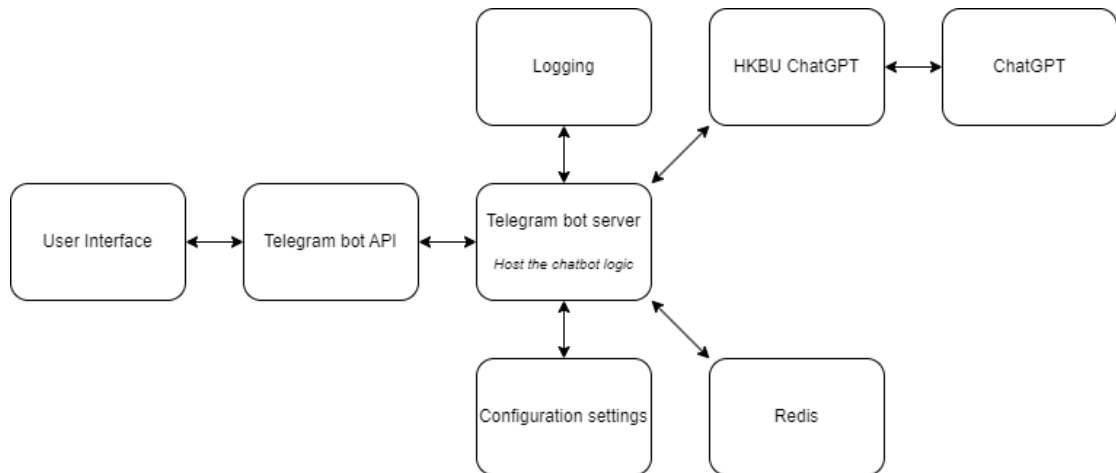


- 1) Please describe the architecture of the current chatbot system. Identify the components and check where they are running now.



<u>System components</u>	<u>Run on</u>	<u>Description</u>
User interface	User device	As an interface
Telegram bot API	Telegram server	As an intermediary between bot server and the user interface
Telegram bot Server	Telegram server	Host the python code of the bot application
Logging	Telegram server	Logging events and messages
Configuration settings	Local	Read by the bot server during initialization
Redis	Redis server	Open source in memory storage
HKBU ChatGPT	Local	As a bridge between telegram bot and official ChatGPT
ChatGPT	ChatGPT server	AI model developed by OpenAI

- 2) Explain how do your chatbot handles the special command. You need to trace the code and explain that.

There are currently 2 special commands in my chatbot, namely /add and /help, a third special command /hello will be added later too.

```
# on different commands - answer in Telegram
dispatcher.add_handler(CommandHandler("add", add))
dispatcher.add_handler(CommandHandler("help", help_command))
```

The '/add' command will trigger the add function to count the number of same inputs.

Steps:

- The 'CommandHandler' will call the add function and check the user input.
- If no keyword is added after typing '/add', the system will execute the except block and return a message 'Use: /add <keyword>'
- If the system detects an input, the system will log the first argument passed in the input. For instance, the first argument of '/add my name is' will be 'my'.
- The system then passes the argument to Redis and increment the argument count associated with the argument stored in Redis
- Finally, the system replies with a message indicating the argument and the number of times the Redis has stored.

The '/help' command will trigger the 'help_command' and output 'Helping you helping you'.

Steps:

- The 'CommandHandler' will call the 'help_command' function.
- The system will reply with the message 'Helping you helping you'.

Detail code:

```
def help_command(update: Update, context: CallbackContext) -> None:
    """Send a message when the command /help is issued."""
    update.message.reply_text('Helping you helping you.')

1 usage Christopher
def add(update: Update, context: CallbackContext) -> None:
    """Send a message when the command /add is issued."""
    try:
        global redis1
        logging.info(context.args[0])
        msg = context.args[0] # /add keyword <-- this should store the keyword
        redis1.incr(msg)
        update.message.reply_text('You have said ' + msg + ' for ' + redis1.get(msg).decode('UTF-8') + ' times.')
    except (IndexError, ValueError):
        update.message.reply_text('Usage: /add <keyword>')
```

- Update your code so that when user type /hello Kevin, it will reply 'Good day, Kevin!' Write down the changes you have made.

Output:



Changes made:

```
# on different commands - answer in Telegram
dispatcher.add_handler(CommandHandler('add', add))
dispatcher.add_handler(CommandHandler('help', help_command))
dispatcher.add_handler(CommandHandler('hello', hello))
```

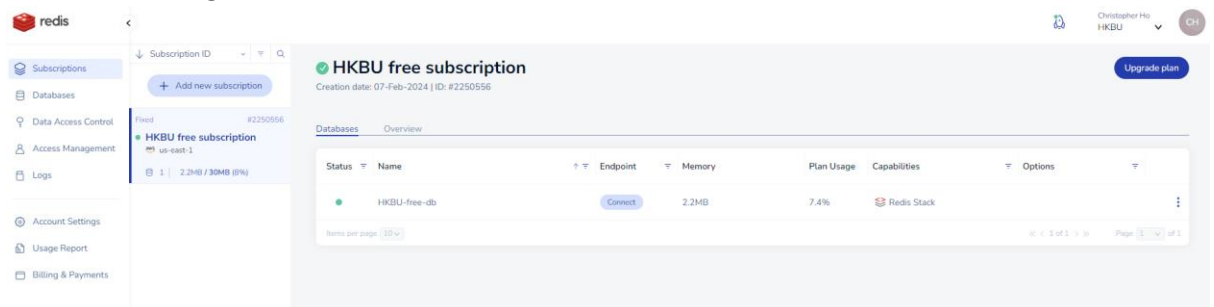
- Added a dispatcher line

```
1 usage Christopher
def hello(update: Update, context: CallbackContext) -> None:
    """Send a message when the command /hello is issued."""
    try:
        global redis1
        logging.info(context.args[0])
        name = context.args[0]
        redis1.incr(name)
        update.message.reply_text('Good day, ' + name + '!')
    except (IndexError, ValueError):
        update.message.reply_text('Usage: /help <keyword>')
```

b) Defined the hello function

- 4) Make a few screen caps to prove that you have applied your own Redis account, used it in your chatbot, and push the code on GitHub (at least 2 commits – lab3/lab4)

Redis Account registration:



Used in the chatbot:



```
1 usage Christopher
def hello(update: Update, context: CallbackContext) -> None:
    """Send a message when the command /hello is issued."""
    try:
        global redis1
        logging.info(context.args[0])
        name = context.args[0]
        redis1.incr(name)
        update.message.reply_text('Good day, ' + name + '!')
    except (IndexError, ValueError):
        update.message.reply_text('Usage: /help <keyword>')
```

9 commits in total:


 **comp7940_lab** Public

 Pin  Unwatch 1


 main  1 Branch  0 Tags




 Go to file 

 Add file

 Code

 **hwk-c** Add hello function

968cb36 · 19 minutes ago  9 Commits

 lab1_2_writeup	Revert "Lab3_4 Writeup"	1 hour ago
 lab3_4_writeup	Add hello function	19 minutes ago
 telegram_bot	create telegram bot	2 weeks ago