Федеральное государственное автономное образовательное учреждение высшего образования

«МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий  
Кафедра «Информатика и информационные технологии»

Направление подготовки/ специальность: \_Информационные системы и технологии\_\_\_\_\_\_\_

ОТЧЕТ

по проектной практике

Студенты: Гавриш Александрина Алексеевна Группа: 241-334

Место прохождения практики: Московский Политех, кафедра \_Информатики и информационных технологий\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Отчет принят с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Дата \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель практики: \_Захаров Илья Андреевич\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Москва 2025

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

1. Общая информация о проекте:

* Название проекта
* Цели и задачи проекта

1. Общая характеристика деятельности организации *(заказчика проекта)*

* Наименование заказчика
* Организационная структура
* Описание деятельности

1. Описание задания по проектной практике
2. Описание достигнутых результатов по проектной практике

ЗАКЛЮЧЕНИЕ *(выводы о проделанной работе и оценка ценности выполненных задач для заказчика)*

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

ПРИЛОЖЕНИЯ *(при необходимости)*

**ВВЕДЕНИЕ.**

Современное образование всё больше опирается на цифровые технологии, создавая потребность в развитии ИТ-инфраструктуры вузов. В условиях стремительной цифровизации и роста объёмов информации особенно актуальны проекты, направленные на создание сервисов, упрощающих учебный и административный процессы.

Проектная практика, проведённая в рамках обучения, была направлена на разработку и реализацию цифровых решений для образовательного пространства вуза. В рамках практики велась работа над двумя задачами: созданием информационного сайта «ИТ-сервисы для Цифрового университета» и разработкой Telegram-бота «Рандомайзер», предназначенного для интерактивного взаимодействия с пользователями.

**1. Общая информация о проекте**

Название проекта: ИТ-сервисы для Цифрового университета

Цель проекта:

Создание информационного ресурса и инструмента автоматизации, которые будут служить удобными помощниками студентам, преподавателям и сотрудникам вуза в повседневной работе и обучении.

Задачи проекта:

Разработать и оформить статический сайт, отражающий основные цифровые ИТ-сервисы вуза.

Сконструировать архитектуру проекта на базе HTML и CSS с соблюдением адаптивности и пользовательской доступности.

Реализовать понятную навигацию, описания сервисов и интерфейс, ориентированный на пользователей.

Создать Telegram-бота, который будет предлагать пользователю случайное число в заданном диапазоне, с возможностью повторного выбора и гибкого взаимодействия через кнопки.

Закрепить навыки работы с системами контроля версий (Git), документирования (Markdown) и организации командной разработки.

**2. Общая характеристика деятельности организации (заказчика проекта)**

Наименование заказчика: Московский университет

Организационная структура:

Проектная команда условно делится на следующие группы:

Разработчики — отвечают за техническую реализацию сайта и Telegram-бота.

Контент-менеджеры — подбирают и структурируют информацию для сайта.

Тестировщики — проверяют корректность работы сервисов.

Кураторы — контролируют сроки и соответствие продукта требованиям заказчика.

Описание деятельности:

Университет инициирует проекты цифровой трансформации учебной среды. Среди приоритетных направлений — создание удобных сервисов для студентов и преподавателей: электронный дневник, расписание, облачные сервисы хранения, средства взаимодействия и уведомлений. Цель заказчика — упростить доступ к информации, минимизировать бюрократические процессы и повысить вовлечённость студентов.

**3. Описание задания по проектной практике**

В рамках практики были выполнены задания в соответствии с установленной структурой, включающей базовую часть и вариативное задание. Работа велась в двух направлениях: создание сайта и реализация Telegram-бота.

**3.1. Базовая часть задания**

1. Настройка Git и репозитория

Был создан репозиторий на GitHub.

Изучены базовые команды Git: clone, commit, push, branch, merge.

Работа велась в ветвях с обязательным созданием pull request'ов.

Папки были логично организованы: docs, src, bot, website, images, readme.

2. Написание документов в формате Markdown

Составлены документы:

README.md с описанием проекта;

main.py — код бота;

bot.md — описание работы бота;

site(папка с кодам HTML и CSS);

task(папка с заданием на проектную практику);

Оформление строго соответствовало синтаксису Markdown.

