Proyecto: Comparador de luz — LUGAZ SL (para novatos)

He creado un paquete listo para copiar/pegar con todo lo que pediste: una página estática (HTML+JS) que hace lo siguiente **en el cliente** (sin enviar facturas a ningún servidor):

- Página inicial con el nombre **LUGAZ SL** y el eslogan: *"Compara tu mismo tu ahorro, sin llamadas comerciales"*.
- Botón **Ver comparativas**.
- Pantalla obligatoria para poner **EMAIL** antes de continuar.
- Subida de factura **2.0TD** (PDF) — el parser extrae la **potencia contratada** (hasta 2 valores) y **hasta 3 periodos de consumo**.
- Calcula y muestra comparativas con las compañías que figuran en `tariffs.json`.
- `tariffs.json` es un archivo estático en el repo; **solo tú** puedes modificarlo (edita el fichero en GitHub / en el repo donde subas la web). Esto cumple tu requisito de poder cambiar precios en el futuro.
- Opcional: encriptación **cliente** (AES-GCM usando Web Crypto). Tras procesar la factura puedes descargar el fichero cifrado `.enc` a tu ordenador; así guardas una copia cifrada si lo deseas.

> Nota: La aplicación funciona completamente en el navegador (cliente). No hay subida obligatoria a ningún servidor, por privacidad y simplicidad. Si en el futuro quieres que las facturas cifradas se guarden en la nube, te doy el endpoint y el código para subir a Supabase/Vercel/etc.

Contenido del paquete (archivos)

A continuación están los archivos necesarios. Cópialos tal cual dentro de un repositorio y sigue el README más abajo.

1) `index.html`

```
<!doctype html>
<html lang="es">
<head>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width,initial-scale=1" />
  <title>LUGAZ SL — Comparador de luz</title>
  <style>
    /* Estilos sencillos para novatos */
    body{font-family:Inter,system-ui,Arial;display:flex;align-
items:center;justify-content:center;min-height:100vh;margin:
0;background:#f7fafc}
    .card{background:white;padding:24px;border-radius:12px;box-shadow:0 6px
24px rgba(15,23,42,0.08);width:480px;max-width:94%}
    h1{margin:0 0 6px;font-size:22px}
    p.lead{color:#374151;margin-top:0}
```

```
    button{background:#0ea5a4;color:white;border:0;padding:10px 14px;border-
radius:8px;cursor:pointer}
    input[type=email], input[type=file], .hidden{width:100%;padding:
8px;margin:8px 0;border-radius:6px;border:1px solid #e5e7eb}
    .center{display:flex;gap:8px;align-items:center}
    pre{background:#f3f4f6;padding:12px;border-radius:8px;overflow:auto}
    table{width:100%;border-collapse:collapse}
    th,td{padding:8px;border-bottom:1px solid #e6e7eb;text-align:left}
    .small{font-size:13px;color:#6b7280}
  </style>
</head>
<body>
  <div class="card" id="app">
    <!-- Pantalla inicial -->
    <div id="home">
      <h1>LUGAZ SL</h1>
      <p class="lead">Compara tu mismo tu ahorro, sin llamadas comerciales</
p>
      <div style="margin-top:14px">
        <button id="btnStart">Ver comparativas</button>
      </div>
    </div>

    <!-- Paso email -->
    <div id="emailScreen" style="display:none">
      <h2>Introduce tu email</h2>
      <p class="small">Necesitamos tu email para mostrar comparativas. No
enviamos spam.</p>
      <input id="emailInput" type="email" placeholder="tu@ejemplo.com" />
      <div style="margin-top:10px"><button id="btnEmailContinue">Continuar</
button></div>
    </div>

    <!-- Upload -->
    <div id="uploadScreen" style="display:none">
      <h2>Sube tu factura 2.0TD</h2>
      <p class="small">Soportado: PDF versión 2.0TD. Extraemos hasta 2
potencias contratadas y hasta 3 periodos de consumo.</p>
      <input id="fileInput" type="file" accept="application/pdf" />
      <div style="margin-top:10px" class="center">
        <button id="btnProcess">Procesar factura</button>
        <button id="btnDownloadEncrypted"
style="display:none;background:#2563eb">Descargar (cifrado)</button>
      </div>
      <div id="parseErrors" style="margin-top:10px;color:#b91c1c"></div>
    </div>

    <!-- Resultados -->
    <div id="resultsScreen" style="display:none">
      <h2>Comparativa</h2>
      <div id="parsedInfo"></div>
```

```
      <div id="comparatives"></div>
    </div>

  </div>

  <script type="module" src="app.js"></script>
</body>
</html>
```

