Despliegue en cPanel (hosting compartido)

Este proyecto es una app Node.js (Express) que sirve una SPA de React ya construida. En cPanel podemos usar Application Manager (Node.js) o Passenger (Phusion) para ejecutarla. A continuación, pasos prácticos:

1) Preparar build local y subir archivos

- En tu máquina local:
  - Configura tu `.env` con valores reales (SMTP, JWT_SECRET, ADMIN_PASSWORD opcional, FRONTEND_URL a tu dominio público, p. ej. `https://cotizador.tudominio.cl`).
  - Construye el frontend:
    - `npm run build`
  - Verifica que existen:
    - `frontend/dist/` con los archivos estáticos
    - carpeta `backend/` con el servidor Express
    - carpeta `outputs/` (cPanel debe tener permisos de escritura en esta carpeta para PDFs/JSON/QR/logos)
- Sube al hosting (por FTP o gestor de archivos cPanel) el contenido del repositorio, incluyendo `frontend/dist` y excluyendo `node_modules` (se instalarán en el servidor).

2) Crear aplicación Node.js en cPanel

- En cPanel, abre "Setup Node.js App" (Application Manager).
- Elige:
  - Application root: la carpeta donde subiste el proyecto (por ejemplo `/home/usuario/cotizador`).
  - Application URL: tu subdominio o ruta pública (p. ej. `https://cotizador.tudominio.cl`).
  - Application startup file: `backend/server.js` (usa el `server.js` que arranca el `app`).
  - Node.js version: 18 o 20 (recomendado 20).
- Crea la app.

3) Variables de entorno

En la pantalla de la app Node, agrega variables:

- `PORT` (cPanel asigna internamente el puerto; normalmente Passenger ignora PORT, pero no estorba)
- `JWT_SECRET` (seguro y único)
- `ADMIN_PASSWORD` (si usarás login admin)
- `SMTP_HOST`, `SMTP_PORT`, `SMTP_USER`, `SMTP_PASS` (si enviarás correos)
- `FRONTEND_URL` = `https://cotizador.tudominio.cl` (tu URL pública; usada para el QR y enlaces)
- Opcional: `OUTPUTS_DIR` si deseas ubicar `outputs/` fuera del proyecto o con permisos especiales.

Guarda los cambios.

4) Instalar dependencias en el servidor

- Desde la tarjeta de la app en cPanel, abre el Terminal de la app o usa SSH.
- Ejecuta en el root de la app:
  - `npm ci` (o `npm install` si no usas lockfile compatible)
  - `cd frontend && npm ci && cd ..`
  - (El build de frontend ya lo subiste, pero si prefieres construir en servidor: `npm run build`)

5) Estructura estática y SPA

- El backend sirve estáticos de `frontend/dist` si existe, y hace fallback SPA (todas las rutas no `/api` ni `/outputs` envían `index.html`).
- Archivos generados (PDFs, QR, JSON) se guardan en `outputs/` y se sirven de forma de solo lectura vía `/outputs/*`.

6) Comprobación

- Reinicia la app desde cPanel y abre la URL pública.
- Verifica:
  - `GET /` responde JSON ok (salud del backend).
  - Navega a la raíz y carga la SPA.
  - Crear una cotización genera PDF en `outputs/pdfs/` y QR en `outputs/`.

7) Permisos y seguridad

- Asegúrate de que `outputs/` sea escribible por el proceso de Node. Si no, define `OUTPUTS_DIR` a otro path con permisos.
- No subas `.env` al repo público. Configura variables en cPanel.
- Rota `JWT_SECRET` y credenciales SMTP si hiciste pruebas con valores reales.

8) Logs y troubleshooting

- La app usa `morgan` para logs HTTP. Passenger/cPanel registra salida del proceso; revisa error logs del dominio/aplicación.
- Si el frontend no carga, confirma que `frontend/dist` existe en el servidor y que el fallback está activo.

SMTP (Nodemailer) problemas comunes

- `ETIMEDOUT` / `Greeting never received`:
  - Verifica `SMTP_HOST` y `SMTP_PORT` correctos (en cPanel suele ser 465 para SSL/TLS, 587 para STARTTLS).
  - Ajusta `SMTP_SECURE`: `true` para 465, `false` para 587.
  - Si el servidor requiere certificados válidos y hay problemas de CA, puedes probar `SMTP_TLS_REJECT_UNAUTH=false` temporalmente.
  - Asegura que el hosting permita conexiones salientes al puerto SMTP (algunos compartidos bloquean 25/465/587).
  - Usa el script `node backend/scripts/verify_smtp.js` en el servidor para probar la conexión/handshake.

Para probar un envío real (texto simple):

  - Define `SMTP_TEST_TO` (o usa `SMTP_USER`) y ejecuta:
  - `npm run send:test-email`

- Autenticación fallida:

  - Revisa `SMTP_USER`/`SMTP_PASS` y si el proveedor requiere `SMTP_FROM` específico.
  - Algunas cuentas requieren contraseñas de aplicación (p. ej., Gmail/Outlook) o habilitar SMTP en cPanel.

9) Opcional: Subdominio dedicado

- En cPanel, crea un subdominio (p. ej. `cotizador.tudominio.cl`) apuntando a la carpeta del proyecto.
- Configura SSL (AutoSSL o Let's Encrypt) y actualiza `FRONTEND_URL`.

Con esto deberías tener el despliegue funcionando en cPanel.