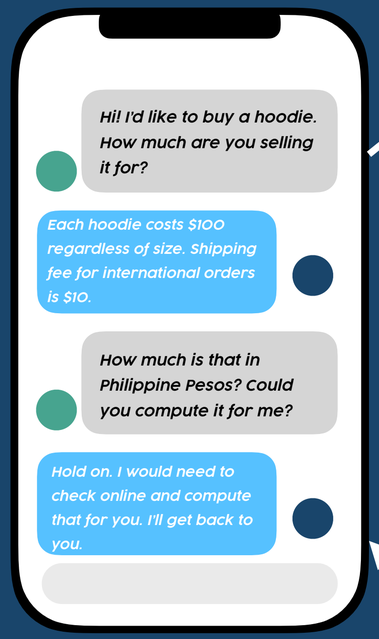
MONEYX BOT

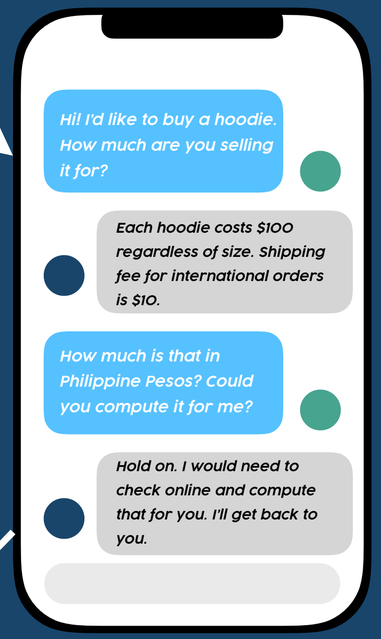
CABOCHENG
FERNANDO
LORENZO

OUR BOT LOGO

CONTEXT



*With the **current restrictions** in place, **many** have **resorted** to **online shopping** and **selling** – both **locally** and, more importantly, **internationally**. Most **business transactions**, **corporate affairs**, and **meetings** happen **online** as well. With **each country** having their **own currency**, it would be **hard** for **people** from **different nations** to **get a hold** on the **value** of a certain amount **without performing conversions** first. Having a **bot perform** this through the **messaging platform** itself would definitely be a **more efficient** and **transparent** way to **conduct business**.*



SELLER FROM USA

BUYER FROM PH

SIGNIFICANCE

ACCESSIBILITY

The bot was programmed to be available for usage at any time of the day.

ACCURACY

Data is kept up-to-date ensuring quality data.

CONVENIENCE

The bot is able to convert prices quickly and provide graphs to aid users in interpreting data.

MULTIFUNCTIONALITY

The bot is able to perform multiple functions to avoid the hassle of individually searching and computing for each function online.

