<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8" />

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

<title>Credit Stress Simulator</title>

<!-- Tailwind CSS CDN (no build step) -->

<script src="https://cdn.tailwindcss.com"></script>

<!-- Chart.js for simple time‑series charts -->

<script src="https://cdn.jsdelivr.net/npm/chart.js@4.4.1/dist/chart.umd.min.js"></script>

<meta name="description" content="Interactive Credit Stress Simulator for teaching / demos. Single‑file app for GitHub Pages." />

<style>

/\* Subtle scrollbars + number input alignment \*/

input[type="number"] { text-align: right; }

.kpi { min-width: 12rem; }

</style>

</head>

<body class="bg-slate-50 text-slate-900">

<header class="sticky top-0 z-30 bg-white/80 backdrop-blur border-b border-slate-200">

<div class="max-w-7xl mx-auto px-4 py-4 flex items-center justify-between">

<h1 class="text-2xl md:text-3xl font-bold tracking-tight">Credit Stress Simulator</h1>

<div class="flex items-center gap-2 text-sm">

<button id="btnSave" class="px-3 py-2 rounded-xl bg-slate-900 text-white hover:bg-slate-800">Save run</button>

<button id="btnLoad" class="px-3 py-2 rounded-xl bg-white border border-slate-300 hover:bg-slate-100">Load</button>

<button id="btnReset" class="px-3 py-2 rounded-xl bg-white border border-slate-300 hover:bg-slate-100">Reset</button>

<a href="#about" class="px-3 py-2 rounded-xl bg-white border border-slate-300 hover:bg-slate-100">About</a>

</div>

</div>

</header>

<main class="max-w-7xl mx-auto px-4 py-6">

<!-- Top KPIs -->

<section class="grid sm:grid-cols-2 lg:grid-cols-6 gap-3 mb-6">

<div class="kpi bg-white rounded-2xl p-4 shadow-sm border border-slate-200">

<div class="text-xs uppercase tracking-wide text-slate-500">Quarters</div>

<div class="mt-1 text-2xl font-semibold"><span id="qNow">0</span> / <span id="qMax">12</span></div>

</div>

<div class="kpi bg-white rounded-2xl p-4 shadow-sm border border-slate-200">

<div class="text-xs uppercase tracking-wide text-slate-500">PD (12Q)</div>

<div class="mt-1 text-2xl font-semibold"><span id="pdNow">0.8%</span></div>

</div>

<div class="kpi bg-white rounded-2xl p-4 shadow-sm border border-slate-200">

<div class="text-xs uppercase tracking-wide text-slate-500">DSCR</div>

<div class="mt-1 text-2xl font-semibold"><span id="dscrNow">2.6x</span></div>

</div>

<div class="kpi bg-white rounded-2xl p-4 shadow-sm border border-slate-200">

<div class="text-xs uppercase tracking-wide text-slate-500">Interest Coverage</div>

<div class="mt-1 text-2xl font-semibold"><span id="icrNow">4.1x</span></div>

</div>

<div class="kpi bg-white rounded-2xl p-4 shadow-sm border border-slate-200">

<div class="text-xs uppercase tracking-wide text-slate-500">Net Leverage</div>

<div class="mt-1 text-2xl font-semibold"><span id="levNow">2.8x</span></div>

</div>

<div class="kpi bg-white rounded-2xl p-4 shadow-sm border border-slate-200">

<div class="text-xs uppercase tracking-wide text-slate-500">Rating (heuristic)</div>

<div class="mt-1 text-2xl font-semibold"><span id="ratingNow">BBB–</span></div>

</div>

</section>

<!-- Layout: Controls left, Charts right -->

<section class="grid lg:grid-cols-12 gap-6">

<div class="lg:col-span-5 space-y-6">

<!-- Company setup -->

<div class="bg-white rounded-2xl border border-slate-200 shadow-sm">

<div class=\"p-4 border-b border-slate-200 flex items-center justify-between\">

<h2 class=\"text-lg font-semibold\">Company Setup<\/h2>

<div class=\"flex items-center gap-2\">

<label class=\"text-xs text-slate-600 hidden sm:block\">Sector<\/label>

<select id=\"sectorSel\" class=\"text-xs px-2 py-1 rounded-lg border border-slate-200 bg-white\">