---

2) `app.js`

```
// app.js — lógica principal (cliente)

// Rutas relativas (tariffs.json debe estar en el mismo directorio)
const TARIFFS_URL = './tariffs.json';

// Elementos UI
const home = document.getElementById('home');
const btnStart = document.getElementById('btnStart');
const emailScreen = document.getElementById('emailScreen');
const btnEmailContinue = document.getElementById('btnEmailContinue');
const emailInput = document.getElementById('emailInput');
const uploadScreen = document.getElementById('uploadScreen');
const fileInput = document.getElementById('fileInput');
const btnProcess = document.getElementById('btnProcess');
const parseErrors = document.getElementById('parseErrors');
const resultsScreen = document.getElementById('resultsScreen');
const parsedInfo = document.getElementById('parsedInfo');
const comparatives = document.getElementById('comparatives');
const btnDownloadEncrypted = document.getElementById('btnDownloadEncrypted');

let detected = null;
let fileBuffer = null;
let tariffs = [];

btnStart.onclick = () => { home.style.display='none';
emailScreen.style.display='block'; }

btnEmailContinue.onclick = () => {
  const email = emailInput.value.trim();
  if (!email || !email.includes('@')) { alert('Introduce un email válido');
return; }
  emailScreen.style.display='none';
  uploadScreen.style.display='block';
  // cargar tarifas

fetch(TARIFFS_URL).then(r=>r.json()).then(j=>tariffs=j).catch(()=>{tariffs=[]});
}
```

```
btnProcess.onclick = async () => {
  parseErrors.innerText='';
  const f = fileInput.files[0];
  if (!f) { parseErrors.innerText='Selecciona un fichero PDF.'; return; }
  // leer PDF en ArrayBuffer
  fileBuffer = await f.arrayBuffer();
  // extraer texto con pdf.js (uso dinámico desde CDN) — implementado pequeño
loader
  try{
    const text = await extractTextFromPDF(fileBuffer);
    // parseo para 2.0TD limitada
    detected = parse2_0TD(text);
    if (!detected) { parseErrors.innerText = 'No se pudo detectar un formato
2.0TD. Revisa la factura.'; return; }

    uploadScreen.style.display='none';
    resultsScreen.style.display='block';
    renderParsed(detected);
    renderComparatives(detected, tariffs);

    // Habilitar botón descarga cifrada
    btnDownloadEncrypted.style.display='inline-block';
  } catch(err){
    console.error(err);
    parseErrors.innerText='Error procesando PDF.';
  }
}

// Descargar fichero cifrado (cliente) — pide contraseña y descarga .enc
btnDownloadEncrypted.onclick = async () => {
  if (!fileBuffer) return;
  const pw = prompt('Introduce una contraseña para cifrar la factura.
Guárdala bien.');
  if (!pw) return alert('Contraseña vacía — operación cancelada');
  const base64 = await encryptFile(fileBuffer, pw);
  const blob = base64ToBlob(base64);
  const url = URL.createObjectURL(blob);
  const a = document.createElement('a');
  a.href = url; a.download = 'factura.enc';
  document.body.appendChild(a); a.click(); a.remove();
  URL.revokeObjectURL(url);
}

function renderParsed(d){
  parsedInfo.innerHTML = `\n    <div class="small">Datos extraídos:</
div>\n    <pre>${JSON.stringify(d, null, 2)}</pre>\n  `;
}

function renderComparatives(d, tariffsList){
  if (!tariffsList || !tariffsList.length){ comparatives.innerHTML='<div
```

```
class="small">No hay tarifas configuradas. Edita tariffs.json en el