3. Создание статического сайта

Название: ИТ-сервисы для Цифрового университета

Языки: HTML5, CSS3

Разделы сайта:

Главная страница — краткое описание проекта, цель, актуальность;

О проекте — описание функционала, цифровые сервисы вуза;

Участники — участники проектной группы;

Журнал — описание этапов разработки и изменений;

Ресурсы — ссылки на репозиторий, использованные технологии и библиотеки.

Особенности реализации:

Использовались адаптивные технологии;

Сайт валиден по стандартам W3C;

Интерфейс визуально чистый, разделён на блоки;

Навигация интуитивно понятна, реализована через <nav> и якорные ссылки.

3.2. Вариативная часть — Telegram-бот “Рандомайзер”

Название: @Randomizer\_alia\_bot

Описание:

Бот, предлагающий пользователю случайное число в заданном диапазоне. Поддерживает кнопки для нового запроса, смены диапазона, повторной генерации.

Язык: Python

Библиотека: telebot (pyTelegramBotAPI)

Архитектура: FSM (finite state machine)

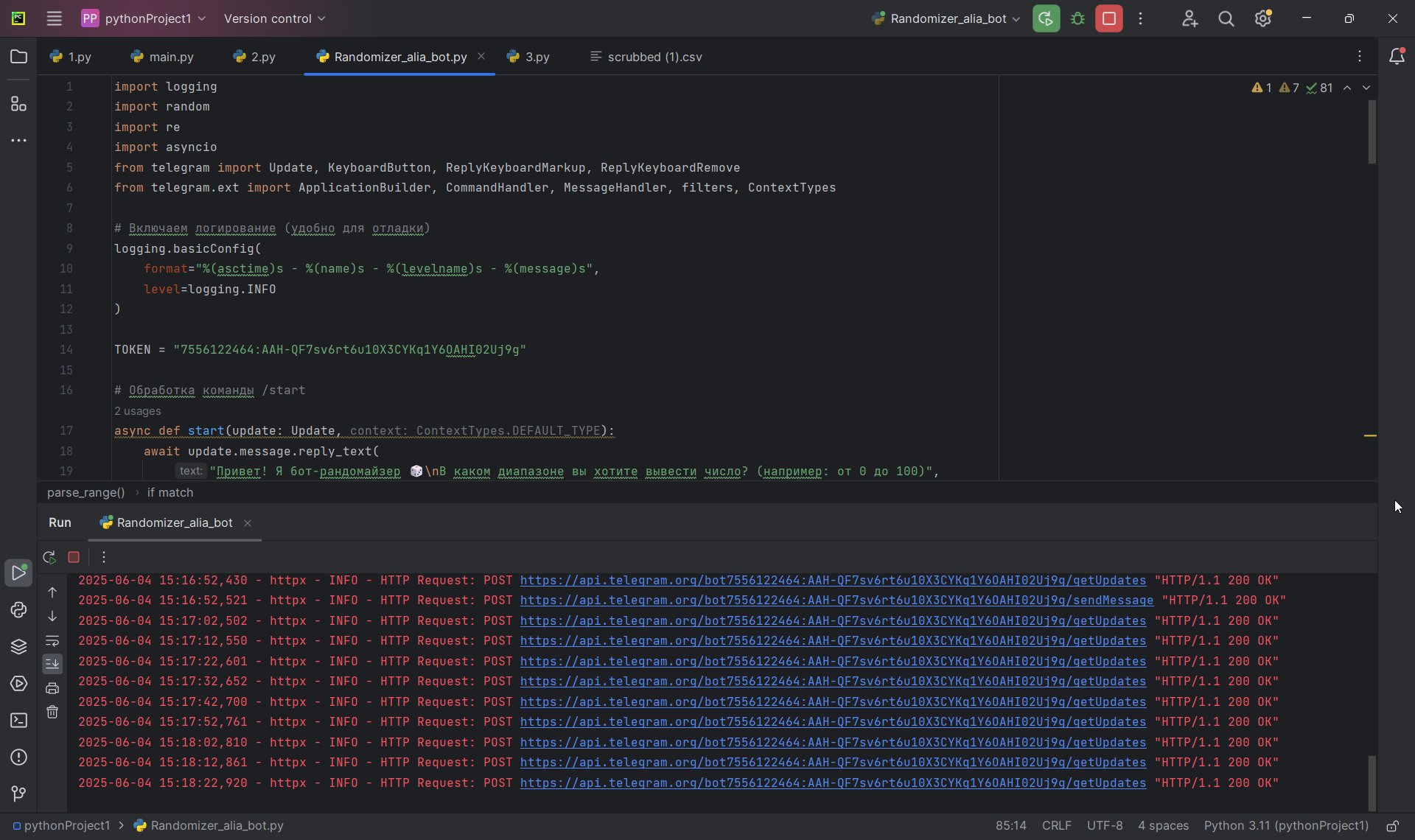


Рисунок 1: Работа кода

Код бота:

import logging  
import random  
import re  
import asyncio  
from telegram import Update, KeyboardButton, ReplyKeyboardMarkup, ReplyKeyboardRemove  
from telegram.ext import ApplicationBuilder, CommandHandler, MessageHandler, filters, ContextTypes  
  
# Включаем логирование (удобно для отладки)  
logging.basicConfig(  
 format="%(asctime)s - %(name)s - %(levelname)s - %(message)s",  
 level=logging.INFO  
)  
  
TOKEN = "7556122464:AAH-QF7sv6rt6u10X3CYKq1Y6OAHI02Uj9g"  
  
# Обработка команды /start  
async def start(update: Update, context: ContextTypes.DEFAULT\_TYPE):  
 await update.message.reply\_text(  
 "Привет! Я бот-рандомайзер 🎲\nВ каком диапазоне вы хотите вывести число? (например: от 0 до 100)",  
 reply\_markup=ReplyKeyboardRemove()  
 )  
  
# Универсальный обработчик сообщений  
async def handle\_message(update: Update, context: ContextTypes.DEFAULT\_TYPE):  
 text = update.message.text.strip()  
  
 if text.lower() == "старт":  
 await start(update, context)  
  
 elif text == "Новое число в том же диапазоне":  
 last\_range = context.user\_data.get("last\_range")  
 if last\_range:  
 number = random.randint(last\_range[0], last\_range[1])  
 await send\_number\_with\_buttons(update, context, number)  
 else:  
 await update.message.reply\_text("Сначала задайте диапазон.")  
  
 elif text == "Задать новый диапазон":  
 await update.message.reply\_text("Хорошо! Укажите новый диапазон (например: от 10 до 500):")  
  
 elif text == "Конец":  
 await update.message.reply\_text(  
 "Рад был вам помочь! Если когда-нибудь снова понадобится случайное число — я всегда здесь.",  
 reply\_markup=ReplyKeyboardMarkup([[KeyboardButton("Старт")]], resize\_keyboard=True)  
 )  
  
 else:  
 # Попытка распознать диапазон  
 min\_val, max\_val = parse\_range(text)  
  
 if min\_val is not None and max\_val is not None:  
 if min\_val > max\_val:  
 min\_val, max\_val = max\_val, min\_val  
 if abs(max\_val - min\_val) > 1\_000\_000:  
 await update.message.reply\_text("Слишком большой диапазон. Ограничьте его до миллиона.")  
 return  
  
 number = random.randint(min\_val, max\_val)  
 context.user\_data["last\_range"] = (min\_val, max\_val)  
 await send\_number\_with\_buttons(update, context, number)  
  
 else:  
 await update.message.reply\_text(  
 "Не удалось распознать диапазон. Пример: от 0 до 100, 0-100 ."  
 )  
  
# Отправка числа и кнопок  
async def send\_number\_with\_buttons(update, context, number):  
 keyboard = [  
 [KeyboardButton("Новое число в том же диапазоне")],  
 [KeyboardButton("Задать новый диапазон")],  
 [KeyboardButton("Конец")]  
 ]  
 await update.message.reply\_text(  
 f"🎲 Ваше число: {number}",  
 reply\_markup=ReplyKeyboardMarkup(keyboard, resize\_keyboard=True)  
 )  
  
# Функция парсинга диапазона  
def parse\_range(text):  
 text = text.lower().strip()  
  
 # Используем регулярные выражения  
 match = re.search(r"(\d+)\s\*(?:до|-|–|—|по)\s\*(\d+)", text)  
 if match:  
 return int(match.group(1)), int(match.group(2))  
  
 return None, None  
  
# Основной запуск  
async def main():  
 app = ApplicationBuilder().token(TOKEN).build()  
  
 app.add\_handler(CommandHandler("start", start))  
 app.add\_handler(MessageHandler(filters.TEXT & ~filters.COMMAND, handle\_message))  
  
 print("Бот запущен...")  
 await app.run\_polling()  
  
# Для запуска в PyCharm  
if \_\_name\_\_ == "\_\_main\_\_":  
 import nest\_asyncio  
 nest\_asyncio.apply()

Функции:

Запуск бота с кнопки «Старт»;

Обработка диапазона от пользователя;

Генерация случайного числа (random.randint);

Кнопки:

«Ещё одно число»;

«Новое число в том же диапазоне»;

«Задать новый диапазон»;

«Конец».