<option value=\"generic\">Generic Industrial<\/option>

<option value=\"packaging\">Packaging & Labeling<\/option>

<option value=\"ofs\">Oilfield Services<\/option>

<option value=\"consumer\">Consumer Staples<\/option>

<option value=\"re\">Real Estate (REIT-like)<\/option>

<\/select>

<button id=\"btnDefaults\" class=\"text-xs px-2 py-1 rounded-lg bg-slate-100 border border-slate-200 hover:bg-slate-200\">Default<\/button>

<\/div>

<\/div>

<div class="p-4 grid grid-cols-2 gap-3 text-sm">

<label class="col-span-2">Starting Revenue (annual, $m)

<input id="rev0" type="number" value="2000" step="10" class="mt-1 w-full px-3 py-2 border rounded-xl" />

</label>

<label>EBITDA Margin (%)

<input id="ebitdam" type="number" value="22" step="0.5" class="mt-1 w-full px-3 py-2 border rounded-xl" />

</label>

<label>Capex (% of revenue)

<input id="capexPct" type="number" value="4" step="0.5" class="mt-1 w-full px-3 py-2 border rounded-xl" />

</label>

<label>ΔNWC (% of Δrevenue)

<input id="nwcPct" type="number" value="8" step="0.5" class="mt-1 w-full px-3 py-2 border rounded-xl" />

</label>

<label>Cash ($m)

<input id="cash0" type="number" value="300" step="10" class="mt-1 w-full px-3 py-2 border rounded-xl" />

</label>

<label>Total Debt ($m)

<input id="debt0" type="number" value="2500" step="10" class="mt-1 w-full px-3 py-2 border rounded-xl" />

</label>

<label>Fixed‑rate debt (%)

<input id="fixedPct" type="number" value="65" step="1" class="mt-1 w-full px-3 py-2 border rounded-xl" />

</label>

<label>Fixed coupon (%)

<input id="fixedCpn" type="number" value="5.0" step="0.1" class="mt-1 w-full px-3 py-2 border rounded-xl" />

</label>

<label>Float spread (bps over base)

<input id="floatSpr" type="number" value="250" step="10" class="mt-1 w-full px-3 py-2 border rounded-xl" />

</label>

<label>Tax rate (%)

<input id="taxRate" type="number" value="24" step="1" class="mt-1 w-full px-3 py-2 border rounded-xl" />

</label>

<label>Covenant: Max Net Leverage (x)

<input id="covLev" type="number" value="3.5" step="0.1" class="mt-1 w-full px-3 py-2 border rounded-xl" />

</label>

<label>Covenant: Min ICR (x)

<input id="covICR" type="number" value="3.0" step="0.1" class="mt-1 w-full px-3 py-2 border rounded-xl" />

</label>

<label class="col-span-2">Maturity wall (% of total debt due each year)

<div class="grid grid-cols-4 gap-2 mt-1">

<input id="mat1" type="number" value="10" step="1" class="px-3 py-2 border rounded-xl" title="Due within Y1"/>

<input id="mat2" type="number" value="20" step="1" class="px-3 py-2 border rounded-xl" title="Due in Y2"/>

<input id="mat3" type="number" value="25" step="1" class="px-3 py-2 border rounded-xl" title="Due in Y3"/>

<input id="mat4" type="number" value="45" step="1" class="px-3 py-2 border rounded-xl" title="Due Y4+ (refi laddered)"/>

</div>

</label>

</div>

</div>

<!-- Macro & Stress -->

<div class="bg-white rounded-2xl border border-slate-200 shadow-sm">

<div class="p-4 border-b border-slate-200 flex items-center justify-between">

<h2 class="text-lg font-semibold">Macro & Stress</h2>

<div class="text-xs text-slate-500">Quarterly model (12Q horizon)</div>

</div>

<div class=\"p-4 grid grid-cols-2 gap-3 text-sm\">

<label>Base GDP growth (q/q, %)

<input id=\"gdpBase\" type=\"number\" value=\"0.6\" step=\"0.1\" class=\"mt-1 w-full px-3 py-2 border rounded-xl\" />

<\/label>

<label>Inflation (q/q, %)