repositorio.</div>'; return; }
  // cálculo simple: para cada tarifa calcular coste = sum(period_kwh *
precio_kwh_period) + potencia diaria * dias
  // asumimos tarifas con precio_kwh por periodo index (0..2) y
terminoPotencia €/kW/día
  const rows = tariffsList.map(t => {
    let energia = 0;
    for(let i=0;i<d.periods.length;i++){
      const kwh = d.periods[i] || 0;
      const price = (t.precio_kwh_per_period && t.precio_kwh_per_period[i]!
=null) ? t.precio_kwh_per_period[i] : t.precio_kwh;
      energia += kwh * price;
    }
    // potencia: si hay 2 potencias sumamos ambos * terminoPotencia
    const dias = d.days || 30;
    const potenciaKW = (d.potencias && d.potencias.length>0) ?
d.potencias.reduce((a,b)=>a+b,0) : 0;
    const termino = (t.termino_potencia_eur_per_kw_day || 0) * potenciaKW *
dias;
    const total = Math.round((energia + termino + (t.otros_costes||0))*100)/
100;
    return { provider: t.name, total, energia: Math.round(energia*100)/100,
termino };
  }).sort((a,b)=>a.total-b.total);

  let html = '<table><thead><tr><th>Proveedor</th><th>Coste estimado (€)</
th><th>Detalle</th></tr></thead><tbody>';
  for(const r of rows){
    html += `<tr><td>${r.provider}</td><td><strong>${r.total}</strong></
td><td class="small">Energía ${r.energia}€ · Potencia $
{Math.round(r.termino*100)/100}€</td></tr>`;
  }
  html += '</tbody></table>';
  comparatives.innerHTML = html;
}

// ======================================================
// ---------- Parser simple para 2.0TD (limitado) -----
// ======================================================

function parse2_0TD(text){
  // 2.0TD typical PDF has lines like: "Energía consumida periodo 1: 123 kWh"
or "Consumo periodo 1: 123 kWh"
  // y potencias como "Potencia contratada: 4.6 kW". Cada factura varía
mucho; esto es un parser muy simple
  const lower = text.toLowerCase();

  // buscar potencias (hasta 2)
  const potenciaRegex = /potencia contratada[:\s]*([0-9]+[\.,]?[0-9]*)\s*k?
w?/gi;
```

```
  const potencias = [];
  let m;
  while((m = potenciaRegex.exec(lower)) && potencias.length<2){
potencias.push(parseFloat(m[1].replace(',', '.'))); }

  // buscar consumos por periodo (hasta 3)
  const periodRegex = /periodo\s*(?:1|i)[:\s]*([0-9]+[\.,]?[0-9]*)\s*kwh|
consumo periodo\s*1[:\s]*([0-9]+[\.,]?[0-9]*)\s*kwh/gi;
  // en vez de patrones excesivos, recolectaremos coincidencias generales de
"123 kWh" próximas a palabras "periodo" o "consumo"
  const genericKwhRegex = /([0-9]+[\.,]?[0-9]*)\s*kwh/g;
  const periods = [];
  // estrategia: si aparece la palabra "periodo 1/2/3" intentar capturar el
número después
  for(let i=1;i<=3;i++){
    const r = new RegExp('periodo\\s*' + i + '[^0-9]{0,30}([0-9]+[\\.,]?
[0-9]*)\\s*kwh','i');
    const mm = lower.match(r);
    if(mm) periods.push(parseFloat(mm[1].replace(',', '.')));
  }
  // si no se encontraron, intentar capturar hasta 3 KWh frecuentes
  if(periods.length===0){
    let gm;
    while((gm = genericKwhRegex.exec(lower)) && periods.length<3){
periods.push(parseFloat(gm[1].replace(',', '.'))); }
  }

  // buscar periodo de días (fecha inicio/fin) para calcular días
  const dateRange = lower.match(/periodo\s*([0-9]{1,2}\/\d{1,2}\/\d{2,4})\s*-
\s*([0-9]{1,2}\/\d{1,2}\/\d{2,4})/i);
  let days = 30;
  if(dateRange){
    try{
      const d1 = parseEuropeanDate(dateRange[1]);
