import express from "express";

import fetch from "node-fetch";

const app = express();

app.use(express.json({ type: "\*/\*" }));

// ENV vars from Render

const {

OPENAI\_API\_KEY,

LIVECHAT\_CLIENT\_ID,

LIVECHAT\_ORG\_ID,

LIVECHAT\_BOT\_ID,

LIVECHAT\_BOT\_SECRET,

LIVECHAT\_PAT,

WEBHOOK\_SECRET

} = process.env;

// issue a bot token (valid 24h)

async function getBotToken() {

const res = await fetch("https://api.livechatinc.com/v3.6/configuration/action/issue\_bot\_token", {

method: "POST",

headers: { "Authorization": `Bearer ${LIVECHAT\_PAT}`, "Content-Type": "application/json" },

body: JSON.stringify({

bot\_id: LIVECHAT\_BOT\_ID,

client\_id: LIVECHAT\_CLIENT\_ID,

bot\_secret: LIVECHAT\_BOT\_SECRET,

organization\_id: LIVECHAT\_ORG\_ID

})

});

const data = await res.json();

return data.token;

}

// send reply into chat

async function sendMessage(botToken, chat\_id, text) {

await fetch("https://api.livechatinc.com/v3.6/agent/action/send\_event", {

method: "POST",

headers: { "Authorization": `Bearer ${botToken}`, "Content-Type": "application/json" },

body: JSON.stringify({ chat\_id, event: { type: "message", text } })

});

}

// call ChatGPT

async function askOpenAI(prompt) {

const res = await fetch("https://api.openai.com/v1/responses", {

method: "POST",

headers: { "Authorization": `Bearer ${OPENAI\_API\_KEY}`, "Content-Type": "application/json" },

body: JSON.stringify({

model: "gpt-4o-mini",

input: [

{ role: "system", content: "You are a helpful real estate assistant." },

{ role: "user", content: prompt }

]

})

});

const data = await res.json();

return data.output\_text?.trim() || "I'm here to help!";

}

// webhook endpoint

app.post("/webhooks/livechat", async (req, res) => {

if (req.body.secret\_key !== WEBHOOK\_SECRET) return res.sendStatus(401);

const { action, payload } = req.body;

if (action === "incoming\_event") {

const chat\_id = payload.chat\_id;

const event = payload.event;

if (event?.type === "message" && event?.author\_type === "customer") {

const botToken = await getBotToken();

const reply = await askOpenAI(event.text);

await sendMessage(botToken, chat\_id, reply);

}

}

if (action === "incoming\_chat") {

const chat\_id = payload.chat?.id;

if (chat\_id) {

const botToken = await getBotToken();

await sendMessage(botToken, chat\_id, "Hi 👋 I’m your AI assistant. How can I help?");

}

}

res.send("ok");

});

app.get("/", (req, res) => res.send("Bridge is running ✅"));

const PORT = process.env.PORT || 3000;

app.listen(PORT, () => console.log(`Listening on ${PORT}`));