<input id=\"infl\" type=\"number\" value=\"0.6\" step=\"0.1\" class=\"mt-1 w-full px-3 py-2 border rounded-xl\" />

<\/label>

<label>Policy/base rate (%, annual)

<input id=\"baseRate\" type=\"number\" value=\"4.5\" step=\"0.1\" class=\"mt-1 w-full px-3 py-2 border rounded-xl\" />

<\/label>

<label>Commodity shock (margin bps)

<input id=\"commShock\" type=\"number\" value=\"0\" step=\"25\" class=\"mt-1 w-full px-3 py-2 border rounded-xl\" />

<\/label>

<div class=\"col-span-2 flex flex-wrap gap-2 mt-1\">

<button data-preset=\"soft\" class=\"preset px-3 py-2 rounded-xl bg-slate-100 border border-slate-200 hover:bg-slate-200\">Soft patch<\/button>

<button data-preset=\"recession\" class=\"preset px-3 py-2 rounded-xl bg-amber-100 border border-amber-300 hover:bg-amber-200\">Mild recession<\/button>

<button data-preset=\"rateshock\" class=\"preset px-3 py-2 rounded-xl bg-rose-100 border border-rose-300 hover:bg-rose-200\">Rate shock<\/button>

<button data-preset=\"supply\" class=\"preset px-3 py-2 rounded-xl bg-emerald-100 border border-emerald-300 hover:bg-emerald-200\">Supply shock<\/button>

<button data-preset=\"custom\" class=\"preset px-3 py-2 rounded-xl bg-indigo-100 border border-indigo-300 hover:bg-indigo-200\">Custom<\/button>

<span class=\"text-xs text-slate-500 ml-auto\">Sector cycles auto-adjust with preset<\/span>

<\/div>

<\/div>

</div>

<!-- Controls -->

<div class="bg-white rounded-2xl border border-slate-200 shadow-sm">

<div class="p-4 border-b border-slate-200 flex items-center justify-between">

<h2 class="text-lg font-semibold">Controls</h2>

<div class="text-xs text-slate-500">Policy & management levers</div>

</div>

<div class="p-4 grid grid-cols-2 gap-3 text-sm">

<label>Pricing power (bps to margin)

<input id="pricingBps" type="number" value="25" step="25" class="mt-1 w-full px-3 py-2 border rounded-xl" />

</label>

<label>Cost takeout (bps to margin)

<input id="costOutBps" type="number" value="50" step="25" class="mt-1 w-full px-3 py-2 border rounded-xl" />

</label>

<label>Capex flex (% of rev)

<input id="capexFlex" type="number" value="-1" step="0.5" class="mt-1 w-full px-3 py-2 border rounded-xl" />

</label>

<label>Buyback ($m, this qtr)

<input id="buyback" type="number" value="0" step="10" class="mt-1 w-full px-3 py-2 border rounded-xl" />

</label>

<label>Dividend ($m, this qtr)

<input id="dividend" type="number" value="30" step="5" class="mt-1 w-full px-3 py-2 border rounded-xl" />

</label>

<label>Refi spread (bps over base)

<input id="refiSpread" type="number" value="300" step="25" class="mt-1 w-full px-3 py-2 border rounded-xl" />

</label>

<label>Revolver limit ($m)

<input id="revolverLim" type="number" value="500" step="10" class="mt-1 w-full px-3 py-2 border rounded-xl" />

</label>

<label>Amortization (% qtr on total debt)

<input id="amortPct" type="number" value="1.0" step="0.1" class="mt-1 w-full px-3 py-2 border rounded-xl" />

</label>

</div>

<div class="p-4 flex flex-wrap gap-2">

<button id="btnStep" class="px-4 py-2 rounded-xl bg-slate-900 text-white hover:bg-slate-800">Advance 1 Quarter</button>

<button id="btnAuto" class="px-4 py-2 rounded-xl bg-white border border-slate-300 hover:bg-slate-100">Autopilot (12Q)</button>

<button id="btnUndo" class="px-4 py-2 rounded-xl bg-white border border-slate-300 hover:bg-slate-100">Undo</button>