IDEA

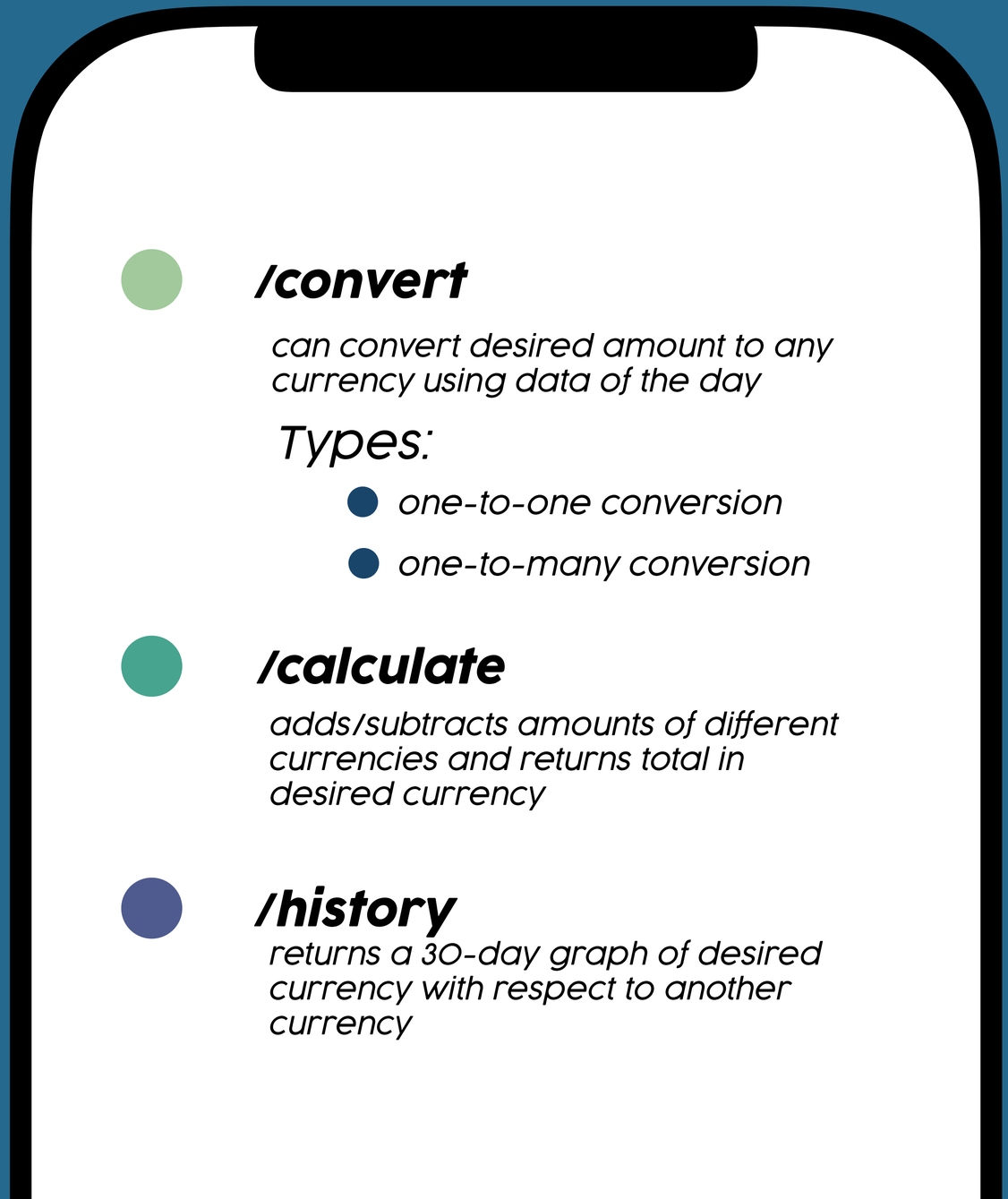
An **enhanced currency converter** in the **comfort** of a **messaging platform (Telegram)**

GOALS

- ***Efficient transactions*** during ***foreign negotiations*** in the ***comfort*** of ***one's own chat box***
- Provide a ***one-stop bot*** for ***providing info*** on ***currency conversions*** and ***graphs of conversion rates***
- ***Promote transparency*** in ***currency conversion calculations*** during ***international trade*** and ***negotiations***

NOTABLE BOT FEATURES

* Bot is hosted on Heroku thus works 24/7
* Bot can be called into telegram chats
* Currency rates are updated daily



/convert

can convert desired amount to any currency using data of the day

Types:

