Московский государственный технический университет им. Н.Э. Баумана

Факультет «Информатика и системы управления»
Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Парадигмы и конструкции языков программирования»

Отчет по лабораторной работе №4

Выполнил:
студент группы ИУ5-35Б
Талаев А.П.
Подпись и дата:

Проверил:
Подпись и дата:

Москва, 2024 г

Задание:
Телеграмм бот, отправляет актуальную информацию по криптовалюте через API CoinGecko.

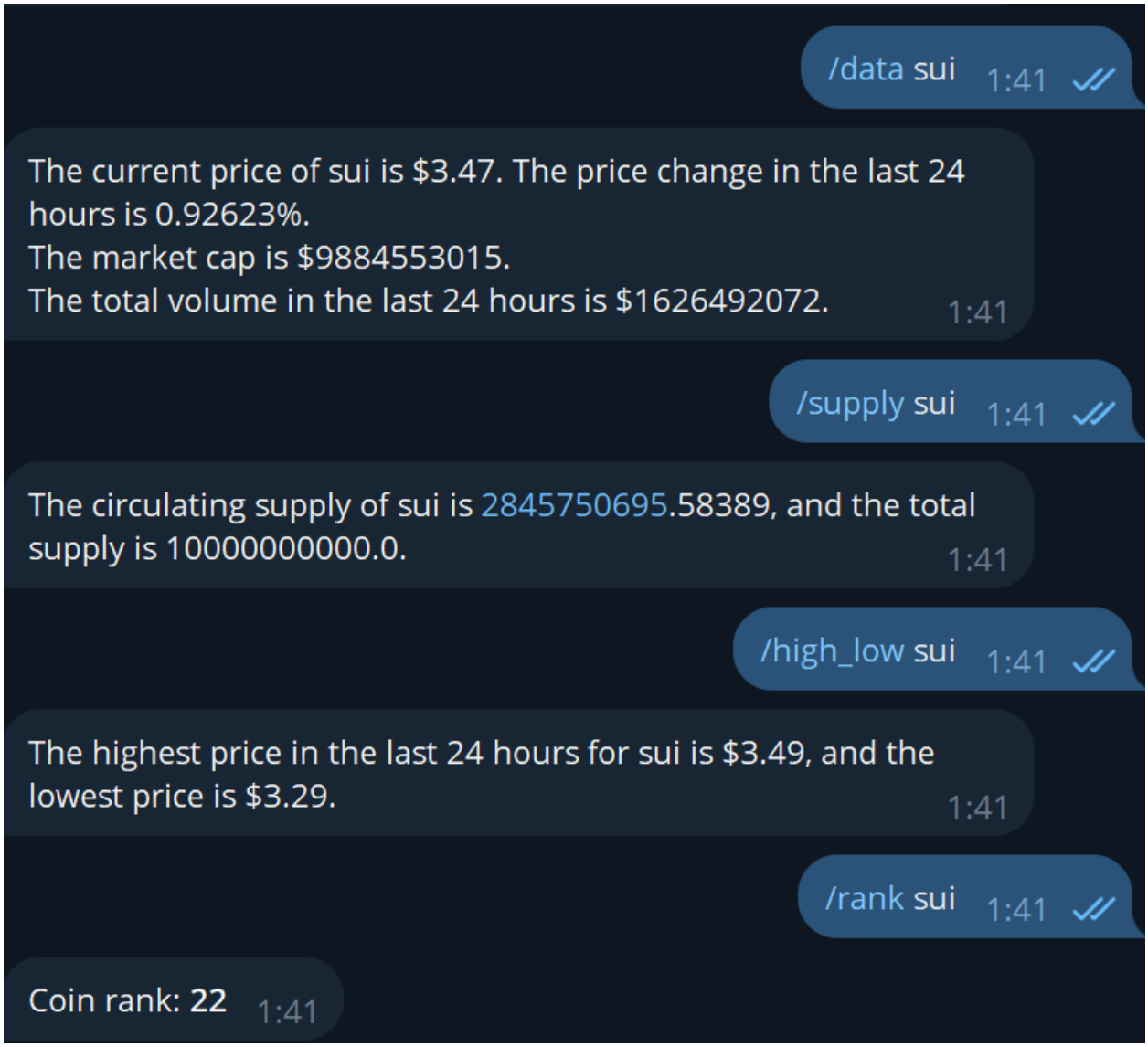
Выполнение:

```
from telegram import Update
from telegram.ext import ApplicationBuilder, CommandHandler, ContextTypes
import requests
import json

TG_KEY = '7600099992:AAHX1q7jHmEmPHDaInHZTXSHwAwx2WGT9Hw'
API_DEMO_KEY = 'CG-zvKXVQbn9hzZ3Reh2ZC2v55v'

def get_crypto_data(crypto):
    try:
        response =
requests.get(f'https://api.coingecko.com/api/v3/coins/markets?vs_currency=usd&id
s={crypto}&api_key={keys.API_DEMO_KEY}')
        response.raise_for_status()
        data = response.json()[0]
        return data
    except requests.exceptions.RequestException as e:
        print ("e^^rrr#ror:", e)

def get_top_cryptos():
    try:
        response =
requests.get(f'https://api.coingecko.com/api/v3/coins/markets?vs_currency=usd&o
rder=market_cap_desc&per_page=10&page=1&sparkline=false&api_key={keys.
API_DEMO_KEY}')
        response.raise_for_status()
        data = response.json()
        return data
    except requests.exceptions.RequestException as e:
        print ("e^^rrr#ror:", e)

async def start(update: Update, context: ContextTypes.DEFAULT_TYPE) ->
None:
    welcome_message = 'Hola amigo, используй /pleaseHELP чтобы узнать
доступные команды;'
    await update.message.reply_text(welcome_message)

async def data(update: Update, context: ContextTypes.DEFAULT_TYPE) ->
None:
    crypto = update.message.text.split()[1]
```

```
    data = get_crypto_data(crypto)
    if data:
        await update.message.reply_text(f'The current price of {crypto} is
${data["current_price"]}. The price change in the last 24 hours is
{data["price_change_percentage_24h"]}%.\nThe market cap is
${data["market_cap"]}.\nThe total volume in the last 24 hours is
${data["total_volume"]}.')
    else:
        await update.message.reply_text('хз что это')

async def high_low(update: Update, context: ContextTypes.DEFAULT_TYPE) ->
None:
    crypto = update.message.text.split()[1]
    data = get_crypto_data(crypto)
    if data:
        await update.message.reply_text(f'The highest price in the last 24 hours for
{crypto} is ${data["high_24h"]}, and the lowest price is ${data["low_24h"]}.')
    else:
        await update.message.reply_text('хз что это')

async def supply(update: Update, context: ContextTypes.DEFAULT_TYPE) ->
None:
    crypto = update.message.text.split()[1]
    data = get_crypto_data(crypto)
    if data:
        await update.message.reply_text(f'The circulating supply of {crypto} is