</div>

</div>

<!-- Alerts -->

<div id="alerts" class="space-y-2"></div>

</div>

<!-- Charts & Table -->

<div class="lg:col-span-7 space-y-6">

<div class="bg-white rounded-2xl border border-slate-200 shadow-sm p-4">

<h2 class="text-lg font-semibold mb-3">Key Metrics Over Time</h2>

<div class="grid md:grid-cols-2 gap-4">

<canvas id="chartRevenue" height="160"></canvas>

<canvas id="chartLeverage" height="160"></canvas>

<canvas id="chartCashDebt" height="160"></canvas>

<canvas id="chartCoverage" height="160"></canvas>

</div>

</div>

<div class="bg-white rounded-2xl border border-slate-200 shadow-sm p-4 overflow-auto">

<h2 class="text-lg font-semibold mb-3">Quarterly Detail</h2>

<table id="tbl" class="min-w-full text-sm">

<thead class="text-left text-slate-600">

<tr>

<th class="pr-4 py-2">Q</th>

<th class="pr-4 py-2">Revenue</th>

<th class="pr-4 py-2">EBITDA</th>

<th class="pr-4 py-2">Int Exp</th>

<th class="pr-4 py-2">FCF</th>

<th class="pr-4 py-2">Cash</th>

<th class="pr-4 py-2">Debt</th>

<th class="pr-4 py-2">Net Lev</th>

<th class="pr-4 py-2">ICR</th>

<th class="pr-4 py-2">DSCR</th>

<th class="pr-4 py-2">Flags</th>

</tr>

</thead>

<tbody id="tblBody" class="font-mono"></tbody>

</table>

</div>

</div>

</section>

<!-- About -->

<section id="about" class="mt-10 bg-white rounded-2xl border border-slate-200 shadow-sm p-6">

<h2 class="text-xl font-semibold mb-2">About This Simulator</h2>

<p class="text-sm text-slate-700 leading-relaxed">

This single‑file web app models a simplified corporate credit over a 12‑quarter horizon. It is designed for classroom demos and interview prep.

The engine estimates EBITDA, interest, cash flow, leverage, and coverage using tunable assumptions (fixed vs floating mix, spreads, maturities, pricing/cost actions).

It also flags covenant breaches and infers a notional rating from leverage & coverage. Save and load runs via your browser’s localStorage.

</p>

<ul class=\"mt-3 text-sm list-disc pl-5 text-slate-700\">

<li><strong>Heuristic PD & Rating (updated):</strong> Bands calibrated to stylized S&P/Moody’s leverage/coverage grids and historical average default rates (educational approximations).<\/li>

<li><strong>Sector presets:</strong> Toggle baseline margins, capex intensity, working capital, debt mix, and cyclicality for Packaging, Oilfield Services, Consumer Staples, and REIT-like profiles.<\/li>

<li><strong>Limitations:</strong> Not a pricing tool; ignores FX, detailed working-capital seasonality, and idiosyncratic risks.<\/li>

<\/ul>

<p class="mt-3 text-xs text-slate-500">MIT License. Built for quick GitHub Pages deployment—just drop this as <code>index.html</code> into a repo and enable Pages.</p>

</section>

</main>

<footer class="max-w-7xl mx-auto px-4 py-10 text-xs text-slate-500">

© <span id="year"></span> Credit Stress Simulator · Educational use only.

</footer>

<script>

// ---------- Utilities ----------

const sectorPresets = {

generic:{ ebitdam:22, capexPct:4, nwcPct:8, fixedPct:65, fixedCpn:5.0, floatSpr:250, gdpAdj:0, margAdj:0 },

packaging:{ ebitdam:18, capexPct:6, nwcPct:10, fixedPct:70, fixedCpn:5.0, floatSpr:225, gdpAdj: -0.1, margAdj: -0.001 },

ofs:{ ebitdam:20, capexPct:5, nwcPct:12, fixedPct:55, fixedCpn:6.0, floatSpr:350, gdpAdj: 0.3, margAdj: -0.003 },

consumer:{ ebitdam:24, capexPct:3, nwcPct:6, fixedPct:75, fixedCpn:4.8, floatSpr:200, gdpAdj: -0.05, margAdj: 0.001 },

re:{ ebitdam:55, capexPct:2, nwcPct:2, fixedPct:80, fixedCpn:5.2, floatSpr:200, gdpAdj: -0.2, margAdj: 0 },

