<!doctype html>

<html lang="fr">

<head>

<meta charset="utf-8" />

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

<title>ColorIAge magique</title>

<link rel="preconnect" href="https://fonts.googleapis.com">

<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

<link href="https://fonts.googleapis.com/css2?family=Poppins:wght@400;500;700&display=swap" rel="stylesheet">

<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/katex@0.16.9/dist/katex.min.css" xintegrity="sha384-n8MVd4RsNIU0KOVEMQNogNje1upCVQLpbEtjsxCVUvODKI7zDt4BJFGW/2z2neul" crossorigin="anonymous">

<script src="https://cdn.jsdelivr.net/npm/katex@0.16.9/dist/katex.min.js" xintegrity="sha384-XjKyOOlGwcjNTAIQHIpgOno0Hl1YQqzUOEleOLALmuqehneUG+vnGctmUb0ZY0l8" crossorigin="anonymous"></script>

<script src="https://cdn.jsdelivr.net/npm/katex@0.16.9/dist/contrib/auto-render.min.js" xintegrity="sha384-+VBxd3r6XgURPlLHumrCFGdNuT0bqKBUwsuUGewblj+SNrD8RUNhdftvUem3gLRI" crossorigin="anonymous"></script>

<style>

:root{

--bg: #f0f4f8;

--panel: #ffffff;

--ink: #1e293b;

--muted: #64748b;

--border: #e2e8f0;

--primary: #3b82f6;

--cell: 22px;

}

\*{box-sizing:border-box}

body{

margin:0;

background:var(--bg);

color:var(--ink);

font-family: 'Poppins', system-ui, -apple-system, BlinkMacSystemFont, 'Segoe UI', Roboto, Oxygen, Ubuntu, Cantarell, 'Open Sans', 'Helvetica Neue', sans-serif;

font-size:15px;

line-height:1.6;

padding-bottom: 120px; /\* Space for sticky footer \*/

}

header{

padding:1rem 1.5rem;

background: var(--panel);

border-bottom:1px solid var(--border);

position:sticky;

top:0;

z-index:10;

box-shadow: 0 2px 10px rgba(0,0,0,0.05);

}

h1, h3{

margin:0;

font-weight: 700;

}

h1 {

font-size:clamp(20px, 5vw, 28px);

color: var(--primary);

}

.wrap{max-width:1280px;margin:0 auto;padding:1.5rem 1rem;}

.grid{display:grid;gap:1.5rem;grid-template-columns:1fr 1fr}

@media (max-width: 920px){ .grid{grid-template-columns:1fr} }

.card{

background:var(--panel);

border:1px solid var(--border);

border-radius:16px;

padding:1.5rem;

box-shadow: 0 4px 15px rgba(0, 0, 0, 0.07);

}

label{display:block;margin:.5rem 0 .4rem;color:var(--muted); font-weight: 500;}

input[type="number"],input[type="text"],input[type="file"],textarea{

width:100%;

background:#f8fafc;

border:1px solid #cbd5e1;

color:var(--ink);

border-radius:12px;

padding:12px;

font-family: inherit;

font-size: 15px;

}

input:focus, textarea:focus {

outline: 2px solid var(--primary);

border-color: transparent;

}

textarea{min-height:75px}

.row{display:grid;grid-template-columns:1fr 1fr;gap:1rem}

.actions{display:flex;flex-wrap:wrap;gap:10px;margin-top:1rem}

.btn{

appearance:none;

border:none;

background: var(--primary);

color: white;

padding:12px 18px;

border-radius:12px;

cursor:pointer;

font-weight: 500;

font-family: inherit;

font-size: 15px;

transition: transform 0.2s ease, box-shadow 0.2s ease;

}

.btn:hover{

transform: translateY(-2px);

box-shadow: 0 4px 12px rgba(59, 130, 246, 0.3);

}

.btn[disabled]{opacity:.5;cursor:not-allowed; transform: none; box-shadow: none;}

.btn-export{flex-grow:1;font-weight:700;font-size:17px; padding: 1.1rem; text-align: center;}

.btn-enonce{background-color:#2563eb; color:white;}

.btn-enonce:hover { box-shadow: 0 4px 12px rgba(37, 99, 235, 0.4); }

.btn-corr{background-color:#9333ea; color:white;}

.btn-corr:hover { box-shadow: 0 4px 12px rgba(147, 51, 234, 0.4); }

canvas{border:1px solid var(--border);background:#e2e8f0;image-rendering:pixelated;display:block;border-radius:12px}

.tableWrap{overflow:auto;max-height:60vh;border:1px solid var(--border);border-radius:12px}

table.letters{border-collapse:separate;border-spacing:0;margin:0;table-layout:fixed;cursor:pointer;}

table.letters td{box-sizing:border-box;padding:0;text-align:center;font-family:ui-monospace,monospace;color: #334155;width:var(--cell);height:var(--cell);line-height:var(--cell);font-size:var(--font);font-variant-ligatures:none;vertical-align:middle;border:none;outline:1px solid #cbd5e1;position:relative;}

.cell-editor-dropdown{position:absolute;z-index:20;background-color:#ffffff;border:1px solid #94a3b8;border-radius:8px;max-height:150px;overflow-y:auto;min-width:100%;box-shadow: 0 5px 15px rgba(0,0,0,0.1);}

.cell-editor-option{padding:4px 10px;cursor:pointer;color:var(--ink);text-align:center;}

.cell-editor-option:hover{background-color:#eff6ff;}

#palette{display:grid;grid-template-columns:repeat(auto-fill, minmax(90px, 1fr));gap:1rem;margin-top:1rem;}