- one-to-one conversion
- one-to-many conversion

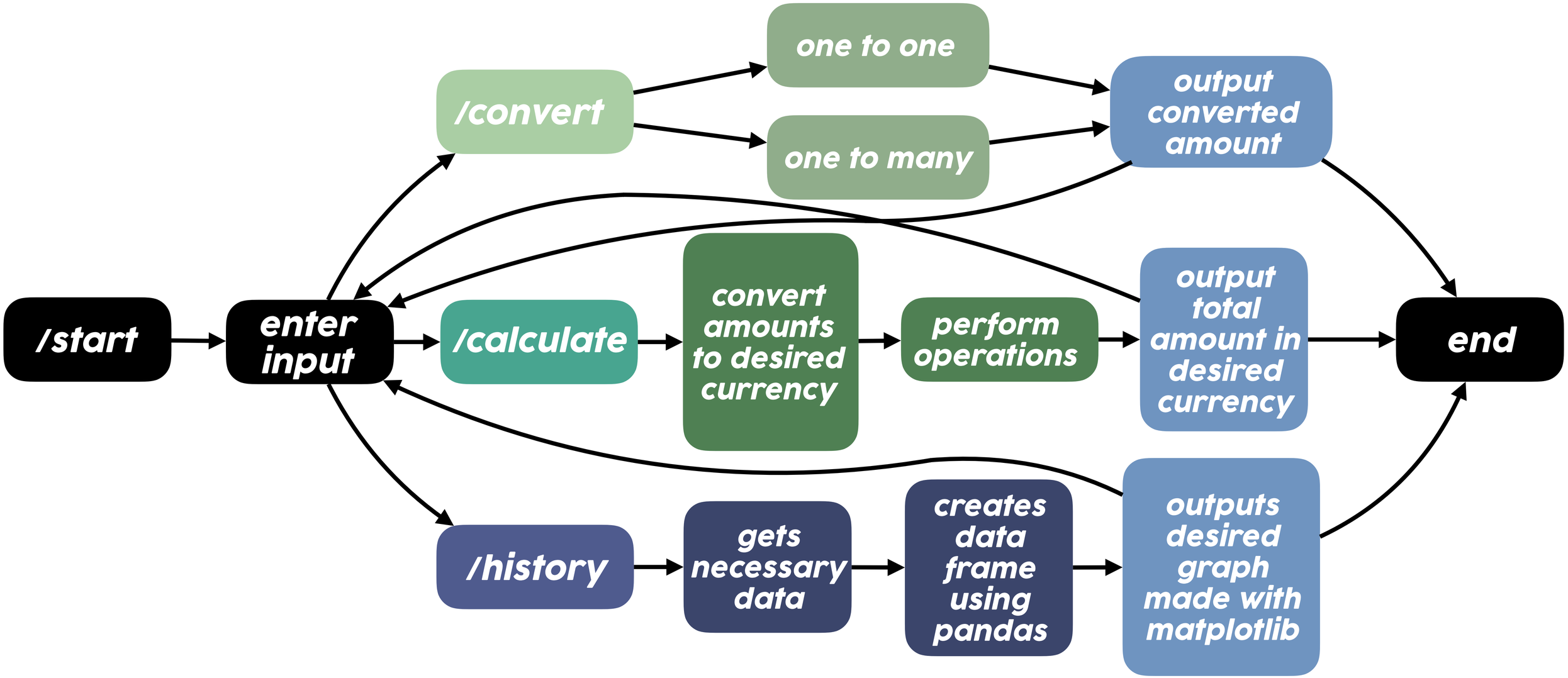
/calculate

adds/subtracts amounts of different currencies and returns total in desired currency

/history

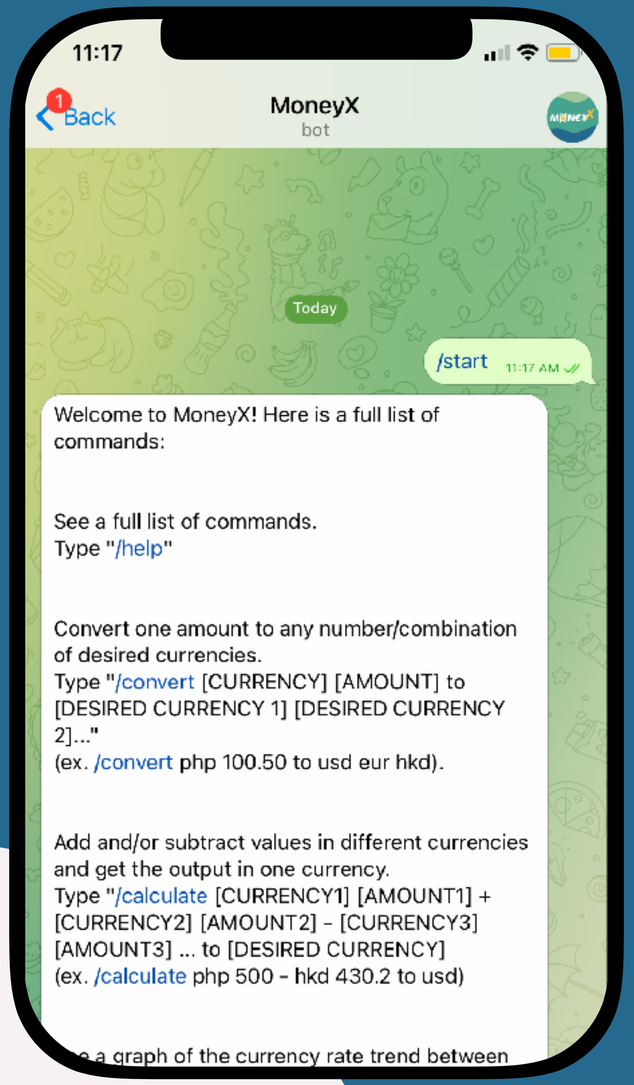
returns a 30-day graph of desired currency with respect to another currency