};

function applySector(key){

const s = sectorPresets[key] || sectorPresets.generic;

$('ebitdam').value = s.ebitdam;

$('capexPct').value = s.capexPct;

$('nwcPct').value = s.nwcPct;

$('fixedPct').value = s.fixedPct;

$('fixedCpn').value = s.fixedCpn;

$('floatSpr').value = s.floatSpr;

state.macro = state.macro || {};

state.macro.sector = key;

state.macro.gdpAdj = s.gdpAdj/100;

state.macro.margAdj = s.margAdj;

}

const fmt = (n) => n.toLocaleString(undefined, { maximumFractionDigits: 1 });

const $ = (id) => document.getElementById(id);

const toNum = (id) => parseFloat($(id).value || 0);

const state = {

qNow: 0,

qMax: 12,

hist: [], // keep snapshots for Undo

series: {

labels: [],

revenue: [],

ebitda: [],

interest: [],

fcf: [],

cash: [],

debt: [],

netLev: [],

icr: [],

dscr: [],

},

company: {},

macro: {},

controls: {},

flags: [],

};

function readInputs() {

state.company = {

revAnn: toNum('rev0'),

ebitdam: toNum('ebitdam'),

capexPct: toNum('capexPct'),

nwcPct: toNum('nwcPct'),

cash: toNum('cash0'),

debt: toNum('debt0'),

fixedPct: toNum('fixedPct') / 100,

fixedCpn: toNum('fixedCpn') / 100,

floatSpr: toNum('floatSpr') / 10000, // bps -> decimal

taxRate: toNum('taxRate') / 100,

covLev: toNum('covLev'),

covICR: toNum('covICR'),

mat: [toNum('mat1'), toNum('mat2'), toNum('mat3'), toNum('mat4')],

};

// Normalize maturity buckets to 100%

const s = state.company.mat.reduce((a,b)=>a+b,0);

state.company.mat = state.company.mat.map(x=>x/s);

state.macro = {

gdpBase: toNum('gdpBase')/100, // q/q

infl: toNum('infl')/100,

baseRate: toNum('baseRate')/100,

commShock: toNum('commShock')/10000, // bps

preset: currentPreset,

};

state.controls = {

pricingBps: toNum('pricingBps')/10000,

costOutBps: toNum('costOutBps')/10000,

capexFlex: toNum('capexFlex')/100,

buyback: toNum('buyback'),

dividend: toNum('dividend'),

refiSpread: toNum('refiSpread')/10000,

revolverLim: toNum('revolverLim'),

amortPct: toNum('amortPct')/100,

};

}

function resetState() {

state.qNow = 0;

state.flags = [];

for (const k in state.series) state.series[k] = [];

state.series.labels = [];

renderKPIs({pd:0.008, dscr:2.6, icr:4.1, lev:2.8, rating:'BBB–'});

$('tblBody').innerHTML='';

chartsInit();

pushSnapshot();

}

function pushSnapshot() {

state.hist.push(JSON.stringify({

qNow: state.qNow,

series: state.series,

company: state.company,

macro: state.macro,

controls: state.controls,

flags: state.flags,

}));

if (state.hist.length > 50) state.hist.shift();

}

function undo() {

if (state.hist.length < 2) return;

state.hist.pop();

const last = JSON.parse(state.hist[state.hist.length-1]);

Object.assign(state, last);

repaintAll();

}

function ratingFrom(lev, icr) {

// Stylized grid inspired by public rating practices (educational).

// Priority: both leverage and coverage must fit the band; else take the worse bucket.

const bands = [

{ r: 'A', levMax: 2.0, icrMin: 8.0 },

{ r: 'BBB+', levMax: 2.5, icrMin: 6.0 },

{ r: 'BBB', levMax: 3.0, icrMin: 4.5 },

{ r: 'BBB–', levMax: 3.5, icrMin: 3.0 },

{ r: 'BB+', levMax: 4.0, icrMin: 2.5 },

{ r: 'BB', levMax: 4.75, icrMin: 2.0 },

{ r: 'BB–', levMax: 5.5, icrMin: 1.6 },

{ r: 'B+', levMax: 6.5, icrMin: 1.3 },

{ r: 'B', levMax: 7.5, icrMin: 1.1 },

];

for (const b of bands) {

if (lev <= b.levMax && icr >= b.icrMin) return b.r;