.color-item{display:flex;flex-direction:column;align-items:center;gap:6px;}

.color-letter{font-weight:700;color:var(--muted); font-size: 14px;}

.color-box{width:100%;height:45px;border-radius:12px;display:flex;align-items:center;justify-content:center;cursor:pointer;font-size:14px;position:relative; border: 1px solid rgba(0,0,0,0.05);}

.color-box .color-name{padding:0 6px; font-weight: 500;}

.color-box .color-input{width:100%;height:100%;border:none;background:transparent;color:inherit;text-align:center;padding:0 6px;font-size:14px;border-radius:12px;outline:2px solid var(--primary);}

small.muted{color:#94a3b8}

.status{color:var(--primary);margin-top:1rem;min-height:1em; font-weight: 500;}

.qa-list{margin-top:1rem;border-top:1px solid var(--border);padding-top:1rem}

.qa-list h4{margin:6px 0 8px 0;color:var(--muted)}

.qa-list ol{margin:0;padding-left:20px}

/\* Overlay zone \*/

.overlay{position:relative;display:inline-block;border:1px solid var(--border);border-radius:12px;overflow:hidden;background:#f8fafc}

.overlay .lettersHost{position:absolute;inset:0;display:flex;align-items:center;justify-content:center}

.overlay .lettersHost .sizer{display:block}

.overlay canvas{position:absolute;inset:0;pointer-events:none;}

.export-footer {

position: sticky;

bottom: 0;

left: 0;

width: 100%;

background: rgba(255, 255, 255, 0.85);

backdrop-filter: blur(8px);

-webkit-backdrop-filter: blur(8px);

padding: 1rem;

border-top: 1px solid var(--border);

box-shadow: 0 -4px 15px rgba(0,0,0,0.08);

z-index: 100;

}

.export-actions {

max-width: 800px;

margin: 0 auto;

display: grid;

grid-template-columns: 1fr 1fr;

gap: 1rem;

}

.fallback{display:none;margin-top:1rem}

.fallback textarea{width:100%;min-height:160px}

</style>

</head>

<body>

<header>

<h1>ColorIAge magique 🎨</h1>

</header>

<div class="wrap">

<div class="grid">

<div class="card">

<h3>1. Paramètres du dessin</h3>

<label>Image à pixeliser</label>

<input type="file" id="file" accept="image/\*" />

<div class="row">

<div>

<label>Titre du colorIAge</label>

<input type="text" id="theme" />

</div>

</div>

<div class="row">

<div>

<label>Largeur de grille</label>

<input type="number" id="gridW" min="2" max="128" value="32" />

</div>

<div>

<label>Hauteur de grille</label>

<input type="number" id="gridH" min="2" max="128" value="" />

</div>

</div>

<div class="row" style="margin-top:8px">

<div>

<label>Nombre de couleurs</label>

<input type="number" id="k" min="2" max="26" value="5" />

</div>

</div>

</div>

<div class="card">

<h3>2. Aperçu et personnalisation</h3>

<label>Grille (cliquez sur une case pour la modifier)</label>

<div id="overlay" class="overlay">

<div class="lettersHost"><div id="gridOverlaySizer" class="sizer"><div id="gridOverlayWrap"></div></div></div>

<canvas id="cnv" width="32" height="32"></canvas>

</div>

<div class="row" style="margin-top:8px">

<div>

<label>Opacité de l'image</label>

<input id="alpha" type="range" min="0" max="100" value="50" />

<small class="muted">0 : lettres seules • 100 : image visible</small>

</div>

<div></div>

</div>

<label>Palette <small class="muted">(Cliquez sur un nom pour le modifier)</small></label>

<div id="palette"></div>

</div>

</div>

<div class="card" style="margin-top:1.5rem">

<h3 style="margin:0 0 8px">3. Questions et réponses</h3>

<div id="qaManual" class="row" style="margin-top:10px"></div>

<div class="qa-list" id="qaPreview" style="display:none">

<h4>Aperçu</h4>

<ol id="qaList"></ol>

</div>

<small class="muted">Vous pouvez éditer les champs ci‑dessus à tout moment ; l'export LaTeX reprendra exactement ces contenus.</small>

</div>

</div>

<div class="export-footer">

<div class="export-actions">

<button id="btnExportEnonce" class="btn btn-export btn-enonce">📄 Exporter l'énoncé (LaTeX)</button>

<button id="btnExportCorr" class="btn btn-export btn-corr">✅ Exporter la correction (LaTeX)</button>

</div>

<div id="fallback" class="fallback" style="max-width: 800px; margin: 1rem auto 0;">

<small class="muted">Export bloqué ? Le fichier s'ouvre ci‑dessous : copiez/collez.</small>

<textarea id="fallbackText" readonly></textarea>

</div>

</div>

<script>

(function(){

'use strict';

const qs=s=>document.querySelector(s);

const WORKER\_URL=''; //

const val=(sel,def="")=>{const el=qs(sel);return (el && el.value!==undefined && el.value!==null && el.value!=='' )?el.value:def};

function setStatus(msg){ const s=qs('#status'); if(s) s.textContent=msg||''; }

// --- Export robuste (Canvas/Canvas Share/iframes) ---

function openInNewTab(name, text, mime){

const dataUrl = 'data:'+(mime||'text/plain')+';charset=utf-8,'+encodeURIComponent(text||'');

const w = window.open(dataUrl, '\_blank', 'noopener');

if(!w){ setStatus('Pop‑up bloqué : autorisez l\'ouverture d\'onglets.'); }

const fb=qs('#fallback'); const ta=qs('#fallbackText');

if(fb && ta){ ta.value = text || ''; fb.style.display='block'; }

}

function download(name, text, mime){

const str = (text==null ? '' : String(text));

const type = mime || 'text/plain;charset=utf-8';

try{

const blob = new Blob([str], {type:type});