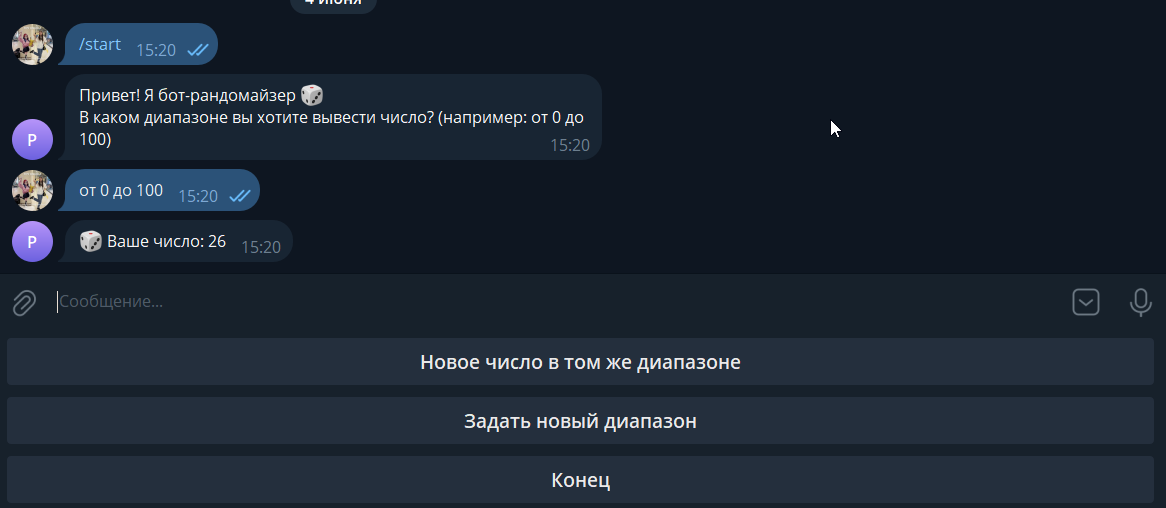


Рисунок 2: Работа бота

Технические особенности:

Модульная структура (main.py, handlers.py, states.py);

Использование FSMContext для хранения состояния пользователя;

Интеграция с Telegram API через токен;

Обработка ошибок (try-except).