}

return 'CCC';

}

function pdHeuristic(lev, icr, dscr, cash, debt, breach) {

// Baseline 1-yr PD by rating bucket (very rough long-run averages)

const r = ratingFrom(lev, icr);

const base1y = {

'A': 0.0005, 'BBB+': 0.001, 'BBB': 0.002, 'BBB–': 0.004,

'BB+': 0.010, 'BB': 0.020, 'BB–': 0.040, 'B+': 0.070, 'B': 0.100, 'CCC': 0.250

}[r] || 0.25;

// Convert to 3-year (12Q) cumulative assuming independence

let pd = 1 - Math.pow(1 - base1y, 3);

// Liquidity & covenant adjustments

const liqRatio = debt > 0 ? cash / debt : 1;

if (liqRatio < 0.05) pd \*= 1.4; else if (liqRatio < 0.10) pd \*= 1.2;

if (dscr < 1.0) pd \*= 1.25;

if (breach) pd \*= 1.4;

// Clamp

return Math.min(0.95, Math.max(0.002, pd));

}

function quarterStep() {

readInputs();

const { company: c, macro: m, controls: u } = state;

const q = state.qNow + 1;

// Revenue path (quarterized from annual starting point)

const revPrev = state.series.revenue.length

? state.series.revenue[state.series.revenue.length-1]

: c.revAnn/4;

// macro shock to growth

let growth = m.gdpBase + (m.gdpAdj||0); // q/q + sector tilt

switch (m.preset) {

case 'soft': growth -= 0.1/100; break; // -10bps

case 'recession': growth -= 1.2/100; break; // -120bps q/q

case 'rateshock': growth -= 0.4/100; break;

case 'supply': growth -= 0.3/100; break;

}

const rev = revPrev \* (1 + growth);

// EBITDA margin adjustments

let marg = c.ebitdam/100 + (m.margAdj||0);

marg += u.pricingBps + u.costOutBps; // management actions

marg -= m.commShock; // commodity squeeze

if (m.preset==='supply') marg -= 0.003; // -30bps

if (m.preset==='recession') marg -= 0.002; // -20bps

marg = Math.max(0.05, Math.min(0.45, marg));

const ebitda = rev \* marg;

// Capex & NWC

const capexRate = Math.max(0, (c.capexPct/100) + u.capexFlex);

const capex = rev \* capexRate;

const dRev = rev - revPrev;

const dNWC = (c.nwcPct/100) \* dRev; // sign follows dRev

// Interest expense

const fixedDebt = c.debt \* c.fixedPct;

let floatDebt = c.debt - fixedDebt;

// Maturities (annual buckets, allocate 1/4 each qtr within year)

const yearIdx = Math.min(3, Math.floor((q-1)/4));

const dueThisYear = c.debt \* c.mat[yearIdx];

const dueThisQ = dueThisYear / 4;

// Amortization

const amort = c.debt \* u.amortPct;

// Refinance due + amort at new rate (adds to fixed mix at 50/50 for simplicity)

const refiRate = m.baseRate + u.refiSpread;

const refiInt = dueThisQ \* refiRate + amort \* refiRate;

// Base interest on remaining

const baseIntFixed = Math.max(0, (fixedDebt - dueThisQ\*0.5 - amort\*0.5)) \* c.fixedCpn / 4;

const baseIntFloat = Math.max(0, (floatDebt - dueThisQ\*0.5 - amort\*0.5)) \* (m.baseRate + c.floatSpr) / 4;

const interest = baseIntFixed + baseIntFloat + refiInt/4; // quarterize refi interest

// Taxes (on EBIT ~ EBITDA for simplicity; capex/NWC ignored for tax)

const tax = Math.max(0, (ebitda - interest)) \* c.taxRate;

// Free cash flow

const fcf = ebitda - interest - tax - capex - dNWC - u.buyback - u.dividend;