// IE/Edge legacy

if(window.navigator && 'msSaveOrOpenBlob' in window.navigator){

// @ts-ignore

window.navigator.msSaveOrOpenBlob(blob, name);

return;

}

const url = URL.createObjectURL(blob);

const a = document.createElement('a'); a.href=url; a.download=name; a.rel='noopener'; a.target='\_self'; a.style.display='none';

document.body.appendChild(a);

a.click();

setTimeout(()=>{ URL.revokeObjectURL(url); a.remove(); }, 800);

}catch(e){

console.warn('Download blocked, fallback to new tab', e);

openInNewTab(name, str, type.split(';')[0]);

}

}

function srgbToLinear(c){ c/=255; return (c<=0.04045)? c/12.92 : Math.pow((c+0.055)/1.055,2.4); }

function rgb2xyz(rgb){ const r=srgbToLinear(rgb[0]); const g=srgbToLinear(rgb[1]); const b=srgbToLinear(rgb[2]); return [0.4124\*r+0.3576\*g+0.1805\*b,0.2126\*r+0.7152\*g+0.0722\*b,0.0193\*r+0.1192\*g+0.9505\*b]; }

function xyz2lab(xyz){ const Xr=0.95047, Yr=1.00000, Zr=1.08883; const x=xyz[0]/Xr, y=xyz[1]/Yr, z=xyz[2]/Zr; const f=t=> (t>0.008856)? Math.pow(t,1/3):(7.787\*t+16/116); const fx=f(x), fy=f(y), fz=f(z); return [116\*fy-16, 500\*(fx-fy), 200\*(fy-fz)]; }

function rgb2lab(c){ return xyz2lab(rgb2xyz(c)); }

function deltaE76(l1,l2){ const dL=l1[0]-l2[0], da=l1[1]-l2[1], db=l1[2]-l2[2]; return Math.sqrt(dL\*dL+da\*da+db\*db); }

const hex=v=> '#'+v.map(x=>x.toString(16).padStart(2,'0')).join('');

function autoPickCenters(colors, K){

let centers=[], labs=[]; const thresholds=[44,40,36,32,28,24,20,16,12,8];

for(let t=0;t<thresholds.length;t++){

const thr=thresholds[t]; centers.length=0; labs.length=0;

for(let i=0;i<colors.length && centers.length<K;i++){

const rgb=colors[i].rgb; const lab=rgb2lab(rgb); let ok=true;

for(let j=0;j<labs.length;j++){ if(deltaE76(lab,labs[j])<thr){ ok=false; break; } }

if(ok){ centers.push(rgb); labs.push(lab); }

}

if(centers.length>=K) break;

}

for(let k=0;k<colors.length && centers.length<K;k++){

const rgb2=colors[k].rgb; if(!centers.some(c=>c[0]===rgb2[0]&&c[1]===rgb2[1]&&c[2]===rgb2[2])) centers.push(rgb2);

}

return centers;

}

const imgEl=new Image(); imgEl.decoding='async'; imgEl.style.display='none'; document.body.appendChild(imgEl);

function scaleCanvasCSS(cvs,W,H){

const maxW=Math.min(document.body.clientWidth-36, 900);

const s=Math.max(6, Math.min(28, Math.floor(maxW/Math.max(1,W))));

const cssW=(W\*s)+'px';

const cssH=(H\*s)+'px';

// Canvas size (CSS)

cvs.style.width=cssW; cvs.style.height=cssH;

// Overlay container sizes to avoid collapse

const ov=qs('#overlay'); if(ov){ ov.style.width=cssW; ov.style.height=cssH; }

const sizer=qs('#gridOverlaySizer'); if(sizer){ sizer.style.width=cssW; sizer.style.height=cssH; }

// Letter cell size and font size tied to the SAME pixel scale 's'

const wrap=qs('#gridOverlayWrap'); if(wrap){

wrap.style.setProperty('--cell', s+'px');

wrap.style.setProperty('--font', Math.max(7, Math.floor(s\*0.50))+'px');

}

window.\_\_cellSize = s; // remember for perfect alignment

return s;

}

function sizeLetterCells(W){

const wrap=qs('#gridOverlayWrap'); if(!wrap) return;

const s = window.\_\_cellSize || 16;

wrap.style.setProperty('--cell', s+'px');

wrap.style.setProperty('--font', Math.max(7, Math.floor(s\*0.52))+'px');

}

function closeAllCellEditors() {

document.querySelectorAll('.cell-editor-dropdown').forEach(editor => editor.remove());

}

document.addEventListener('mousedown', function(event) {

if (!event.target.closest('td') && !event.target.closest('.cell-editor-dropdown')) {

closeAllCellEditors();

}

});

function handleCellClick(event) {

event.preventDefault();

event.stopPropagation();

closeAllCellEditors();

const td = event.currentTarget;

const x = parseInt(td.dataset.x, 10);

const y = parseInt(td.dataset.y, 10);

const letters = (window.\_\_letters || '').split('');

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

dropdown.className = 'cell-editor-dropdown';

letters.forEach(letter => {

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

option.className = 'cell-editor-option';

option.textContent = letter;

option.dataset.letter = letter;

option.addEventListener('mousedown', (e) => {

e.stopPropagation();

const newLetter = option.dataset.letter;

td.textContent = newLetter;

if (window.\_\_grid && window.\_\_grid[y]) {

window.\_\_grid[y][x] = newLetter;