      const d2 = parseEuropeanDate(dateRange[2]);
      const diff = Math.round((d2-d1)/(1000*60*60*24));
      if(diff>0) days = diff;
    }catch(e){}
  }

  if (periods.length===0 && potencias.length===0) return null;
  return { potencias, periods, days };
}

function parseEuropeanDate(s){
  // acepta dd/mm/yyyy o d/m/yy
  const parts = s.split('/').map(x=>parseInt(x,10));
  let [d,m,y]=parts; if(y<100) y+=2000; return new Date(y,m-1,d);
}

// ======================================================
```

```
// === Extracción de texto desde PDF usando pdf.js CDN ===
// =======================================================
async function extractTextFromPDF(arrayBuffer){
  // Carga dinámica del pdf.js (versión ligera CDN)
  const scriptUrl = 'https://cdn.jsdelivr.net/npm/pdfjs-dist@2.16.105/build/
pdf.min.js';
  if (!window.pdfjsLib){
    await loadScript(scriptUrl);
  }
  const pdfjsLib = window.pdfjsLib;
  const loadingTask = pdfjsLib.getDocument({data: arrayBuffer});
  const pdf = await loadingTask.promise;
  let fullText = '';
  for(let i=1;i<=pdf.numPages;i++){
    const page = await pdf.getPage(i);
    const content = await page.getTextContent();
    const pageText = content.items.map(it=>it.str).join(' ');
    fullText += '\n' + pageText;
  }
  return fullText;
}

function loadScript(url){
  return new Promise((resolve,reject)=>{
    const s = document.createElement('script'); s.src = url; s.onload =
()=>{resolve();}; s.onerror = reject; document.head.appendChild(s);
  });
}

// =======================================================
// === Cifrado cliente: PBKDF2 -> AES-GCM (Web Crypto) ===
// =======================================================
async function deriveKeyFromPassword(password, salt){
  const enc = new TextEncoder();
  const keyMaterial = await crypto.subtle.importKey('raw',
enc.encode(password), {name:'PBKDF2'}, false, ['deriveKey']);
  return crypto.subtle.deriveKey({name:'PBKDF2', salt, iterations:200000,
hash:'SHA-256'}, keyMaterial, {name:'AES-GCM', length:256}, true,
['encrypt','decrypt']);
}

async function encryptFile(arrayBuffer, password){
  const salt = crypto.getRandomValues(new Uint8Array(16));
  const iv = crypto.getRandomValues(new Uint8Array(12));
  const key = await deriveKeyFromPassword(password, salt);
  const cipher = await crypto.subtle.encrypt({name:'AES-GCM', iv}, key,
arrayBuffer);
  // empaquetar: salt(16) + iv(12) + cipher
  const saltA = new Uint8Array(salt);
  const ivA = new Uint8Array(iv);
  const cipherA = new Uint8Array(cipher);
```

```
  const out = new Uint8Array(16 + 12 + cipherA.length);
  out.set(saltA,0); out.set(ivA,16); out.set(cipherA,28);
  // devolver base64
  return arrayBufferToBase64(out.buffer);
}

function arrayBufferToBase64(buffer){
  let binary=''; const bytes=new Uint8Array(buffer); const
len=bytes.byteLength; for(let i=0;i<len;i++)
binary+=String.fromCharCode(bytes[i]); return btoa(binary);
}

function base64ToBlob(b64){
  const binary = atob(b64);
  const len = binary.length; const bytes = new Uint8Array(len);
  for(let i=0;i<len;i++) bytes[i]=binary.charCodeAt(i);
  return new Blob([bytes], {type:'application/octet-stream'});
}

// =====================================================
// === util pequeño ===
function round2(x){ return Math.round(x*100)/100 }
```