// Cash update & liquidity management

let cash = c.cash + fcf;

let debt = c.debt;

let flags = [];

let revolverDraw = 0;

if (cash < 0) {

const need = -cash;

const avail = Math.max(0, u.revolverLim - (state.revolverUsed||0));

revolverDraw = Math.min(need, avail);

cash += revolverDraw;

debt += revolverDraw;

state.revolverUsed = (state.revolverUsed||0) + revolverDraw;

if (revolverDraw<need) flags.push('Liquidity shortfall');

}

// Apply amortization & refinance principal changes

debt = Math.max(0, debt - amort) + dueThisQ; // due amount refinanced (added back)

// Metrics

const netDebt = Math.max(0, debt - cash);

const netLev = ebitda>0 ? netDebt / (ebitda\*4) : 10; // annualize EBITDA for x metric

const icr = interest>0 ? (ebitda / interest) : 10;

const dscr = interest>0 ? Math.max(0.01, (ebitda - capex) / interest) : 10;

// Covenants

const breach = (netLev > c.covLev) || (icr < c.covICR);

if (breach) flags.push('Covenant breach');

// Heuristic PD & Rating

const rating = ratingFrom(netLev, icr);

const pd = pdHeuristic(netLev, icr, dscr, cash, debt, breach);

// Store path

state.series.labels.push('Q'+q);

state.series.revenue.push(rev);

state.series.ebitda.push(ebitda);

state.series.interest.push(interest);

state.series.fcf.push(fcf);

state.series.cash.push(cash);

state.series.debt.push(debt);

state.series.netLev.push(netLev);

state.series.icr.push(icr);

state.series.dscr.push(dscr);

state.flags.push(flags);

// Render row

appendRow({q, rev, ebitda, interest, fcf, cash, debt, netLev, icr, dscr, flags});

// Update top KPIs

$('qNow').textContent = q;

renderKPIs({ pd, dscr, icr, lev: netLev, rating });

// Keep company state rolling

state.qNow = q;

state.company.cash = cash;

state.company.debt = debt;

state.company.revAnn = rev\*4; // rolling annualization for simplicity

// Repaint charts

chartsUpdate();

// Alerts

renderAlerts(q, pd, flags);

pushSnapshot();

}

function renderKPIs({pd, dscr, icr, lev, rating}) {

$('pdNow').textContent = (pd\*100).toFixed(1) + '%';

$('dscrNow').textContent = dscr.toFixed(2) + 'x';

$('icrNow').textContent = icr.toFixed(2) + 'x';

$('levNow').textContent = lev.toFixed(2) + 'x';

$('ratingNow').textContent = rating;

}

function appendRow(o) {

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

tr.innerHTML = `

<td class="pr-4 py-1">${o.q}</td>

<td class="pr-4 py-1">$${fmt(o.rev)}</td>

<td class="pr-4 py-1">$${fmt(o.ebitda)}</td>

<td class="pr-4 py-1">$${fmt(o.interest)}</td>

<td class="pr-4 py-1">$${fmt(o.fcf)}</td>

<td class="pr-4 py-1">$${fmt(o.cash)}</td>

<td class="pr-4 py-1">$${fmt(o.debt)}</td>

<td class="pr-4 py-1">${o.netLev.toFixed(2)}x</td>

<td class="pr-4 py-1">${o.icr.toFixed(2)}x</td>

<td class="pr-4 py-1">${o.dscr.toFixed(2)}x</td>

<td class="pr-4 py-1 text-rose-600">${o.flags.join(', ')}</td>

`;

$('tblBody').appendChild(tr);

}

function repaintAll(){

$('qNow').textContent = state.qNow;

$('qMax').textContent = state.qMax;

if (state.series.netLev.length){

const lastIdx = state.series.netLev.length-1;

const rating = ratingFrom(state.series.netLev[lastIdx], state.series.icr[lastIdx]);

const pd = pdHeuristic(

state.series.netLev[lastIdx],

state.series.icr[lastIdx],

state.series.dscr[lastIdx],

state.series.cash[lastIdx],

state.series.debt[lastIdx],

(state.flags[lastIdx]||[]).includes('Covenant breach')

);

renderKPIs({ pd, dscr: state.series.dscr[lastIdx], icr: state.series.icr[lastIdx], lev: state.series.netLev[lastIdx], rating });