}

updateCanvasCell(x, y, newLetter);

window.\_\_gridCSV = window.\_\_grid.map(r => r.join(',')).join('\n');

closeAllCellEditors();

});

dropdown.appendChild(option);

});

td.appendChild(dropdown);

}

function updateCanvasCell(x, y, newLetter) {

const cvs = qs('#cnv');

if (!cvs) return;

const ctx = cvs.getContext('2d');

const letterIndex = (window.\_\_letters || '').indexOf(newLetter);

let color = 'rgba(0,0,0,0)'; // Transparent for blank

if (letterIndex > -1 && window.\_\_centers[letterIndex]) {

const rgb = window.\_\_centers[letterIndex];

color = `rgb(${rgb[0]}, ${rgb[1]}, ${rgb[2]})`;

}

ctx.fillStyle = color;

ctx.fillRect(x, y, 1, 1);

}

function renderTable(grid, W){

const wrap=qs('#gridOverlayWrap'); if(!wrap) return;

const table=document.createElement('table'); table.className='letters';

const tbody=document.createElement('tbody');

for(let y=0;y<grid.length;y++){

const tr=document.createElement('tr');

for(let x=0;x<W;x++){

const td=document.createElement('td');

td.textContent=grid[y][x];

td.dataset.x = x;

td.dataset.y = y;

td.addEventListener('mousedown', handleCellClick);

tr.appendChild(td);

}

tbody.appendChild(tr);

}

table.appendChild(tbody);

wrap.innerHTML=''; wrap.appendChild(table);

sizeLetterCells(W);

}

function applyAlpha(){

const a=parseInt(val('#alpha','0'),10)||0;

const maxOpacity = 0.85; // lettres toujours visibles

const cvs=qs('#cnv');

if(cvs){ cvs.style.opacity = ((a/100)\*maxOpacity).toString(); }

}

function renderPalette() {

const pal = qs('#palette');

if (!pal) return;

pal.innerHTML = '';

const centers = window.\_\_centers || [];

const letters = window.\_\_letters || '';

centers.forEach((c, i) => {

const colorName = (window.\_\_colorNames && window.\_\_colorNames[i]) ? window.\_\_colorNames[i] : '';

const letter = letters[i] || '?';

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

item.className = 'color-item';

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

letterDiv.className = 'color-letter';

letterDiv.textContent = letter;

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

box.className = 'color-box';

box.style.background = hex(c);

box.style.color = lum(c) > 0.60 ? 'black' : 'white';

const nameSpan = document.createElement('span');

nameSpan.className = 'color-name';

nameSpan.textContent = colorName;

const nameInput = document.createElement('input');

nameInput.type = 'text';

nameInput.className = 'color-input';

nameInput.value = colorName;

nameInput.style.display = 'none';

box.appendChild(nameSpan);

box.appendChild(nameInput);

item.appendChild(letterDiv);

item.appendChild(box);

pal.appendChild(item);

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

nameSpan.style.display = 'none';

nameInput.style.display = 'block';

nameInput.focus();

});

const updateName = () => {

const newName = nameInput.value.trim();

window.\_\_colorNames[i] = newName;

nameSpan.textContent = newName;

nameSpan.style.display = 'block';

nameInput.style.display = 'none';

};

nameInput.addEventListener('blur', updateName);

nameInput.addEventListener('keydown', (e) => {

if (e.key === 'Enter') {

updateName();

}

});

});

}

function process(){

try{

const W=parseInt(val('#gridW','32'),10)||32; const H=parseInt(val('#gridH','32'),10)||32; const K=Math.max(2, Math.min((parseInt(val('#k','7'),10)||7), 26));

const cvs=qs('#cnv'); if(!cvs) return; cvs.width=W; cvs.height=H; const ctx=cvs.getContext('2d'); ctx.imageSmoothingEnabled=false; scaleCanvasCSS(cvs,W,H);

const tmp=document.createElement('canvas'); tmp.width=W; tmp.height=H; const tctx=tmp.getContext('2d'); tctx.imageSmoothingEnabled=false;

if(imgEl && imgEl.naturalWidth>0){

const ratio=Math.min(W/imgEl.naturalWidth, H/imgEl.naturalHeight);

const w=Math.max(1, Math.round(imgEl.naturalWidth\*ratio));

const h=Math.max(1, Math.round(imgEl.naturalHeight\*ratio));

const ox=Math.floor((W-w)/2), oy=Math.floor((H-h)/2);

tctx.clearRect(0,0,W,H); tctx.drawImage(imgEl,0,0,imgEl.naturalWidth,imgEl.naturalHeight,ox,oy,w,h);

} else { tctx.fillStyle='#222'; tctx.fillRect(0,0,W,H); }

const id=tctx.getImageData(0,0,W,H); const data=id.data;

const counts=new Map();

for(let i=0;i<W\*H;i++){

const r=data[4\*i], g=data[4\*i+1], b=data[4\*i+2];

const key=r+','+g+','+b; counts.set(key,(counts.get(key)||0)+1);