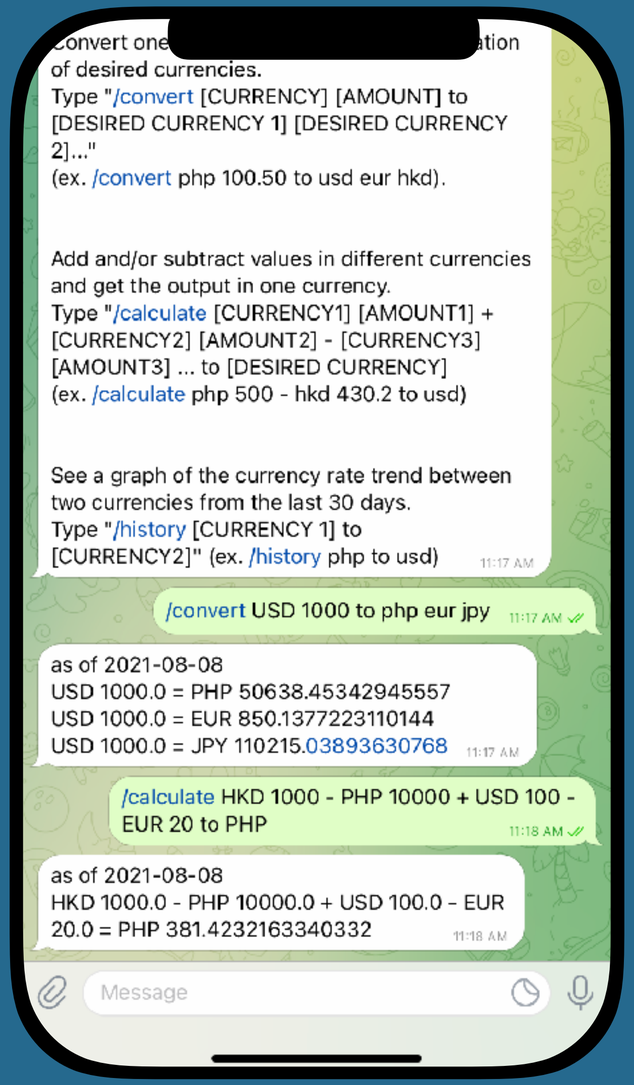
PROGRAM FLOWCHART

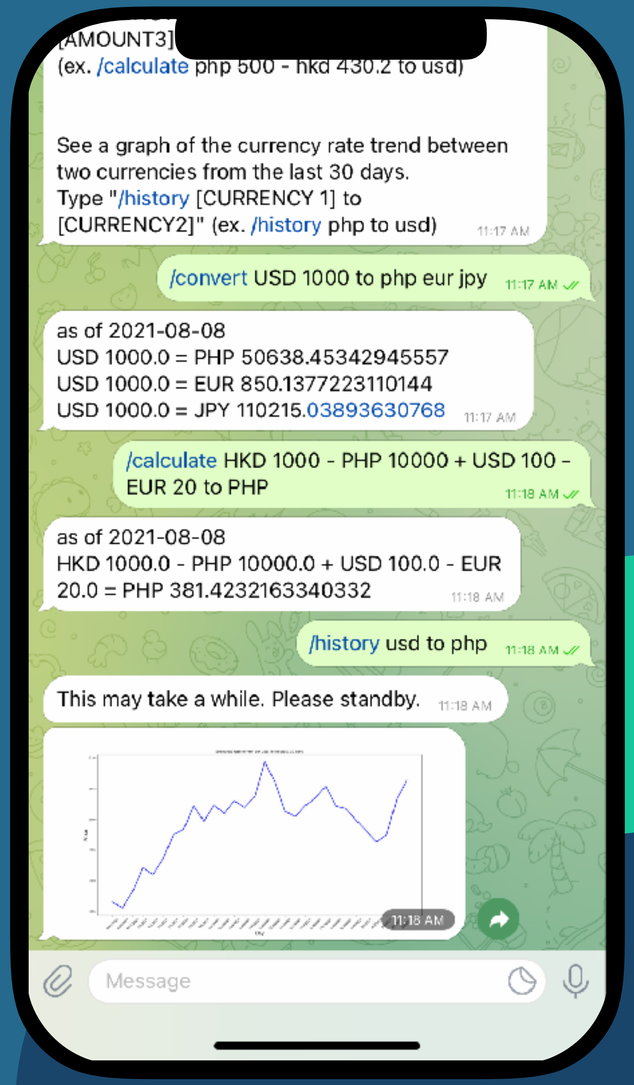


* did not include /help since it just prints out an options menu

SAMPLE TELEGRAM BOT SESSION







THE CODE

PROCFILE

```
worker: python itmgt2021bot.py
```

Tells heroku what kind of project it's going to be
Tells heroku what commands to run in startup

REQUIREMENTS.TXT

```
python-telegram-bot
requests
pandas
matplotlib

bob-telegram-tools
lxml
```

Tells heroku what modules it needs to install with pip

IMPORTING LIBRARIES AND INITIALIZATION

```
1  import logging
2  import requests
3  from telegram import Update, ForceReply
4  from telegram.ext import Updater, CommandHandler, MessageHandler, Filters, CallbackContext
5
6  import pandas as pd
7  import matplotlib.pyplot as plt
8  import subprocess
9  from bob_telegram_tools.bot import TelegramBot
10
11 logging.basicConfig(format='%(asctime)s - %(name)s - %(levelname)s - %(message)s', level=logging.INFO)
12 logger = logging.getLogger(__name__)
13
```

/start function

```
14  def start(update: Update, context: CallbackContext) -> None:
15      update.message.reply_text('''
16      Welcome to MoneyX! Here is a full list of commands:\n
17
18  See a full list of commands.\t
19  Type "/help"\n
20
21  Convert one amount to any number/combination of desired currencies.\t
22  Type "/convert [CURRENCY] [AMOUNT] to [DESIRED CURRENCY 1] [DESIRED CURRENCY 2]..."\t
23  (ex. /convert php 100.50 to usd eur hkd).\n
24
25  Add and/or subtract values in different currencies and get the output in one currency.\t
26  Type "/calculate [CURRENCY1] [AMOUNT1] + [CURRENCY2] [AMOUNT2] - [CURRENCY3] [AMOUNT3] ... to [DESIRED CURRENCY]\t
27  (ex. /calculate php 500 - hkd 430.2 to usd)\n
28
29  See a graph of the currency rate trend between two currencies from the last 30 days.\t
30  Type "/history [CURRENCY 1] to [CURRENCY2]" (ex. /history php to usd)
31      ''')
32
```

/convert function

```
33  def convert(update: Update, context: CallbackContext) -> None:
34      r = requests.get('http://api.exchangeratesapi.io/v1/latest?access_key=9a223be759130d6d46862c6665f63926')
35
36      try:
37          currency1_name = context.args[0].upper()
38          currency1_rate = r.json()['rates'][currency1_name]
