<!doctype html>

<html lang="en">

<head>

<meta charset="utf-8" />

<meta name="viewport" content="width=device-width,initial-scale=1" />

<title>Pablo's Quick Lines</title>

<style>

:root{

--bg:#f6f7fb; --card:#fff; --accent:#0b84ff; --danger:#e14b4b;

--muted:#888; --shadow:0 6px 18px rgba(13,20,37,0.08);

font-family: -apple-system, BlinkMacSystemFont, "Segoe UI", Roboto, "Helvetica Neue", Arial;

}

body{margin:12px;background:linear-gradient(180deg,#f0f4ff 0%,var(--bg) 100%);color:#111}

header{display:flex;align-items:center;justify-content:space-between;gap:12px;margin-bottom:12px}

header h1{font-size:18px;margin:0}

.wrap{display:grid;grid-template-columns:repeat(3,1fr);gap:12px}

@media(max-width:900px){ .wrap{grid-template-columns:1fr} }

.section{

background:linear-gradient(180deg,#fff,#fcfdff);

border-radius:8px;padding:12px;box-shadow:var(--shadow);border:1px solid rgba(10,10,10,0.03);

min-height:150px;

}

.section h2{margin:0 0 8px 0;font-size:15px}

.addBtn{display:inline-block;background:var(--accent);color:#fff;padding:8px 10px;border-radius:6px;border:none;cursor:pointer;font-weight:600}

.card{background:var(--card);border-radius:8px;padding:10px;margin:10px 0;border:1px solid rgba(0,0,0,0.05);box-shadow:0 2px 6px rgba(10,10,10,0.03)}

.card h3{margin:0 0 6px 0;font-size:13px}

.card .text{white-space:pre-wrap;font-size:13px;color:#222;padding:6px 0;min-height:34px}

.card .controls{display:flex;gap:6px;margin-top:8px;flex-wrap:wrap}

.btn{padding:6px 10px;border-radius:6px;border:none;cursor:pointer;font-weight:600}

.copy{background:#3cb371;color:#fff}

.edit{background:#f0f0f0;color:#111}

.del{background:var(--danger);color:#fff}

.small{font-size:12px;padding:5px 8px}

.footer{margin-top:8px;font-size:12px;color:var(--muted)}

.topbar{display:flex;gap:8px;align-items:center}

.search{flex:1;padding:8px;border-radius:8px;border:1px solid #e6e6e6}

/\* Modal \*/

.modalWrap{position:fixed;inset:0;display:none;align-items:center;justify-content:center;background:rgba(8,10,20,0.45);z-index:40}

.modal{background:#fff;padding:14px;border-radius:10px;max-width:520px;width:94%;box-shadow:var(--shadow)}

.modal input, .modal

textarea{width:100%;padding:8px;margin:6px 0;border-radius:8px;border:1px solid #e6e6e6}

.row{display:flex;gap:8px;align-items:center}

.note{font-size:12px;color:var(--muted);margin-top:6px}

</style>

</head>

<body>

<header>

<div>

<h1>Pablo's Quick Lines</h1>

<div class="note">Tap <strong>Copy</strong> then paste into Roblox chat. Save to Home Screen for quick access.</div>

</div>

<div class="topbar">

<input id="search" class="search" placeholder="Search your lines (title or text)..." />

<button id="resetBtn" class="btn small" title="Reset to defaults">Reset</button>

</div>

</header>

<div class="wrap">

<div class="section" data-section="waiter">

<h2>Waiter (Pablo's) <button class="addBtn" data-section="waiter">+ Add Widget</button></h2>

<div class="cards" id="waiter-cards"></div>

</div>

<div class="section" data-section="host">

<h2>Host (Pablo's) <button class="addBtn" data-section="host">+ Add Widget</button></h2>

<div class="cards" id="host-cards"></div>

</div>

<div class="section" data-section="barista">

<h2>Barista (Pablo's) <button class="addBtn" data-section="barista">+ Add Widget</button></h2>

<div class="cards" id="barista-cards"></div>

</div>

</div>

<div class="footer">

Tip: Add to Home Screen for one-tap access. Your data is stored locally on this device only.

</div>

<!-- Modal for add/edit -->

<div id="modal" class="modalWrap">

<div class="modal">

<h3 id="modalTitle">Add widget</h3>

<input id="widgetTitle" placeholder="Widget title (e.g. Taking the order(s))" />

<textarea id="widgetText" rows="6" placeholder="Type the line here..."></textarea>

<div class="row" style="justify-content:flex-end;margin-top:8px">

<button id="cancel" class="btn edit">Cancel</button>

<button id="save" class="btn copy">Save</button>

</div>

</div>

</div>

<script>

(function(){

const LSKEY = 'pablo\_cards\_v1';

const defaults = {

waiter: [

{id:1, title:"Taking the order(s)", text:"🍝 Ciao ciao! I will be your waiter for today. What can I start you off with? Please remember that you can only order one food and one drink item at a time. 🍷"},

{id:2, title:"Preparing the order(s)", text:"✅ Outstanding choice(s)! I’ll be back with your order(s) shortly. 🍽"},

{id:3, title:"Placing down the order(s)", text:"🍽 I’m back with your meal(s)! Please give me a moment to hand them to you. Bon appetit! ✨"},

