**Техническая документация Telegram-бота "AISID 2025”**

****

**Содержание**

1. **Введение**  
   1.1 Назначение  
   1.2 Основные функции
2. **Технические требования**  
   2.1 Среда выполнения  
   2.2 Зависимости
3. **Архитектура проекта**  
   3.1 Общая схема взаимодействия  
   3.2 Используемые технологии
4. **Установка и запуск**  
   4.1 Получение API-токена  
   4.2 Запуск бота
5. **Логика работы бота**  
   5.1 Описание команд  
   5.2 Описание работы обработчиков
6. **Принцип работы**  
   6.1 Главное меню  
   6.2 Разработчики  
   6.3 Техническая документация  
   6.4 Презентация проекта
7. **Описание кода**  
   7.1 Объяснение кода  
   7.2 Листинг программы

**1. ВВЕДЕНИЕ**

**1.1 Назначение**

AISID 2025 Telegram Bot – это бот для Telegram, предназначенный для предоставления информации о разработчиках, технической документации и презентации проекта.

**1.2 Основные функции**

* Отправка списка разработчиков
* Отправка ссылки и файла с технической документацией
* Отправка ссылки и файла с презентацией проекта

**2. ТЕХНИЧЕСКИЕ ТРЕБОВАНИЯ**

**2.1 Среда выполнения**

* Python 3.8+
* telebot для работы с Telegram API
* Файлы документации и презентации в формате PDF/MD

**2.2 Зависимости**

Перед запуском установите все необходимые библиотеки:

pip install pyTelegramBotAPI

**3. АРХИТЕКТУРА ПРОЕКТА**

**3.1 Общая схема взаимодействия**

1. Пользователь отправляет команду или нажимает кнопку в боте.
2. Бот принимает сообщение через Telegram API.
3. Обработчик telebot анализирует сообщение.
4. В зависимости от команды бот отправляет нужную информацию.

**3.2 Используемые технологии**

* **Python 3.8+** – основной язык программирования
* **telebot** – библиотека для работы с Telegram API

**4. УСТАНОВКА И ЗАПУСК**

**4.1 Получение API-токена**

1. Написать @BotFather в Telegram.
2. Создать нового бота и получить TOKEN.
3. Вставить TOKEN в код.

**4.2 Запуск бота**

python bot.py

**5. ЛОГИКА РАБОТЫ БОТА**

**5.1 Описание команд**

Бот использует библиотеку telebot. При запуске /start отправляется клавиатура с тремя кнопками:

* **"Разработчики"** – выводит список разработчиков
* **"Техническая документация"** – отправляет ссылку и MD-файл
* **"Презентация проекта"** – отправляет ссылку и PDF-файл

**5.2 Описание работы обработчиков**

| **Команда/Кнопка** | **Действие** |
| --- | --- |
| /start | Отправляет клавиатуру с кнопками |
| "Разработчики" | Выводит список разработчиков |
| "Техническая документация" | Показывает меню с возможностью открыть ссылку или скачать MD-файл |
| "Презентация проекта" | Показывает меню с возможностью открыть ссылку или скачать PDF-файл |

**6. ПРИНЦИП РАБОТЫ**

**6.1 Главное меню**

1. Пользователь вводит /start.
2. Бот отправляет клавиатуру с основными кнопками.

**6.2 Разработчики**

1. Пользователь нажимает кнопку **"Разработчики"**.
2. Бот отправляет список разработчиков.

**6.3 Техническая документация**

1. Пользователь нажимает кнопку **"Техническая документация"**.
2. Бот отправляет клавиатуру с кнопками **"Открыть документацию"**, **"Скачать документацию"**.
3. При выборе **"Скачать документацию"** бот отправляет MD-файл.

**6.4 Презентация проекта**

1. Пользователь нажимает кнопку **"Презентация проекта"**.
2. Бот отправляет клавиатуру с кнопками **"Открыть презентацию"**, **"Скачать презентацию"**.
3. При выборе **"Скачать презентацию"** бот отправляет PDF-файл.

**7. ОПИСАНИЕ КОДА**

**7.1 Объяснение кода**

 **Создание бота:**

TOKEN = "123"

bot = telebot.TeleBot(TOKEN, parse\_mode="HTML")

Здесь создаётся объект bot, который взаимодействует с Telegram API, используя заданный токен.