39          currency1_value = float(context.args[1])
40
41          currency2_name = context.args[3:]
42          result_formatted = 'as of ' + r.json()['date']
43          for i in range(len(currency2_name)):
44              currency2_name_indiv = currency2_name[i].upper()
45              currency2_rate = r.json()['rates'][currency2_name_indiv]
46              currency2_value = (currency1_value/currency1_rate) * currency2_rate
47
48              result_formatted += '\n' + currency1_name + ' ' + str(currency1_value) + ' = ' + currency2_name_indiv + ' ' + str(currency2_value)
49
50          update.message.reply_text(result_formatted)
51      except Exception as e:
52          print(e)
53          update.message.reply_text('Error. Please try again. Enter /help to see a list of commands.')
54
```

/calculate function part 1

```
55  def calculate(update: Update, context: CallbackContext) -> None:
56      r = requests.get('http://api.exchangeratesapi.io/v1/latest?access_key=9a223be759130d6d46862c6665f63926')
57
58      try:
59          # first conversion
60          currency1_name = context.args[0].upper()
61          currency1_rate = r.json()['rates'][currency1_name]
62          currency1_value = float(context.args[1])
63
64          currency2_name = context.args[-1].upper()
65          currency2_rate = r.json()['rates'][currency2_name]
66          currency2_value = (currency1_value/currency1_rate) * currency2_rate
67          result_value = currency2_value
68          result_formatted = 'as of ' + r.json()['date']
69          result_formatted += '\n' + currency1_name + ' ' + str(currency1_value)
70
71          # second conversion
72          try:
73              currency_other = context.args[2:-2]
74              currency_other_num = int(len(currency_other) / 3)
75              for i in range(currency_other_num):
76                  currency3_name = context.args[1 + (i * 3) + 2].upper()
77                  currency3_rate = r.json()['rates'][currency3_name]
78                  currency3_operator = context.args[1 + (i * 3) + 1]
79                  currency3_absvalue = float(context.args[1 + (i * 3) + 3])
80                  currency3_value = float(currency3_operator + str(currency3_absvalue))
81
```