---

3) `tariffs.json` (ejemplo editable)

```
[
  {
    "name": "Compañía A",
    "precio_kwh": 0.18,
    "precio_kwh_per_period": [0.20, 0.15, 0.10],
    "termino_potencia_eur_per_kw_day": 0.12,
    "otros_costes": 0.0
  },
  {
    "name": "Compañía B",
    "precio_kwh": 0.16,
    "precio_kwh_per_period": [0.18, 0.14, 0.09],
    "termino_potencia_eur_per_kw_day": 0.10,
    "otros_costes": 0.0
  },
  {
    "name": "Compañía C",
    "precio_kwh": 0.195,
    "precio_kwh_per_period": [0.22, 0.16, 0.12],
    "termino_potencia_eur_per_kw_day": 0.08,
    "otros_costes": 0.0
  }
]
```

Cómo editar los precios: simplemente abre `tariffs.json` en el repositorio y cambia los valores. Si despliegas con GitHub Pages o Netlify, basta con hacer commit y deploy. Como propietario del repo nadie podrá editarlo a menos que tenga acceso a tu cuenta GitHub (cumple el requisito "solo por mí").

---

4) `README.md` — paso a paso para novatos

```
# LUGAZ SL — Deploy rápido (novatos)

Sigue estos pasos para tener la web en internet usando **GitHub Pages** (gratuito) o **Netlify** / **Vercel**. Aquí explico GitHub Pages por ser el más sencillo.

## 1) Crea un repositorio en GitHub
1. Entra a github.com y crea una cuenta (si no tienes).
2. Nuevo repositorio -> nombre: `lugaz-comparador` (por ejemplo). Marca public/private según prefieras.
3. Clona el repo en tu ordenador o sube los archivos usando "Add file" -> "Upload files" en la web.

## 2) Añade los archivos
Sube `index.html`, `app.js`, `tariffs.json` y `README.md` al repo y haz commit.

## 3) Activa GitHub Pages
1. Ve al repo -> Settings -> Pages.
2. En "Source" elige la rama `main` y carpeta `/ (root)`. Guarda.
3. GitHub te dará una URL tipo `https://<tu-usuario>.github.io/<repo>/` en unos minutos.

## 4) Probar la web
Abre la URL y prueba el flujo: Ver comparativas → introducir email → subir PDF 2.0TD.

## 5) Cambiar tarifas (solo tú)
Para actualizar los precios edita `tariffs.json` en el repo (Edit file -> Commit). Después de un deploy automático la nueva tarifa estará activa.

## 6) Opcional: desplegar en Netlify o Vercel
Si prefieres Netlify/Vercel (ambos ofrecen HTTPS y un dominio más bonito), conecta tu repo y sigue su asistente de deploy.

## Notas y siguientes pasos
- Este proyecto procesa todo en el navegador: es privado y no sube tus facturas a ningún servidor por defecto.
- Si en el futuro quieres guardar las facturas cifradas en la nube o crear un panel de administración para editar tarifas desde la web, te preparo la versión con Supabase/Serverless.
```

¿Qué he colocado en el canvas?

He incluido los archivos principales (`index.html`, `app.js`, `tariffs.json`, `README.md`) en este documento para que puedas copiarlos y pegarlos en tu repo.

Si quieres, puedo ahora: - Generar automáticamente un repositorio público en GitHub con estos archivos (necesitaré que me autorices a usar tu cuenta — si no quieres eso, te doy los pasos manuales). **No puedo crear repos repos en tu cuenta sin permiso.** - Crear una versión que guarde las facturas cifradas en Supabase y una función serverless para descargar/gestionar (si quieres almacenamiento en la nube).

Dime qué prefieres: **(A)** te creo el repo con los archivos (te doy instrucciones para añadirlo a tu cuenta), o **(B)** prefieres que te explique y te guíe para que lo subas tú paso a paso.