{data["circulating_supply"]}, and the total supply is {data["total_supply"]}.')
    else:
        await update.message.reply_text('хз что это')

async def top10(update: Update, context: ContextTypes.DEFAULT_TYPE) ->
None:
    data = get_top_cryptos()
    if data:
        message = "Топ 10, 10 лучших криптовалют из первых 10 входящих топ
10, первые 10 из топ 100:\n"
        for i, crypto in enumerate(data, start=1):
            message += f'{i}) *{crypto["name"]}* #{crypto["symbol"].upper()}:\n-
Price: ${crypto["current_price"]}\n- Market capitalization is
${crypto["market_cap"]}\n- Total volume in the last 24 hours is
${crypto["total_volume"]}\n\n'
        await update.message.reply_text(message, parse_mode='Markdown')
    else:
        await update.message.reply_text('хз что это')
```

```
def check_rank(query) -> None:
    try:
        url = "https://api.coingecko.com/api/v3/search"
        params = {'query': query}
        response = requests.get(url, params=params)
        response.raise_for_status()
        data = response.json()
        return data
    except requests.exceptions.RequestException as e:
        print ("e^^rrr#ror:", e)

async def rank(update: Update, context: ContextTypes.DEFAULT_TYPE) ->
None:
    data = check_rank(update.message.text.split()[1:])
    await update.message.reply_text(f'Coin rank:
*{data["coins"][0]['market_cap_rank']}*', parse_mode='Markdown')
    print(data["coins"][0]['market_cap_rank'])

async def help(update: Update, context: ContextTypes.DEFAULT_TYPE) ->
None:
    await update.message.reply_text("/data <crypto name> - Получает текущие
данные для конкретной криптовалюты;\n/supply <crypto name> - Получает
информацию о циркулирующем предложении и общем объеме определенной
криптовалюты;\n/high_low <crypto name> - Получает информацию о самой
высокой и самой низкой цене конкретной криптовалюты за последние 24
часа;\n/top10 - топ 10 криптовалют по объему торгов, капитализации и
т.д.;\n/rank <crypto name>- рейтинг валюты;")

app = ApplicationBuilder().token(keys.TG_KEY).build()

app.add_handler(CommandHandler("start", start))
app.add_handler(CommandHandler("data", data))
app.add_handler(CommandHandler("high_low", high_low))
app.add_handler(CommandHandler("supply", supply))
app.add_handler(CommandHandler("top10", top10))
app.add_handler(CommandHandler("rank", rank))
app.add_handler(CommandHandler("pleaseHELP", help))

app.run_polling()
```

Результаты:


/data sui 1:41
The current price of sui is $3.47. The price change in the last 24 hours is 0.92623%.
The market cap is $9884553015.
The total volume in the last 24 hours is $1626492072. 1:41
/supply sui 1:41
The circulating supply of sui is 2845750695.58389, and the total supply is 10000000000.0. 1:41
/high_low sui 1:41
The highest price in the last 24 hours for sui is $3.49, and the lowest price is $3.29. 1:41
/rank sui 1:41
Coin rank: 22 1:41