 **Функция main\_menu()**

def main\_menu():

keyboard = InlineKeyboardMarkup()

keyboard.row(InlineKeyboardButton("Разработчики", callback\_data="developers"))

keyboard.row(InlineKeyboardButton("Техническая документация", callback\_data="docs"))

keyboard.row(InlineKeyboardButton("Презентация проекта", callback\_data="presentation"))

return keyboard

* Создаёт **интерактивное меню** с кнопками:
  + "Разработчики" – показывает список разработчиков.
  + "Техническая документация" – даёт доступ к документации.
  + "Презентация проекта" – позволяет открыть презентацию.
* Каждая кнопка отправляет **callback-запрос** с уникальным callback\_data.

 **Обработка команды /start**

@bot.message\_handler(commands=['start'])

def start\_command(message):

bot.send\_message(message.chat.id, "Выберите действие:", reply\_markup=main\_menu())

* Отправляет пользователю **главное меню** после ввода команды /start.

 **Вывод списка разработчиков**

@bot.callback\_query\_handler(func=lambda call: call.data == "developers")

def send\_developers(call: CallbackQuery):

developers = "Проект разрабатывали:\n1. <b>Айтбаев Жеңіс Жұмабайұлы</b>\n2. <b>Сундет Сумая Байжанқызы</b>"

keyboard = InlineKeyboardMarkup().row(InlineKeyboardButton("Главное меню", callback\_data="back"))

bot.edit\_message\_text(developers, call.message.chat.id, call.message.message\_id, reply\_markup=keyboard)

* При нажатии на кнопку **"Разработчики"** бот обновляет сообщение и показывает список разработчиков.

 **Меню с технической документацией**

@bot.callback\_query\_handler(func=lambda call: call.data == "docs")

def send\_docs\_menu(call: CallbackQuery):

doc\_link = "https://github.com/nimbl9/tgbotaiasid/blob/main/tech\_doc.md"

keyboard = InlineKeyboardMarkup()

keyboard.row(InlineKeyboardButton("Открыть документацию", url=doc\_link))

keyboard.row(InlineKeyboardButton("Скачать документацию", callback\_data="download\_docs"))

keyboard.row(InlineKeyboardButton("Главное меню", callback\_data="back"))

bot.edit\_message\_text("Выберите действие:", call.message.chat.id, call.message.message\_id, reply\_markup=keyboard)

* Позволяет пользователю **открыть ссылку** на документацию или скачать файл.

 **Отправка файла с документацией**

@bot.callback\_query\_handler(func=lambda call: call.data == "download\_docs")

def send\_docs\_file(call: CallbackQuery):

bot.send\_document(call.message.chat.id, InputFile("Техническая документация проекта AISID2025.docx"), caption="Техническая документация")

bot.send\_message(call.message.chat.id, "Выберите действие:", reply\_markup=main\_menu())

* Загружает и отправляет файл с технической документацией.

 **Меню с презентацией**

@bot.callback\_query\_handler(func=lambda call: call.data == "presentation")

def send\_presentation\_menu(call: CallbackQuery):

presentation\_link = "https://docs.google.com/presentation/d/..."

keyboard = InlineKeyboardMarkup()

keyboard.row(InlineKeyboardButton("Открыть презентацию", url=presentation\_link))

keyboard.row(InlineKeyboardButton("Скачать презентацию", callback\_data="download\_presentation"))

keyboard.row(InlineKeyboardButton("Главное меню", callback\_data="back"))

bot.edit\_message\_text("Выберите действие:", call.message.chat.id, call.message.message\_id, reply\_markup=keyboard)

* Позволяет пользователю открыть ссылку на презентацию или скачать её.

 **Отправка файла с презентацией**

@bot.callback\_query\_handler(func=lambda call: call.data == "download\_presentation")

def send\_presentation\_file(call: CallbackQuery):

bot.send\_document(call.message.chat.id, InputFile("AISID2025.pptx"), caption="Презентация проекта")

bot.send\_message(call.message.chat.id, "Вернуться в главное меню:", reply\_markup=main\_menu())

* Отправляет файл с презентацией пользователю.

 **Возврат в главное меню**

@bot.callback\_query\_handler(func=lambda call: call.data == "back")

def go\_back(call: CallbackQuery):

bot.edit\_message\_text("Выберите действие:", call.message.chat.id, call.message.message\_id, reply\_markup=main\_menu())

* Позволяет вернуться в **главное меню**, обновляя предыдущее сообщение.