/calculate function part 2

```
82                  currency4_name = context.args[-1].upper()
83                  currency4_rate = r.json()['rates'][currency4_name]
84                  currency4_value = (currency3_value/currency3_rate) * currency4_rate
85
86                  result_value += currency4_value
87                  result_formatted += ' ' + currency3_operator + ' ' + currency3_name + ' ' + str(currency3_absvalue)
88
89              result_formatted += ' = ' + currency4_name + ' ' + str(result_value)
90
91          except:
92              result_formatted += ' = ' + currency2_name + ' ' + str(currency2_value)
93
94          update.message.reply_text(result_formatted)
95      except:
96          update.message.reply_text('Error. Please try again. Enter /help to see a list of commands.')
97
```

/history function

```
98  def history(update: Update, context: CallbackContext) -> None:
99      chat_id = update.message.chat_id
100     bot = TelegramBot("1930727729:AAEEsKKfPyVlW-Ea2UNGBTzkugMXLeBIB3E", chat_id)
101
102     try:
103         #ex. USD to PHP
104         currency1_name = context.args[0].upper()
105         currency2_name = context.args[2].upper()
106         base_url="https://www.exchange-rates.org/history/"
107         extension=f"{currency2_name}/{currency1_name}/T"
108         url=f"{base_url}{extension}"
109
110         dfs = pd.read_html(url)
111
112         date=dfs[0][0][0:30]
113
114         price=dfs[0][2][0:30].str.rstrip(currency2_name).astype('float')
115         fig=plt.figure(figsize=(20, 10), dpi=80)
116         plt.plot(date,price,"b",linewidth=2.50)
117         plt.gca().invert_xaxis()
118         plt.xlabel("Day",fontsize=16)
119         plt.ylabel("Price",fontsize=16)
120         plt.xticks(rotation=45)
121         plt.title(f"Exchange rate of {currency2_name} per {currency1_name} in the past 30 days")
122
123         update.message.reply_text('This may take a while. Please standby.')
124         bot.send_plot(plt)
125
126     except:
127         update.message.reply_text('Error. Please try again. Enter /help to see a list of commands.')
128
```

