Magic Tricks Shop

Корзина с магией

Монеты Призрака - 1 шт. 999,00 ₽ Удалить

Очки Разоблачителя - 1 шт. 2499,00 ₽ Удалить

Итоговая сумма: 3498,00 ₽

Оформление заказа

Адрес доставки:
ул. Магическая, дом 5, кв. 10

Товары в заказе:

Монеты Призрака - 1 шт.

Очки Разоблачителя - 1 шт.

Итоговая сумма: 3498,00 ₽

Отправить заказ

История магических заказов

Заказ № 1
Дата: 25 октября 2024 г. 0:50
Статус: Завершен
Адрес доставки: ул. Магическая, дом 5, кв. 10

1 x Порошок Дракона 1399,00 ₽

Заказ № 2
Дата: 25 октября 2024 г. 21:30
Статус: Завершен
Адрес доставки: ул. Магическая, дом 5, кв. 10

1 x Шкатулка Тайны 1999,00 ₽

1 x Плащ-невидимка 119999,00 ₽

2 x Книга Чудес 1598,00 ₽

Заказ № 7
Дата: 12 декабря 2024 г. 22:59
Статус: Ожидается
Адрес доставки: ул. Магическая, дом 5, кв. 10

1 x Монеты Призрака 999,00 ₽

1 x Очки Разоблачителя 2499,00 ₽

© 2024 | Все права защищены. | Политика конфиденциальности | Свяжитесь с нами | Следуйте за нами: [Социальные сети]

© 2024 | Все права защищены. | Политика конфиденциальности | Свяжитесь с нами | Следуйте за нами: [Социальные сети]

ПРОЕКТЫ

Magic Tricks Shop

The image displays three side-by-side screenshots of a web application for a magic shop. The application has a dark-themed header with the logo 'Magic Tricks Shop' and navigation links for 'Главная страница', 'Каталог товаров', 'Личный кабинет', and 'Корзина'. The browser's address bar shows the URL '127.0.0.1:8000/pr...'. The left screenshot shows a product page for a 'Шкатулка Тайны' (Magic Box) with a price of 1999,00 руб. and no reviews. The middle screenshot shows a feedback section titled 'Оцените нашу магию' (Rate our magic) for the same product, featuring a 5-star rating and a text input field with the placeholder 'Отличный подарок!' (Great gift!). The right screenshot shows a detailed product page for the 'Шкатулка Тайны' with a price of 1999,00 руб., a 5-star rating from 1 review, and a summary: 'Магическая шкатулка, открывающаяся только по воле фокусника.' Below this are buttons for 'Заказать снова' (Order again) and 'Оставить отзыв' (Leave a review). The review section lists a comment by 'Лиза:' with a 5-star rating, the text 'Отличный подарок!', and the timestamp '12 декабря 2024 г. 23:04'.

ПРОЕКТЫ

Magic Tricks Shop

The image displays three side-by-side screenshots of a Django-based administrative interface for a shop system, likely named "Magic Tricks Shop".

Screenshot 1: Order Management (Left)

This screenshot shows the "Edit Order" page for "Order #7 from Liza". The status dropdown menu is open, showing options: "Ожидается" (Pending), "В обработке" (In Progress), "Завершен" (Completed), and "Отменен" (Cancelled). The "Completed" option is highlighted. The page includes a sidebar with sections like SHOPGOODS, SHOPORDERS, and SHOPUSERS.

Screenshot 2: Order Management (Middle)

This screenshot shows the "Edit Order" page again, but the status dropdown is closed, and the "Completed" status is selected. A green notification bar at the top right indicates: "Общее количество завершенных заказов: 6 Общая прибыль: 135787 руб." (Total completed orders: 6 Total profit: 135787 rub.).

Screenshot 3: Reporting (Right)

This screenshot shows the "Report" page for December 17, 2024. It displays a summary: "Порошок Дракона: 1 шт., Выручка: 1399 руб." (Dragon Powder: 1 unit, Revenue: 1399 rub.). It also shows breakdowns by order status and user.

ПРОЕКТЫ

Magic Tricks Shop

MagicTricksShop_Bot bot

Выберите вашу роль: 18:53

Покупатель 18:53 ✓

Введите ваше имя пользователя: 18:53

Елизавета 18:53 ✓

Добро пожаловать, Елизавета! 18:53

Мои заказы 18:53 ✓

Заказ #5:
Карты Иллюзиониста x1 - 1199 руб.
Статус: completed 18:53

Заказ #6:
Веревка Волшебника x1 - 899 руб.
Очки Разоблачителя x1 - 2499 руб.
Статус: processing 18:53

/start начало работы
/help список команд
/info информация о магазине

Bot Info

MagicTricksShop_Bot bot

Выберите вашу роль: 19:18

Админ 19:18 ✓

Введите ваши учетные данные в формате username:password:

magicadmin:magicshop123 19:19 ✓

Привет! Я бот магазина магических трюков Magic Tricks Shop

Description

@MagicTricksShopBot Username

Notifications

Add to Group

Bot Help

Bot Privacy Policy

Report

Stop and block bot

MagicTricksShop_Bot bot

Выберите вашу роль: 19:18

Админ 19:18 ✓

Введите ваши учетные данные в формате username:password:

magicadmin:magicshop123 19:19 ✓

Добро пожаловать, magicadmin! 19:19

Все заказы 19:19 ✓

Заказ #1:
Порошок Дракона x1 - 1399 руб.
Статус: completed 19:19

Заказ #2:
Шкатулка Тайны x1 - 1999 руб.
Плащ-невидимка x1 - 119999 руб.
Книга Чудес x2 - 799 руб.
Статус: completed 19:19

