```python
import os
import re
import json
import logging
import threading
import asyncio

from flask import Flask
from telegram import Update
from telegram.ext import (
    Application,
    CommandHandler,
    MessageHandler,
    ContextTypes,
    filters,
)

logging.basicConfig(level=logging.INFO)
logger = logging.getLogger(__name__)

BOT_TOKEN = os.environ.get("BOT_TOKEN")
ADMIN_ID = os.environ.get("ADMIN_ID")  # (اختیاری، برای افزودن پرامپت جدید) شما
PROMPTS_FILE = os.path.join(os.path.dirname(__file__), "prompts.json")

# ---------- مدیریت فایل پرامپت‌ها ----------

def load_prompts():
    if not os.path.exists(PROMPTS_FILE):
        return {}
    with open(PROMPTS_FILE, "r", encoding="utf-8") as f:
        return json.load(f)


def save_prompts(data):
    with open(PROMPTS_FILE, "w", encoding="utf-8") as f:
        json.dump(data, f, ensure_ascii=False, indent=2)


def extract_code(text: str):
    """از متنی مثل 'کد ۱' یا '1' یا 'پرامپت شماره 3' عدد را استخراج می‌کند."""
    persian_digits = "۰۱۲۳۴۵۶۷۸۹"
    translated = text.translate(str.maketrans(persian_digits, "0123456789"))
    match = re.search(r"\d+", translated)
    return match.group() if match else None
```

```
# ---------- دستورات ربات ----------

async def start(update: Update, context: ContextTypes.DEFAULT_TYPE):
    await update.message.reply_text(
        "سلام! 👋\n"
        "کد پرامپت مورد نظرت رو بفرست تا متن کامل پرامپت رو برات ارسال کنم.\n"
        "مثال: فقط بنویس 1 یا کد ۱\n\n"
        "رو بزن /list برای دیدن لیست همه‌ی کدها دستور."
    )


async def list_prompts(update: Update, context: ContextTypes.DEFAULT_TYPE):
    prompts = load_prompts()
    if not prompts:
        await update.message.reply_text("هنوز هیچ پرامپتی ثبت نشده.")
        return
    lines = ["📋 لیست پرامپت‌ها:\n"]
    for code, item in sorted(prompts.items(), key=lambda x: int(x[0]) if x[0].isd
        lines.append(f"کد {code} — {item.get('title', 'بدون عنوان')}")
    await update.message.reply_text("\n".join(lines))


async def add_prompt(update: Update, context: ContextTypes.DEFAULT_TYPE):
    """فقط ادمین: /addprompt کد|عنوان|متن پرامپت"""
    if not ADMIN_ID or str(update.effective_user.id) != str(ADMIN_ID):
        await update.message.reply_text("این دستور فقط برای مدیر ربات فعاله.")
        return

    raw = update.message.text.partition(" ")[2]
    parts = raw.split("|")
    if len(parts) < 3:
        await update.message.reply_text(
            "فرمت درست:\n/addprompt کد|عنوان|متن پرامپت\n\n"
            "مثال:\n/addprompt 1|این یک نمونه متن پرامپت است|ویدیو تبلیغاتی..."
        )
        return

    code, title, text = parts[0].strip(), parts[1].strip(), "|".join(parts[2:]).s
    prompts = load_prompts()
    prompts[code] = {"title": title, "text": text}
    save_prompts(prompts)
    await update.message.reply_text(f"✅ پرامپت با کد {code} ذخیره شد.")


async def delete_prompt(update: Update, context: ContextTypes.DEFAULT_TYPE):
```

```
    """فقط ادمین: /delprompt کد"""
    if not ADMIN_ID or str(update.effective_user.id) != str(ADMIN_ID):
        await update.message.reply_text("این دستور فقط برای مدیر ربات فعاله.")
        return
    code = update.message.text.partition(" ")[2].strip()
    prompts = load_prompts()
    if code in prompts:
        del prompts[code]
        save_prompts(prompts)
        await update.message.reply_text(f"🗑️ پرامپت با کد {code} حذف شد.")
    else:
        await update.message.reply_text("این کد پیدا نشد.")


async def handle_message(update: Update, context: ContextTypes.DEFAULT_TYPE):
    text = update.message.text or ""
    code = extract_code(text)
    prompts = load_prompts()

    if code and code in prompts:
        item = prompts[code]
        title = item.get("title", "")
        body = item.get("text", "")
        reply = f"📌 {title}\n\n{body}" if title else body
        await update.message.reply_text(reply)
    else:
        await update.message.reply_text(
            "این کد پیدا نشد. برای دیدن لیست کدهای موجود /list رو بزن."
        )


# ---------- راه‌اندازی ربات در یک ترد جدا ----------

def run_bot():
    if not BOT_TOKEN:
        logger.error("BOT_TOKEN تنظیم نشده. متغیر محیطی BOT_TOKEN را در Render ید
        return

    loop = asyncio.new_event_loop()
    asyncio.set_event_loop(loop)

    application = Application.builder().token(BOT_TOKEN).build()
    application.add_handler(CommandHandler("start", start))
    application.add_handler(CommandHandler("list", list_prompts))
    application.add_handler(CommandHandler("addprompt", add_prompt))
    application.add_handler(CommandHandler("delprompt", delete_prompt))
    application.add_handler(MessageHandler(filters.TEXT & ~filters.COMMAND, handl
```

```
    application.run_polling(close_loop=False)


# ---------- سرور Flask (فقط برای اینکه Render سرویس را زنده نگه دارد) ----------

flask_app = Flask(__name__)


@flask_app.route("/")
def health():
    return "Bot is running."


if __name__ == "__main__":
    bot_thread = threading.Thread(target=run_bot, daemon=True)
    bot_thread.start()

    port = int(os.environ.get("PORT", 5000))
    flask_app.run(host="0.0.0.0", port=port)
```