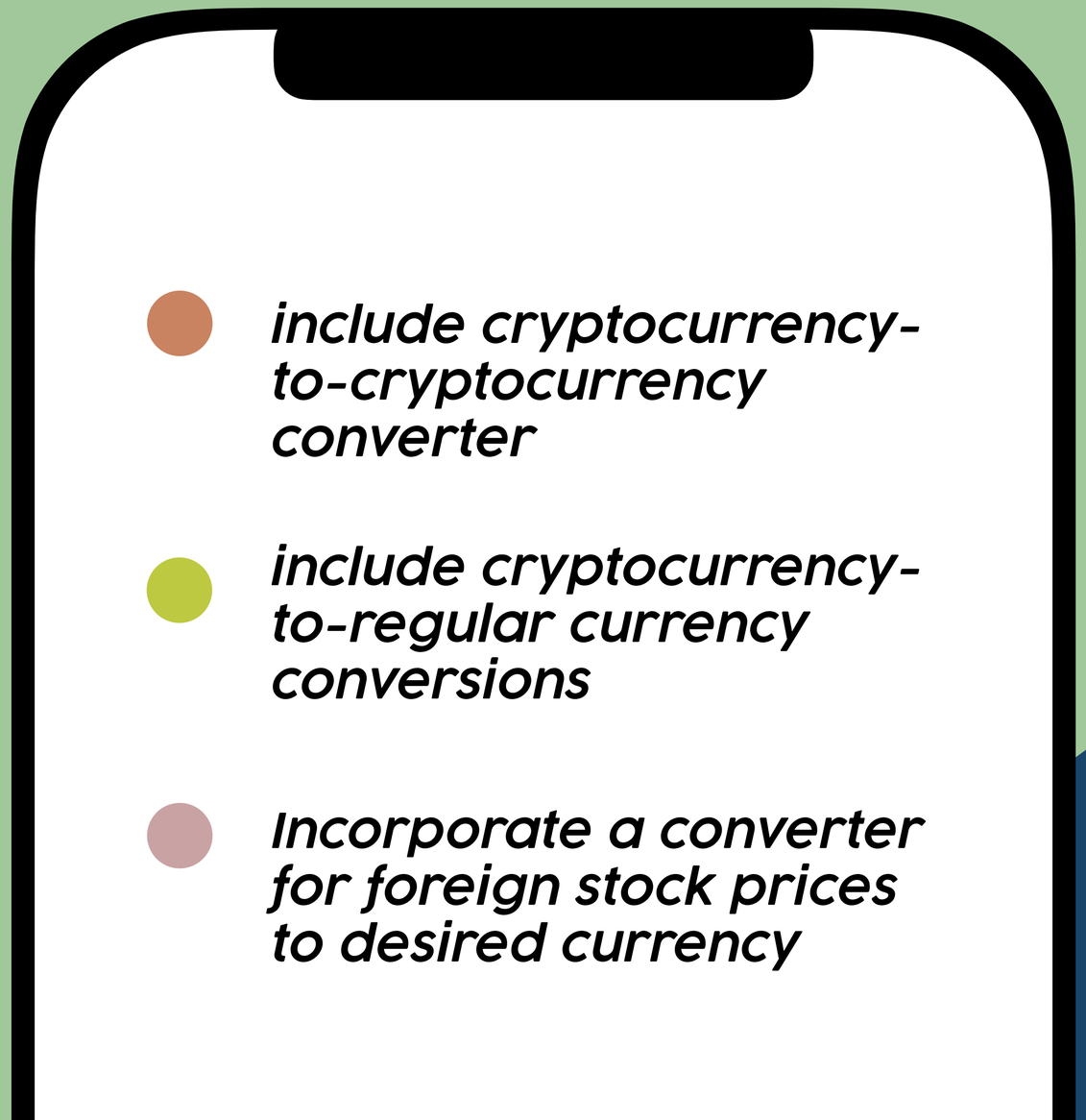
/help function

```
129  def help(update: Update, context: CallbackContext) -> None:
130      update.message.reply_text('''
131      Here is a full list of commands:\n
132
133  Convert one amount to any number/combination of desired currencies.\t
134  Type "/convert [CURRENCY] [AMOUNT] to [DESIRED CURRENCY 1] [DESIRED CURRENCY 2]..."\t
135  (ex. /convert php 100.50 to usd eur hkd).\n
136
137  Add and/or subtract values in different currencies and get the output in one currency.\t
138  Type "/calculate [CURRENCY1] [AMOUNT1] + [CURRENCY2] [AMOUNT2] - [CURRENCY3] [AMOUNT3] ... to [DESIRED CURRENCY]\t
139  (ex. /calculate php 500 - hkd 430.2 to usd)\n
140
141  See a graph of the currency rate trend between two currencies from the last 30 days.\t
142  Type "/history [CURRENCY 1] to [CURRENCY2]" (ex. /history php to usd)
143      ''')
144
```

echo and main function

```
145  def echo(update: Update, context: CallbackContext) -> None:
146      update.message.reply_text(update.message.text)
147
148  def main() -> None:
149      updater = Updater("1930727729:AAEEsKKfPyVlW-Ea2UNGBTzkugMXLeBIB3E")
150      dispatcher = updater.dispatcher
151
152      dispatcher.add_handler(CommandHandler("start", start))
153      dispatcher.add_handler(CommandHandler("help", help))
154      dispatcher.add_handler(CommandHandler("convert", convert))
155      dispatcher.add_handler(CommandHandler("calculate", calculate))
156      dispatcher.add_handler(CommandHandler("history", history))
157
158      updater.start_polling()
159
160      updater.idle()
161
162
163
164
165  if __name__ == '__main__':
166      main()
```

POSSIBLE EXTENSIONS/UPDATES



- *include cryptocurrency-to-cryptocurrency converter*
- *include cryptocurrency-to-regular currency conversions*
- *Incorporate a converter for foreign stock prices to desired currency*