Заказ #3:
Монеты Призрака x1 - 999 руб.
Статус: completed 19:19

Заказ #4:
Карты Иллюзиониста x1 - 1199 руб.
Веревка Волшебника x1 - 899 руб.
Палочка Чародея x1 - 1599 руб.
Порошок Дракона x1 - 1399 руб.
Статус: processing 19:19

MagicTricksShop_Bot bot

Заказ #6:
Веревка Волшебника x1 - 899 руб.
Очки Разоблачителя x1 - 2499 руб.
Статус: processing 19:19

Отчеты 19:19 ✓

Дата: 2024-10-27
Данные продаж: Монеты Призрака: 1 шт., Выручка: 999 руб.
Прибыль: 999 руб.
Расходы: 0 руб. 19:19

Дата: 2024-10-27
Данные продаж: Монеты Призрака: 1 шт., Выручка: 999 руб.
Прибыль: 999 руб.
Расходы: 0 руб. 19:19

Дата: 2024-10-27
Данные продаж: Веревка Волшебника: 1 шт., Выручка: 899 руб.; Карты Иллюзиониста: 1 шт., Выручка: 1199 руб.; Палочка Чародея: 1 шт., Выручка: 1599 руб.; Порошок Дракона: 1 шт., Выручка: 1399 руб.
Прибыль: 5096 руб.
Расходы: 0 руб. 19:19

Дата: 2024-10-27
Данные продаж: Веревка Волшебника: 1 шт., Выручка: 899 руб.; Карты Иллюзиониста: 1 шт., Выручка: 1199 руб.; Палочка Чародея: 1 шт., Выручка: 1599 руб.; Порошок Дракона: 1 шт., Выручка: 1399 руб. 19:19

MagicTricksShop_Bot bot

Аналитика 19:19 ✓

Общее количество завершенных заказов: 5
Общая прибыль: 132289 руб. 19:19

Назад 19:19 ✓

/help 19:38 ✓

Этот бот умеет выполнять команды:
/start
/help
/info 19:38

Выберите вашу роль: 19:38

/info 19:41 ✓

Добро пожаловать в Magic Tricks Shop!
Погружайтесь в захватывающий мир иллюзий и волшебства вместе с нами!
Magic Tricks Shop — это ваше уникальное пространство, где каждый, от начинающего до профессионального иллюзиониста, найдет все необходимое для создания незабываемых магических шоу. Следите за нашими обновлениями, акциями и специальными предложениями.
Подписывайтесь на нашу рассылку и соцсети, чтобы всегда быть в курсе новинок

достижения

- Хакатон «Лидеры цифровой трансформации 2024»: участник проекта по теме «Сервис текстового поиска по медиаконтенту» для приложения Yappy.
Цель сервиса — извлечение текстовой информации из видео и предоставление возможности поиска по этой информации.
Основные функции сервиса:
 - загрузка видео
 - транскрибация аудиоряда, распознавание речи с помощью модели Whisper
 - разбивка видео на кадры с помощью библиотеки OpenCV
 - распознавание символов и субтитров на извлеченных кадрах с помощью библиотеки EasyOCR (OCR – оптическое распознавание символов)
 - преобразование кадров в векторное представление с помощью модели CLIP
 - преобразование текстового запроса пользователя в векторное представление с помощью модели CLIP
 - поиск ближайших по значению векторов (изображения и запроса) с помощью библиотеки FAISS
 - выдача по запросу пользователя релевантных видео, как по результатам поиска ближайших векторов, так и по результатам сравнения запроса с аудиорядом видео и с субтитрами, присутствующими на кадрах

достижения

- **Телеграм-бот - трекер привычек:** участник командной разработки телеграм-бота со следующим функционалом: регистрация пользователей; выбор привычек из существующей базы данных или добавление своих; настройка ежедневных уведомлений о выполнении полезных привычек или об избегании вредных.
- **Телеграм-бот - Python Interviewer:** участник командной разработки телеграм-бота, созданного для подготовки начинающих программистов на Python к интервью в крупных компаниях. Бот регистрирует пользователя; задает вопросы из заранее созданной базы данных и принимает ответы в текстовом виде или в виде голосовых сообщений; выдает отчет со статистикой успешных ответов пользователя.

**ВСЕГДА ОТКРЫТА К НОВЫМ ПРОЕКТАМ И ВОЗМОЖНОСТИЯМ!
СВЯЗАТЬСЯ СО МНОЙ МОЖНО СЛЕДУЮЩИМ ОБРАЗОМ:**

Telegram: @pro_lisbet

E-mail: lizatrushina96@gmail.com
GitHub: <https://github.com/prolisbet>