InlineKeyboardButton("تحديث الحالة", callback\_data=f"update\_status:{loc\_id}")],

[InlineKeyboardButton("حذف", callback\_data=f"delete:{loc\_id}")]

]

if photo\_file\_id:

try:

await update.message.reply\_photo(photo=photo\_file\_id, caption=text, reply\_markup=InlineKeyboardMarkup(buttons))

except:

await update.message.reply\_text(text, reply\_markup=InlineKeyboardMarkup(buttons))

else:

await update.message.reply\_text(text, reply\_markup=InlineKeyboardMarkup(buttons))

async def callback\_query\_handler(update: Update, context: ContextTypes.DEFAULT\_TYPE):

q = update.callback\_query

await q.answer()

data = q.data

if data.startswith("details:"):

loc\_id = int(data.split(":",1)[1])

loc = get\_location(loc\_id)

if not loc:

await q.edit\_message\_text("الموقع غير موجود.")

return

\_, name, lat, lon, tanks\_count, status, photo\_file\_id, created\_at = loc

text = (f"تفاصيل الموقع\n"

f"ID: {loc\_id}\n"

f"الاسم: {name}\n"

f"الموقع: {lat:.6f}, {lon:.6f}\n"

f"الخزانات: {tanks\_count}\n"

f"الحالة: {status}\n"

f"أضيف في: {created\_at}")

buttons = [[InlineKeyboardButton("فتح في خرائط", url=f"https://www.google.com/maps/search/?api=1&query={lat},{lon}")]]

if photo\_file\_id:

try:

await q.message.reply\_photo(photo=photo\_file\_id, caption=text, reply\_markup=InlineKeyboardMarkup(buttons))

except:

await q.message.reply\_text(text, reply\_markup=InlineKeyboardMarkup(buttons))

else:

await q.message.reply\_text(text, reply\_markup=InlineKeyboardMarkup(buttons))

elif data.startswith("update\_status:"):

loc\_id = int(data.split(":",1)[1])

buttons = [

[InlineKeyboardButton("ممتلئ", callback\_data=f"setstatus:{loc\_id}:full")],

[InlineKeyboardButton("فارغ", callback\_data=f"setstatus:{loc\_id}:empty")],

[InlineKeyboardButton("يحتاج تعبئة", callback\_data=f"setstatus:{loc\_id}:need\_fill")],

[InlineKeyboardButton("إلغاء", callback\_data="noop")]

]

await q.edit\_message\_text("اختر الحالة الجديدة:", reply\_markup=InlineKeyboardMarkup(buttons))

elif data.startswith("setstatus:"):

parts = data.split(":")

loc\_id = int(parts[1])

new\_status = parts[2]

mapping = {'full': 'ممتلئ', 'empty': 'فارغ', 'need\_fill': 'يحتاج تعبئة'}

human = mapping.get(new\_status, new\_status)

update\_location\_status(loc\_id, human)

await q.edit\_message\_text(f"تم تحديث حالة الموقع (ID={loc\_id}) إلى: {human}")

elif data.startswith("delete:"):

loc\_id = int(data.split(":",1)[1])

delete\_location(loc\_id)

await q.edit\_message\_text(f"تم حذف الموقع (ID={loc\_id}).")

elif data == "noop":

await q.edit\_message\_text("تم الإلغاء.")

async def cancel(update: Update, context: ContextTypes.DEFAULT\_TYPE):

user\_id = update.effective\_user.id

if not is\_owner(user\_id):

return

await update.message.reply\_text("تم الإلغاء.")

return ConversationHandler.END

def main():

TOKEN = os.getenv("TG\_BOT\_TOKEN")

if not TOKEN:

print("❌ متغير البيئة TG\_BOT\_TOKEN غير موجود!")

return

init\_db()

app = ApplicationBuilder().token(TOKEN).build()

conv\_handler = ConversationHandler(

entry\_points=[CommandHandler('add', add\_start)],

states={

ADD\_NAME: [MessageHandler(filters.TEXT & ~filters.COMMAND, add\_name)],

ADD\_LOCATION: [MessageHandler(filters.LOCATION, add\_location\_received)],

ADD\_TANKS: [MessageHandler(filters.TEXT & ~filters.COMMAND, add\_tanks)],

},

fallbacks=[CommandHandler('cancel', cancel)]

) app.add\_handler(CommandHandler('start', start))

app.add\_handler(CommandHandler('help', help\_cmd))

app.add\_handler(conv\_handler)

app.add\_handler(CommandHandler('list', list\_locations))

app.add\_handler(CallbackQueryHandler(callback\_query\_handler, pattern='^(details:|update\_status:|setstatus:|delete:|noop)'))

print("🤖 البوت يعمل الآن …")

app.run\_polling()

if name == "main":

main()