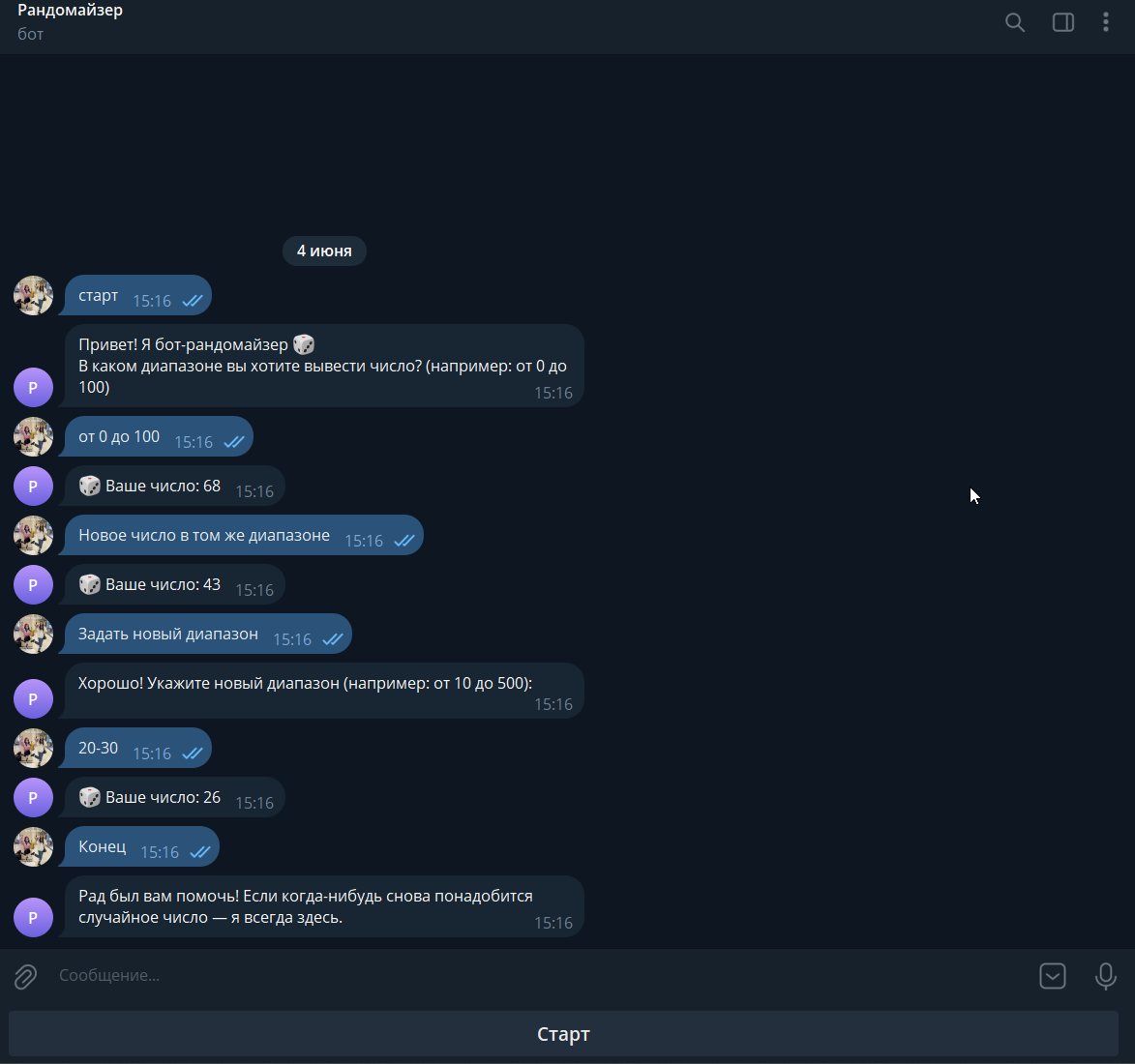


Рисунок 3: Завершенный цикл работы бота

**4. Описание достигнутых результатов по проектной практике**

По сайту:

Разработан полностью функционирующий веб-сайт;

Сайт размещён локально и может быть опубликован через GitHub Pages;

Выполнена вёрстка и стилизация;

Проведено тестирование на разных устройствах;

Получена обратная связь от сокурсников и преподавателей.

По Telegram-боту:

Разработан функциональный Telegram-бот;

Бот корректно обрабатывает пользовательские действия;

Используется конечный автомат состояний для логики диалога;

Код выложен на GitHub, снабжён README;

Бот протестирован в реальных условиях.

Документация:

Сформирован отчёт;

Созданы Markdown-документы по боту и сайту;

Подготовлена презентация проекта.

**Заключение.**

Проектная практика позволила закрепить на практике полученные теоретические знания. Были успешно освоены навыки:

* командной разработки с использованием Git;
* создания адаптивных сайтов;
* взаимодействия с Telegram API;
* проектирования конечных автоматов и ботов на Python;
* оформления документации в формате Markdown;
* презентации проекта заинтересованной аудитории.

Практика показала, что даже небольшой по объёму проект может принести ощутимую пользу и послужить основой для дальнейшего развития в рамках цифровой трансформации вуза.

Все материалы проекта доступны в публичном *GitHub-репозитории*

Все результаты практики задокументированы и размещены в Git-репозитории с соответствующими инструкциями и файлами: https://github.com/aliannanua/241-334

**СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ**

1. Документация Telegram API — https://core.telegram.org/bots/api

2. Официальная документация библиотеки telebot (pyTelegramBotAPI) — https://pytba.readthedocs.io

3. Официальный гайд по Markdown — https://www.markdownguide.org/basic-syntax/

4. GitHub Docs: Руководство пользователя Git — https://docs.github.com/en/get-started/using-git

5. Лекции и методические материалы по дисциплине «Проектная деятельность»



Гавриш Александрина Алексеевна / Подпись \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_