}

const colors=[...counts.entries()].map(([k,c])=>({rgb:k.split(',').map(n=>parseInt(n,10)), count:c})).sort((a,b)=>b.count-a.count);

const centers=autoPickCenters(colors, Math.min(K, colors.length||K));

const alphabet='ABCDEFGHIJKLMNOPQRSTUVWXYZ'; const letters=alphabet.slice(0,centers.length).split('');

const prevLen=(window.\_\_letters||'').length; if(prevLen!==letters.length) renderQA(letters.length);

window.\_\_grid=[];

for(let y=0;y<H;y++){

const row=[];

for(let x=0;x<W;x++){

const idx=y\*W+x; const rr=data[4\*idx], gg=data[4\*idx+1], bb=data[4\*idx+2];

let best=0, bd=1e9;

for(let p=0;p<centers.length;p++){

const c=centers[p]; const d=(rr-c[0])\*(rr-c[0])+(gg-c[1])\*(gg-c[1])+(bb-c[2])\*(bb-c[2]);

if(d<bd){ bd=d; best=p; }

}

row.push(letters[best]||'?');

const cc=centers[best]; data[4\*idx]=cc[0]; data[4\*idx+1]=cc[1]; data[4\*idx+2]=cc[2]; data[4\*idx+3]=255;

}

window.\_\_grid.push(row);

}

const ctx2=cvs.getContext('2d');

ctx2.putImageData(id,0,0);

renderTable(window.\_\_grid, W);

window.\_\_gridCSV = window.\_\_grid.map(r=>r.join(',')).join('\n');

window.\_\_letters = letters.join('');

window.\_\_centers = centers;

const tempItems = centers.map((c, i) => ({rgb: c, name: '', letter: letters[i]}));

const initialNames = ensureUnique(tempItems);

window.\_\_colorNames = initialNames.map(item => item.name);

renderPalette();

applyAlpha();

refreshPreview();

setStatus('');

}catch(err){ console.error(err); setStatus('Erreur d\'affichage : '+err.message); }

}

function renderQA(n){

const cont=qs('#qaManual'); if(!cont) return; cont.innerHTML='';

cont.className='row';

for(let i=0;i<n;i++){

const L=String.fromCharCode(65+i);

const q=document.createElement('input'); q.type='text'; q.placeholder=L+' : question'; q.autocomplete='off'; q.spellcheck=false; q.addEventListener('input', refreshPreview);

const a=document.createElement('input'); a.type='text'; a.placeholder=L+' : réponse'; a.autocomplete='off'; q.spellcheck=false; a.addEventListener('input', refreshPreview);

cont.appendChild(q); cont.appendChild(a);

}

}

function refreshPreview(){

const cont=qs('#qaManual'); const list=qs('#qaList'); const shell=qs('#qaPreview');

if(!cont||!list||!shell) return;

const inputs=[...cont.querySelectorAll('input')];

list.innerHTML='';

for(let i=0;i<inputs.length;i+=2){

const q=inputs[i].value||''; const r=inputs[i+1].value||''; if(!q && !r) continue;

const li=document.createElement('li'); li.textContent=q+(r? ' ➜ '+r:''); list.appendChild(li);

}

shell.style.display = list.children.length? 'block':'none';

}

function texPreamble(){

return String.raw`\documentclass[12pt]{article}

\usepackage{lmodern}

\usepackage[T1]{fontenc}

\usepackage[french]{babel}

\usepackage[a4paper,portrait,margin=2cm]{geometry}

\usepackage{amsmath}

\usepackage{tcolorbox}

\usepackage{amssymb}

\usepackage{amsthm}

\usepackage{lastpage}

\usepackage{fancyhdr}

\usepackage{accents}

\usepackage{enumitem}

\usepackage{icomma}

\usepackage{graphicx}

\usepackage{fancybox}

\usepackage[table]{xcolor}

\usepackage{colortbl}

\usepackage{eurosym}

\usepackage{array}

\usepackage{multicol}

\usepackage{tikz}

\usetikzlibrary{angles, quotes, decorations.markings, calc,decorations.pathmorphing}

\usepackage{sectsty}

\usepackage{siunitx}

\usepackage{PixelArtTikz}

\renewcommand{\familydefault}{\sfdefault}

\fontsize{14}{16.8}\selectfont

\pagestyle{fancy}

\fancyhf{}

\rfoot{\texttt{Généré par IA}}

\usepackage{ccicons}\lfoot{\ccby}

\setlength\parindent{0pt}`;

}

function makeColorDefines(C){ return C.map((rgb,i)=>`\\definecolor{C${i+1}}{RGB}{${rgb[0]},${rgb[1]},${rgb[2]}}`).join("\n"); }

const BASIC\_COLORS=[

{name:'noir',rgb:[0,0,0]},{name:'gris',rgb:[128,128,128]},{name:'blanc',rgb:[255,255,255]},

{name:'rouge',rgb:[220,20,60]},{name:'orange',rgb:[255,140,0]},{name:'jaune',rgb:[255,215,0]},

{name:'marron',rgb:[139,69,19]},{name:'beige',rgb:[245,222,179]},

{name:'vert',rgb:[34,139,34]},{name:'vert clair',rgb:[144,238,144]},{name:'vert foncé',rgb:[0,100,0]},

{name:'bleu',rgb:[65,105,225]},{name:'bleu clair',rgb:[135,206,235]},{name:'turquoise',rgb:[64,224,208]},

{name:'violet',rgb:[138,43,226]},{name:'rose',rgb:[255,105,180]}

];

function d2(a,b){const dr=a[0]-b[0],dg=a[1]-b[1],db=a[2]-b[2];return dr\*dr+dg\*dg+db\*db;}

function colorNameFr(rgb){ let best=BASIC\_COLORS[0],bestd=1e18; for(const c of BASIC\_COLORS){const dd=d2(rgb,c.rgb); if(dd<bestd){bestd=dd; best=c;}} return best.name; }

function lum(rgb){ const [r,g,b]=rgb.map(v=>{v/=255;return (v<=0.04045)?v/12.92:Math.pow((v+0.055)/1.055,2.4);}); return 0.2126\*r+0.7152\*g+0.0722\*b; }

function ensureUnique(items){

const groups=new Map();

for(const it of items){

const base=colorNameFr(it.rgb);

it.base=base; it.\_l=lum(it.rgb);

if(!groups.has(base)) groups.set(base,[]);

groups.get(base).push(it);