{id:4, title:"Refill/Menu", text:"🍹 Hello again! Would you like anything else tonight? I can quickly refill your drink if you'd like as well. I can also get you another menu if you'd like. 📖"},

{id:5, title:"Conclusion", text:"🙏 Thank you for dining at Pablo’s! I’ve had a pleasure serving you today! I hope you had a phenomenal experience and I hope to see you again. Have an amazing day/night! 🌙✨"}

],

host: [

{id:11, title:"Greeting", text:"👋🍝 Welcome to Pablo’s! I hope you are doing well. May I ask how many guests will be dining with you this evening? 🍴"},

{id:12, title:"Seating", text:"✨ Fabulous! Please follow me to your table and I’ll get you all set up for your meal! 🍷"},

{id:13, title:"Table preference", text:"🪑 Would you like to be seated at this table? If not, let me know and we will arrange an alternative ASAP. 🍴"},

{id:14, title:"Conclusion", text:"🙌 Great, a waiter/waitress will arrive shortly. Please let a staff member know if an issue arises. Please enjoy your dining experience! 🍝✨"}

],

barista: [

{id:21, title:"Intro", text:"☕ Ciao! My name is Civil and I will be your barista for today. What can I get you started off with? 🍩"},

{id:22, title:"Preparing the order(s)", text:"🍵 Phenomenal choice(s). I’ll come back with your drink soon! 🥤"},

{id:23, title:"Serving the drinks", text:"🥤 Enjoy your drink(s)! ✨"},

{id:24, title:"Refill/Menu", text:"🍹 I’ve noticed you have finished your drink(s). Please let me know if you would like a refill or something else to refresh you! 🍩"},

{id:25, title:"Conclusion", text:"🙏 Thank you for coming down to Pablo’s today! I hope you had a great time and are well refreshed. I hope to see you another time! ☕✨"}

]

};

let data = loadData();

// DOM refs

const waiterCards = document.getElementById('waiter-

cards');

const hostCards = document.getElementById('host-cards');

const baristaCards = document.getElementById('barista-cards');

const modal = document.getElementById('modal');

const modalTitle = document.getElementById('modalTitle');

const widgetTitle = document.getElementById('widgetTitle');

const widgetText = document.getElementById('widgetText');

const saveBtn = document.getElementById('save');

const cancelBtn = document.getElementById('cancel');

const searchInput = document.getElementById('search');

const resetBtn = document.getElementById('resetBtn');

let editing = null; // {section,id} or null

function loadData(){

try {

const raw = localStorage.getItem(LSKEY);

if(!raw) { localStorage.setItem(LSKEY, JSON.stringify(defaults)); return JSON.parse(JSON.stringify(defaults)); }

return JSON.parse(raw);

} catch(e){

localStorage.setItem(LSKEY, JSON.stringify(defaults));

return JSON.parse(JSON.stringify(defaults));

}

}

function saveData(){

localStorage.setItem(LSKEY, JSON.stringify(data));

}

function resetToDefaults(){

if(confirm('Reset all cards to defaults? This will replace your current saved lines.')) {

data = JSON.parse(JSON.stringify(defaults));

saveData();

renderAll();

}

}

function id(){

return Date.now() + Math.floor(Math.random()\*999);

}

function renderSection(section, container){

container.innerHTML = '';

const q = searchInput.value.trim().toLowerCase();

data[section].forEach(item => {

if(q){

if(!(item.title.toLowerCase().includes(q) || item.text.toLowerCase().includes(q))) return;

}

const card = document.createElement('div');

card.className = 'card';

card.innerHTML = `

<h3>${escapeHtml(item.title)}</h3>

<div class="text" data-id="${item.id}">${escapeHtml(item.text)}</div>

<div class="controls">

<button class="btn copy small" data-action="copy" data-section="${section}" data-id="${item.id}">Copy</button>

<button class="btn edit small" data-action="edit" data-

section="${section}" data-id="${item.id}">Edit</button>

<button class="btn del small" data-action="delete" data-section="${section}" data-id="${item.id}">Delete</button>

</div>

`;

container.appendChild(card);

});

}

function renderAll(){

renderSection('waiter', waiterCards);

renderSection('host', hostCards);

renderSection('barista', baristaCards);

}

// helper: escape plain text to HTML safe

function escapeHtml(str){

}

// open modal to add/edit

function openModal(mode, section, item){

modal.style.display = 'flex';

modalTitle.textContent = mode === 'add' ? 'Add widget' : 'Edit widget';

widgetTitle.value = item?.title || '';

widgetText.value = item?.text || '';

widgetTitle.focus();

editing = mode === 'edit' ? {section, id: item.id} : {section,

id:null};

}

function closeModal(){ modal.style.display='none'; editing=null; widgetTitle.value=''; widgetText.value=''; }

// add or update widget

saveBtn.addEventListener('click', ()=>{

const title = widgetTitle.value.trim() || 'Untitled';

const text = widgetText.value.trim() || '';

if(editing && editing.id){

// update

const arr = data[editing.section];

const idx = arr.findIndex(x=>x.id===editing.id);

if(idx>=0) { arr[idx].title = title; arr[idx].text = text; saveData(); renderAll(); closeModal(); }

} else if(editing){

const newItem = {id:id(), title, text};