Инструкция по выполнению Лабораторной работы №3

Тема: Интеграция интеллектуального модуля в приложение

1. Цель работы

Научиться подключать к приложению интеллектуальный модуль (например, модель машинного обучения или внешний сервис ИИ), обмениваться с ним данными и использовать полученный результат.

2. Что нужно сделать

* Подготовить простое приложение (web-страница или бот).
* Настроить обмен данными с внешним ИИ-модулем через API.
* Получить ответ от ИИ и вывести его пользователю.
* Подготовить отчёт.

3. Варианты выполнения

Вариант 1. Web-приложение с ИИ API

1. Сделайте веб-страницу (можно взять за основу лабораторную №2).
2. В форме пользователь вводит данные (например, вопрос или текст).
3. В JavaScript добавьте обращение к API.  
   Для теста можно использовать публичные сервисы, например:
   * OpenAI (ChatGPT API),
   * HuggingFace (модели NLP),
   * любой учебный «заглушечный» API.

Пример отправки запроса к API (fetch):

fetch("http://localhost:5000/analyze", {

method: "POST",

headers: {"Content-Type": "application/json"},

body: JSON.stringify({text: "Привет"})

})

.then(res => res.json())

.then(data => {

document.getElementById("result").innerText = "Ответ: " + data.answer;

});

Вариант 2. Telegram-бот с ИИ

1. Используйте код бота из лабораторной №2.
2. Добавьте обработку запроса к ИИ-сервису.

Пример Python-бота, который отправляет сообщение на внешний API:

import requests

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! Напиши любой текст, я отправлю его в ИИ-модуль.")

@dp.message\_handler()

async def process(msg: types.Message):

response = requests.post("http://localhost:5000/analyze", json={"text": msg.text})

answer = response.json().get("answer", "Ошибка")

await msg.reply(f"ИИ ответил: {answer}")

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

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

Запустите серверный модуль (например, Flask/FastAPI), который принимает текст и возвращает обработанный ответ.

Пример простого «заглушечного» API на Flask:

from flask import Flask, request, jsonify

app = Flask(\_\_name\_\_)

@app.route("/analyze", methods=["POST"])

def analyze():

data = request.json

text = data.get("text", "")

return jsonify({"answer": text[::-1]}) # просто возвращает текст наоборот

app.run(port=5000)

4. Структура отчёта

1. Титульный лист.
2. Цель и задачи работы.
3. Схема взаимодействия приложения и ИИ-модуля.
4. Код программы (web-страницы или бота + API).
5. Скриншоты работы (ввод данных и ответ от ИИ).
6. Выводы.

5. Контрольные вопросы для защиты

* Что такое API и как оно работает?
* Чем отличается локальная модель от внешнего сервиса?
* Приведите пример использования ИИ в реальной системе.
* Какие форматы данных чаще всего используются для обмена (JSON, XML)?