}

for(const [base,arr] of groups){

arr.sort((a,b)=>a.\_l-b.\_l);

const n=arr.length;

if(n===1){ arr[0].name=base; }

else if(n===2){ arr[0].name=base+' foncé'; arr[1].name=base+' clair'; }

else {

arr[0].name=base+' foncé';

arr[n-1].name=base+' clair';

const mids=arr.slice(1,n-1);

if(mids.length===1){ mids[0].name=base; }

else { mids.forEach((it,i)=> it.name=base+' '+(i+1)); }

}

}

const seen=new Map();

for(const it of items){ let name=it.name; let k=1; while(seen.has(name)){ k++; name=it.name+' '+k; } it.name=name; seen.set(name,true); }

return items;

}

const shuffle=a=>{ for(let i=a.length-1;i>0;i--){ const j=(Math.random()\*(i+1))|0; [a[i],a[j]]=[a[j],a[i]]; } return a; };

function makeRightTables(QA, LIMIT=42){

if(!QA||!QA.length) return '';

const centers=(window.\_\_centers||[]);

let items = QA.map(it => ({

idx: it.index,

rgb: centers[it.index] || [0, 0, 0],

name: (window.\_\_colorNames && window.\_\_colorNames[it.index]) ? window.\_\_colorNames[it.index] : colorNameFr(centers[it.index] || [0, 0, 0]),

ans: it.ans || ''

}));

shuffle(items);

const unit=s=> Math.max(3, Math.ceil(String(s||'').length\*0.9)+2);

const blocks=[]; let cur=[], acc=0;

for(const it of items){

if (!it) continue; // Safety check

const w=Math.max(unit(it.name), unit(it.ans));

if(acc + w > LIMIT && cur.length){ blocks.push(cur); cur=[it]; acc=w; }

else { cur.push(it); acc+=w; }

}

if(cur.length) blocks.push(cur);

return blocks.map(block=>{

const cols='|'+Array(block.length).fill('c').join('|')+'|';

const head=block.map(it=>{ const tcol=(lum(it.rgb)>0.60?'black':'white'); return String.raw` \cellcolor{C${it.idx+1}}\textcolor{${tcol}}{${it.name}}`; }).join(' & ') + String.raw` \\ \hline

`;

const vals=block.map(it=> latexEscape(it.ans)).join(' & ') + String.raw` \\ \hline

`;

return String.raw`\begin{tabular}{${cols}}

\hline

${head}${vals}\end{tabular}`;

}).join(String.raw`\n\medskip\n\n`);

}

// Préserve $...$ et $$...$$ ; échappe le reste

function latexEscape(s){

if(!s) return '';

const ESC = t => String(t)

.replace(/\\/g,'\\textbackslash{}')

.replace(/([{}%&#\_])/g,'\\$1')

.replace(/~/g,'\\textasciitilde{}')

.replace(/\^/g,'\\textasciicircum{}');

const TOKO='\uE000', TOKC='\uE001';

const tokens=[];

let tmp=String(s);

tmp = tmp.replace(/\$\$([\s\S]\*?)\$\$/g,(m,inner)=>{ tokens.push({d:true,v:inner}); return TOKO+(tokens.length-1)+TOKC; });

tmp = tmp.replace(/\$([^$]+)\$/g,(m,inner)=>{ tokens.push({d:false,v:inner}); return TOKO+(tokens.length-1)+TOKC; });

tmp = ESC(tmp);

tmp = tmp.replace(new RegExp(TOKO+'(\\d+)'+TOKC,'g'), (m,i)=>{

const t=tokens[Number(i)];

return t.d ? `$$${t.v}$$` : `$${t.v}$`;

});

return tmp;

}

function buildTex(corr){

const bn='Pixel-maths';

const letters=(window.\_\_letters||'AB').split('');

const qaEls=[...document.querySelectorAll('#qaManual input')];

const QA=[]; for(let i=0;i<qaEls.length;i+=2){ const L=String.fromCharCode(65+i/2); const qv=qaEls[i].value||('Question '+L); const av=qaEls[i+1].value||''; QA.push({index:i/2, letter:L, q:qv, ans:av}); }

const mid = Math.ceil(QA.length/2);

const leftList = QA.slice(0, mid).map(it=> String.raw`\item ${latexEscape(it.q)} \\ \medskip`).join('\n');

const rightList = QA.slice(mid).map(it=> String.raw`\item ${latexEscape(it.q)} \\ \medskip`).join('\n');

const couleursList=(window.\_\_centers||[]).map((\_,i)=>'C'+(i+1)).join(',');

const colorDefs=makeColorDefines(window.\_\_centers||[]);

const csvRaw=(window.\_\_gridCSV||'').replace(/\\/g,'\\\\');

const pre = texPreamble()+"\n"+colorDefs+"\n"+String.raw`\usepackage{filecontents}

\begin{filecontents\*}{${bn}.csv}

${csvRaw}

\end{filecontents\*}

`;

const codes=letters.join('');

const theme = latexEscape(val('#theme',''));

const title = `ColorIAge Magique - ${theme}${corr ? ' (Correction)' : ''}`;

const startRight = (mid+1);