 **Запуск бота**

if \_\_name\_\_ == "\_\_main\_\_":

bot.polling(none\_stop=True)

* Запускает бота в режиме **polling**, который постоянно проверяет новые сообщения и обработчики callback-запросов.

**7.2 Листинг программы**

import telebot

from telebot.types import InlineKeyboardMarkup, InlineKeyboardButton, CallbackQuery, InputFile

TOKEN = "123"

bot = telebot.TeleBot(TOKEN, parse\_mode="HTML")

def main\_menu():

keyboard = InlineKeyboardMarkup()

keyboard.row(InlineKeyboardButton("Разработчики", callback\_data="developers"))

keyboard.row(InlineKeyboardButton("Техническая документация", callback\_data="docs"))

keyboard.row(InlineKeyboardButton("Презентация проекта", callback\_data="presentation"))

return keyboard

@bot.message\_handler(commands=['start'])

def start\_command(message):

bot.send\_message(message.chat.id, "Выберите действие:", reply\_markup=main\_menu())

@bot.callback\_query\_handler(func=lambda call: call.data == "developers")

def send\_developers(call: CallbackQuery):

developers = "Проект разрабатывали:\n1. <b>Айтбаев Жеңіс Жұмабайұлы</b>\n2. <b>Сундет Сумая Байжанқызы</b>"

keyboard = InlineKeyboardMarkup().row(InlineKeyboardButton("Главное меню", callback\_data="back"))

bot.edit\_message\_text(developers, call.message.chat.id, call.message.message\_id, reply\_markup=keyboard)

@bot.callback\_query\_handler(func=lambda call: call.data == "docs")

def send\_docs\_menu(call: CallbackQuery):

doc\_link = "https://docs.google.com/document/d/1hXXYnLH5xNwOOpMC7c3S5zzWkBZx83MW/edit?usp=drive\_link&ouid=113780296157828094590&rtpof=true&sd=true"

keyboard = InlineKeyboardMarkup()

keyboard.row(InlineKeyboardButton("Открыть документацию", url=doc\_link))

keyboard.row(InlineKeyboardButton("Скачать документацию", callback\_data="download\_docs"))

keyboard.row(InlineKeyboardButton("Главное меню", callback\_data="back"))

bot.edit\_message\_text("Выберите действие:", call.message.chat.id, call.message.message\_id, reply\_markup=keyboard)

@bot.callback\_query\_handler(func=lambda call: call.data == "download\_docs")

def send\_docs\_file(call: CallbackQuery):

bot.send\_document(call.message.chat.id, InputFile("Техническая документация проекта AISID2025.docx"), caption="Техническая документация")

bot.send\_message(call.message.chat.id, "Выберите действие:", reply\_markup=main\_menu())

@bot.callback\_query\_handler(func=lambda call: call.data == "presentation")

def send\_presentation\_menu(call: CallbackQuery):

presentation\_link = "https://docs.google.com/presentation/d/1jrCS\_VrKYrJCpALuJDKDp4shAQxc1AH4/edit?usp=drive\_link&ouid=113780296157828094590&rtpof=true&sd=true"

keyboard = InlineKeyboardMarkup()

keyboard.row(InlineKeyboardButton("Открыть презентацию", url=presentation\_link))

keyboard.row(InlineKeyboardButton("Скачать презентацию", callback\_data="download\_presentation"))

keyboard.row(InlineKeyboardButton("Главное меню", callback\_data="back"))

bot.edit\_message\_text("Выберите действие:", call.message.chat.id, call.message.message\_id, reply\_markup=keyboard)

@bot.callback\_query\_handler(func=lambda call: call.data == "download\_presentation")

def send\_presentation\_file(call: CallbackQuery):

bot.send\_document(call.message.chat.id, InputFile("AISID2025.pptx"), caption="Презентация проекта")

bot.send\_message(call.message.chat.id, "Вернуться в главное меню:", reply\_markup=main\_menu())

@bot.callback\_query\_handler(func=lambda call: call.data == "back")

def go\_back(call: CallbackQuery):

bot.edit\_message\_text("Выберите действие:", call.message.chat.id, call.message.message\_id, reply\_markup=main\_menu())

if \_\_name\_\_ == "\_\_main\_\_":

bot.polling(none\_stop=True)