Homework 9

Task 1

```python
from telegram import Update
from telegram.ext import Application, CommandHandler, MessageHandler, filters, ContextTypes

API_TOKEN = '7765799285:AAEspqbsgtbOOgb3anyDO7hqDNKT873RZHI'

async def start(update: Update, context: ContextTypes.DEFAULT_TYPE) -> None:
    await update.message.reply_text("Hello! I am your AI assistant, how can I help you!")

async def handle_message(update: Update, context: ContextTypes.DEFAULT_TYPE) -> None:
    user_response = update.message.text
    await update.message.reply_text("I have received your message:  \"{}\"".format(user_response))

def main():
    # Creation of Application object
    application = Application.builder().token(API_TOKEN).build()

    # Command and message handlers
    application.add_handler(CommandHandler("start", start))
    application.add_handler(MessageHandler(filters.TEXT & ~filters.COMMAND, handle_message))

    # Start the bot
    application.run_polling()

if __name__ == '__main__':
    main()
```

Task 2

GitHub - https://github.com/Shagun-Shah/DSSS_Homework_9
Telegram Bot address - https://t.me/DSSSA9Bot

Task 2

```python
from telegram import Update
from telegram.ext import Application, CommandHandler, MessageHandler, filters, ContextTypes
from transformers import pipeline
import os

os.environ["HF_HUB_DISABLE_SYMLINKS_WARNING"] = "1"
API_TOKEN = "7765799285:AAEspqbsgtbOOgb3anyDO7hqDNKT873RZHI"

pipe = pipeline(
    "text-generation",
    model="TinyLlama/TinyLlama-1.1B-Chat-v1.0",
    torch_dtype="bfloat16",
    device_map="auto"
)
async def start(update: Update, context: ContextTypes.DEFAULT_TYPE) -> None:
    await update.message.reply_text("Hello! I am your AI assistant, how can I help you!")

async def stop(update: Update, context: ContextTypes.DEFAULT_TYPE) -> None:
    await update.message.reply_text("Stopping the bot. Goodbye!")
    await context.application.stop()

async def handle_message(update: Update, context: ContextTypes.DEFAULT_TYPE) -> None:
    if not update.message or not update.message.text:
        return
    user_response = update.message.text
    await update.message.reply_text(f"I have received your message: \"{user_response}\"")
    if not pipe:
        await update.message.reply_text("Sorry, the AI assistant is currently unavailable.")
        return
    try:
        messages = [{"role": "user", "content": user_response}]
        prompt = pipe.tokenizer.apply_chat_template(messages, tokenize=False, add_generation_prompt=True)
        outputs = pipe(prompt, max_new_tokens=180, do_sample=True, temperature=0.5, top_k=50, top_p=0.95)
        assistant_response = outputs[0]["generated_text"].split("<|assistant|>")[-1].strip()
        await update.message.reply_text(assistant_response)
    except Exception as e:
        print(f"Error during message processing: {e}")
        await update.message.reply_text("Something went wrong. Please try again later.")
def main():
    application = Application.builder().token(API_TOKEN).build()
    application.add_handler(CommandHandler("start", start))
    application.add_handler(CommandHandler("stop", stop))
    application.add_handler(MessageHandler(filters.TEXT & ~filters.COMMAND, handle_message))

    import asyncio
    asyncio.run(application.run_polling())

if __name__ == "__main__":
    main()
```