// Préparer les boîtes mesurables pour calculer la hauteur restante

const questionsBox = String.raw`\newsavebox{\QuestionsBox}

\sbox{\QuestionsBox}{%

\begin{minipage}[t]{\ColW}

\begin{enumerate}[label=\fbox{\Alph\*},leftmargin=\*]

${leftList}

\end{enumerate}

\end{minipage}%

\hspace{\Gutter}\vrule width

\MidRule\hspace{\Gutter}%

\begin{minipage}[t]{\ColW}

\begin{enumerate}[label=\fbox{\Alph\*},leftmargin=\*,start=${startRight}]

${rightList}

\end{enumerate}

\end{minipage}%

}`;

const paletteTables = makeRightTables(QA, 90);

const paletteBox = String.raw`\newsavebox{\PaletteBox}

\sbox{\PaletteBox}{%

\begin{minipage}{\linewidth}

\centering

${paletteTables}

\end{minipage}

}`;

const body=String.raw`\begin{document}

\begin{center}

\textbf{${title}}

\end{center}

\text Résoudre les problèmes associés à chaque lettre. Penser à détailler les calculs et à écrire une phrase réponse. À l'aide du tableau, trouver la couleur correspondant à chaque lettre puis colorier le colorIAge.

\medskip

% ---- Réglages du séparateur central ----

\newlength{\Gutter}

\setlength{\Gutter}{6mm}

% espace de chaque côté de la ligne

\newlength{\MidRule}

\setlength{\MidRule}{0.6pt}

% épaisseur de la ligne

\newlength{\ColW}

\setlength{\ColW}{\dimexpr \textwidth/2 - \Gutter - \MidRule/2 \relax}

${questionsBox}

${paletteBox}

\usebox{\QuestionsBox}

% Hauteur disponible pour la grande grille = hauteur de page - (questions + palette + marges fixes)

\newlength{\GridAvailH}

\setlength{\GridAvailH}{\dimexpr \textheight - \pagetotal - \ht\QuestionsBox - 3\ht\PaletteBox \relax}

% Grande grille de lettres ajustée automatiquement en hauteur

\begin{center}

\resizebox{!}{\GridAvailH}{%

\PixelArtTikz[Codes=${codes},Couleurs={${couleursList}}${corr?',Correction':''}]{${bn}.csv}

}

\end{center}

% Palette de couleurs tout en bas

\begin{center}

\usebox{\PaletteBox}

\end{center}

\end{document}`;

return pre+body;

}

// ==== IA ====

function extractJSON(text){

try{ return JSON.parse(text);}catch(\_){/\* try fenced \*/}

const m = text.match(/\{[\s\S]\*\}/);

if(m){ try{ return JSON.parse(m[0]); }catch(\_e){} }

return null;

}

function buildAIInput(payload){

// Le worker/Responses attend un "input" texte. On sérialise nos consignes et paramètres.

const {subject, level, theme, count} = payload;

// Instructions très structurées pour forcer le respect des contraintes critiques.

return `Tu es un générateur d'exercices mathématiques extrêmement rigoureux. Ta mission est de créer une liste de questions-réponses pour un professeur. Ta réputation dépend de ta capacité à suivre les règles à la lettre. Un échec à suivre une seule règle est inacceptable.

\*\*DÉTAILS DE LA MISSION :\*\*

- \*\*Matière\*\*: ${subject}

- \*\*Niveau\*\*: ${level}

- \*\*Thème\*\*: ${theme}

- \*\*Quantité exacte\*\*: ${count} paires question/réponse.

\*\*PROTOCOLE DE GÉNÉRATION EN 3 ÉTAPES OBLIGATOIRES :\*\*

\*\*ÉTAPE 1 : GÉNÉRATION BRUTE\*\*

- Crée ${count} questions variées sur le thème donné, adaptées au niveau.

- Pour chaque question, fournis une réponse numérique ou LaTeX très courte.

- S'il y a une écriture mathématique (fraction, racine carrée, etc ...), utiliser le format "$\[Formule Latex]$" au sein du texte renvoyé.

\*\*ÉTAPE 2 : VÉRIFICATION CRITIQUE (LA PLUS IMPORTANTE)\*\*

- \*\*Contrôle 1 (Unités)\*\* : Vérifie chaque réponse. AUCUNE réponse ne doit contenir d'unité (cm, €, kg, etc.). C'est une erreur grave. Corrige si nécessaire.

- \*\*Contrôle 2 (Unicité)\*\* : C'est le point de contrôle le plus crucial. Compare toutes les réponses entre elles. \*\*IL EST ABSOLUMENT INTERDIT D'AVOIR DEUX RÉPONSES IDENTIQUES.\*\* Si tu trouves un doublon, tu DOIS modifier la ou les questions correspondantes pour obtenir une nouvelle réponse unique. Répète ce contrôle jusqu'à ce qu'il n'y ait plus aucun doublon. L'échec sur ce point rend la génération inutile.

\*\*ÉTAPE 3 : FORMATAGE FINAL\*\*

- Une fois que toutes les vérifications de l'étape 2 sont passées avec succès, formate le résultat final \*\*uniquement\*\* en JSON strict, comme ceci : \`{"items":[{"question":"...","reponse":"..."}, ...]}\`.

- Le JSON doit contenir exactement ${count} items.

- N'inclus aucun texte, commentaire ou explication en dehors de l'objet JSON.

Le respect absolu de l'étape 2 (en particulier le contrôle de l'unicité) est la condition sine qua non de la réussite de cette mission.`;

}

async function generateWithAI(){

if (!WORKER\_URL) {

// NOTE: J'ai retiré la condition qui bloquait si l'URL était vide, car elle est maintenant définie.

