Инструкция по выполнению Лабораторной работы №2

Тема: Реализация базового интерфейса интеллектуальной информационной системы (ИИС)

1. Цель работы

Научиться разрабатывать пользовательский интерфейс, позволяющий взаимодействовать с интеллектуальной системой:

* через веб-страницу,
* или через чат-бот (например, Telegram).

2. Варианты выполнения

Вариант 1. Веб-интерфейс

1. Создать простую HTML-страницу.
2. Добавить форму для ввода данных (например, регистрация пользователя, заказ товара, добавление книги).
3. Обработать данные с помощью JavaScript.
4. При наличии сервера – отправить запрос через API (например, fetch → Flask/FastAPI).
5. Вывести результат на экран.

Пример: форма для добавления книги и вывод результата.

<form id="bookForm">

<label>Название книги:</label>

<input type="text" name="title"><br>

<label>Автор:</label>

<input type="text" name="author"><br>

<button type="submit">Сохранить</button>

</form>

<div id="result"></div>

<script>

document.getElementById("bookForm").addEventListener("submit", function(e) {

e.preventDefault();

let title = e.target.title.value;

let author = e.target.author.value;

document.getElementById("result").innerText =

"Книга добавлена: " + title + " (" + author + ")";

});

</script>

Вариант 2. Чат-бот (например, Telegram)

1. Создать бота в Telegram через @BotFather.
   * Получить токен доступа.
2. Установить библиотеку Python aiogram или python-telegram-bot.
3. Написать код для обработки сообщений пользователя.

Пример простого Telegram-бота:

from aiogram import Bot, Dispatcher, executor, types

API\_TOKEN = "ТОКЕН\_ВАШЕГО\_БОТА"

bot = Bot(token=API\_TOKEN)

dp = Dispatcher(bot)

@dp.message\_handler(commands=["start"])

async def start(msg: types.Message):

await msg.reply("Привет! Я бот ИИС. Напиши название книги.")

@dp.message\_handler()

async def echo(msg: types.Message):

await msg.reply(f"Книга '{msg.text}' сохранена в базе!")

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

executor.start\_polling(dp, skip\_updates=True)

- Запустить бота на своём компьютере.

- Проверить работу: ввести текст → бот отвечает.

- При желании – подключить API (например, сохранение данных в файл или базу).

3. Структура отчёта

1. Титульный лист.
2. Цель и задание.
3. Выбранный вариант (веб-страница или чат-бот).
4. Код программы (HTML/JS или Python-бот).
5. Скриншоты работы интерфейса.
6. Выводы.

4. Контрольные вопросы для защиты

* Чем отличается веб-интерфейс от чат-бота?
* Какие преимущества у чат-бота по сравнению с веб-страницей?
* Что такое API и зачем оно нужно?
* Как пользователь взаимодействует с системой через интерфейс?