// Si vous remettez une URL vide, le message d'erreur s'affichera à nouveau.

}

const count = parseInt(val('#k','0'),10) || (window.\_\_letters? window.\_\_letters.length: 0) || 8;

const payloadClient = {

subject: val('#aiSubject','Mathématiques'),

level: val('#aiLevel','6e'),

theme: val('#aiTheme', val('#theme','')),

count

};

setStatus('Génération IA en cours…');

try{

// MODIFICATION : La structure de la requête est simplifiée pour notre worker

const prompt = buildAIInput(payloadClient);

const res = await fetch(WORKER\_URL, {

method: 'POST',

headers: { 'Content-Type': 'application/json' },

body: JSON.stringify({

prompt: prompt

})

});

if(!res.ok){

const errorData = await res.json().catch(() => ({error: "Réponse invalide du worker"}));

throw new Error(`HTTP ${res.status} — ${errorData.error || 'Erreur inconnue'}`);

}

// La réponse est maintenant la réponse directe d'OpenAI

const parsed = await res.json();

// On cherche la réponse dans le format de l'API Chat Completions

if(parsed && parsed.choices && parsed.choices[0] && parsed.choices[0].message && parsed.choices[0].message.content){

const bodyText = parsed.choices[0].message.content;

const maybe = extractJSON(bodyText);

if(maybe && Array.isArray(maybe.items)) {

applyQA(maybe.items);

setStatus('Questions insérées (IA).');

refreshPreview();

return;

}

}

// Si le format n'est pas bon, on affiche une erreur

console.debug('Réponse IA inattendue:', parsed);

setStatus('Réponse IA reçue mais format inattendu.');

}catch(e){ console.error(e); setStatus('Échec génération IA : '+e.message); }

}

function applyQA(items){

const cont=qs('#qaManual'); if(!cont) return;

const n = Math.max( (window.\_\_letters||'').length, items.length );

renderQA(n);

const inputs=[...cont.querySelectorAll('input')];

for(let i=0;i<n && i<items.length;i++){

const q=items[i].question||items[i].texte||items[i].q||'';

const r=items[i].reponse||items[i].réponse||items[i].a||items[i].answer||'';

if(inputs[2\*i]) inputs[2\*i].value=q;

if(inputs[2\*i+1]) inputs[2\*i+1].value=r;

}

refreshPreview();

}

function wire(){

qs('#file').addEventListener('change', e=>{

try{

const f=e.target.files && e.target.files[0]; if(!f) return;

const r=new FileReader();

r.onload=async ev=>{ try{ imgEl.onload=()=>process(); imgEl.onerror=()=>{setStatus('Image illisible');}; imgEl.src=ev.target.result; await imgEl.decode().catch(()=>{}); process(); }catch(err){ setStatus('Chargement image: '+err.message); } };

r.readAsDataURL(f);

}catch(err){ setStatus('Lecture fichier: '+err.message); }

});

['#gridW','#gridH','#k','#theme'].forEach(id=>{ const el=qs(id); if(el) el.addEventListener('input', process); });

qs('#alpha').addEventListener('input', applyAlpha);

qs('#btnExportEnonce').addEventListener('click', ()=>{ const tex=buildTex(false); download('enonce.tex', tex, 'application/x-tex;charset=utf-8'); });

qs('#btnExportCorr').addEventListener('click', ()=>{ const tex=buildTex(true); download('correction.tex', tex, 'application/x-tex;charset=utf-8'); });

qs('#btnGenAI').addEventListener('click', generateWithAI);

window.addEventListener('resize', ()=>{ try{ const W=parseInt(val('#gridW','32'),10)||32; const H=parseInt(val('#gridH','32'),10)||32; const cvs=qs('#cnv'); if(cvs) scaleCanvasCSS(cvs,W,H); sizeLetterCells(W); }catch(\_e){} });

}

// --- mini self-tests pour éviter les régressions ---

function selfTests(){

try{

if(typeof scaleCanvasCSS!=='function' || typeof process!=='function') throw new Error('APIs manquantes');

const cvs=document.createElement('canvas'); document.body.appendChild(cvs); const s=scaleCanvasCSS(cvs,8,8); if(!(s>0)) throw new Error('scale invalide'); cvs.remove();

// Test latexEscape conserve les maths

const tIn = 'a $\\frac{1}{4}$ b';

const tOut = latexEscape(tIn);

if(!/\$\\frac\{1\}\{4\}\$/.test(tOut)) throw new Error('latexEscape altère les maths');

// Test prompt IA

const pr = buildAIInput({subject:'Math',level:'6e',theme:'proportionnalité',count:3});

if(typeof pr !== 'string' || pr.length<10) throw new Error('buildAIInput invalide');

console.debug('[PIXEL-MATHS] Tests OK');

}catch(e){ console.warn('[PIXEL-MATHS] Test échoué:', e.message); }

}

window.addEventListener('DOMContentLoaded', ()=>{

const DEFAULT\_IMAGE\_BASE64 = "data:image/jpeg;base64,";

if (window.renderMathInElement) {

renderMathInElement(document.body, {

delimiters: [

{left: '$$', right: '$$', display: true},

{left: '$', right: '$', display: false}

],

ignoredTags: ['script', 'noscript', 'style', 'textarea', 'pre', 'code']

});

}

wire();

// Load default image

imgEl.onload = () => {

process();

const k=parseInt(val('#k','6'),10)||6;

renderQA(k);

applyAlpha();

selfTests();

};

imgEl.src = DEFAULT\_IMAGE\_BASE64;

});

})();

</script>

</body>

</html>