}

// table

$('tblBody').innerHTML='';

state.series.labels.forEach((\_,i)=>{

appendRow({

q: i+1,

rev: state.series.revenue[i],

ebitda: state.series.ebitda[i],

interest: state.series.interest[i],

fcf: state.series.fcf[i],

cash: state.series.cash[i],

debt: state.series.debt[i],

netLev: state.series.netLev[i],

icr: state.series.icr[i],

dscr: state.series.dscr[i],

flags: state.flags[i]||[],

});

});

chartsUpdate(true);

}

function renderAlerts(q, pd, flags){

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

const color = pd>0.3 ? 'bg-rose-50 border-rose-200' : pd>0.15 ? 'bg-amber-50 border-amber-200' : 'bg-emerald-50 border-emerald-200';

box.className = `rounded-xl border ${color} p-3 text-sm`;

const issues = flags.length ? ` · <span class="text-rose-600">${flags.join(' | ')}</span>` : '';

box.innerHTML = `<strong>Q${q}:</strong> PD=${(pd\*100).toFixed(1)}%${issues}`;

$('alerts').prepend(box);

}

// ---------- Charts ----------

let chRevenue, chLeverage, chCashDebt, chCoverage;

function chartsInit(){

const common = {responsive:true, plugins:{legend:{display:false}}, scales:{x:{grid:{display:false}}, y:{grid:{color:'#eef2f7'}}}};

chRevenue = new Chart($('chartRevenue'), {

type: 'line', data: { labels: [], datasets:[{ label:'Revenue ($m)', data: [] }] }, options: common

});

chLeverage = new Chart($('chartLeverage'), {

type: 'line', data: { labels: [], datasets:[{ label:'Net Leverage (x)', data: [] }] }, options: common

});

chCashDebt = new Chart($('chartCashDebt'), {

type: 'line', data: { labels: [], datasets:[{ label:'Cash ($m)', data: [] }, { label:'Debt ($m)', data: [] }] }, options: common

});

chCoverage = new Chart($('chartCoverage'), {

type: 'line', data: { labels: [], datasets:[{ label:'ICR (x)', data: [] }, { label:'DSCR (x)', data: [] }] }, options: common

});

}

function chartsUpdate(full){

const L = state.series.labels;

chRevenue.data.labels = L; chRevenue.data.datasets[0].data = state.series.revenue; chRevenue.update();

chLeverage.data.labels = L; chLeverage.data.datasets[0].data = state.series.netLev; chLeverage.update();

chCashDebt.data.labels = L; chCashDebt.data.datasets[0].data = state.series.cash; chCashDebt.data.datasets[1].data = state.series.debt; chCashDebt.update();

chCoverage.data.labels = L; chCoverage.data.datasets[0].data = state.series.icr; chCoverage.data.datasets[1].data = state.series.dscr; chCoverage.update();

}

// ---------- Presets ----------

let currentPreset = 'custom';

function applyPreset(key){

currentPreset = key;

const set = (id,val)=>$(id).value=val;

if (key==='soft'){

set('gdpBase', 0.4); set('infl', 0.5); set('baseRate', 4.25); set('commShock', 0);

} else if (key==='recession'){

set('gdpBase', -0.6); set('infl', 0.3); set('baseRate', 2.75); set('commShock', 25);

} else if (key==='rateshock'){

set('gdpBase', 0.2); set('infl', 0.7); set('baseRate', 6.0); set('commShock', 0);

} else if (key==='supply'){

set('gdpBase', 0.1); set('infl', 0.9); set('baseRate', 4.75); set('commShock', 125);

} else {

// custom: do nothing

}

}

// ---------- Save / Load ----------

function saveRun(){

const key = 'credit-sim-v1';

const payload = JSON.stringify(state);

localStorage.setItem(key, payload);

alert('Saved ✓');

}

function loadRun(){

const key = 'credit-sim-v1';

const payload = localStorage.getItem(key);

if (!payload) return alert('No saved run found');

const loaded = JSON.parse(payload);

Object.assign(state, loaded);

repaintAll();

alert('Loaded ✓');

}

// ---------- Autopilot ----------

function autopilot(){

// Simple strategy: if leverage > covenant-0.3, cut capex, add costOut, stop buybacks; else modest improvement.

for (let i=0; i< (state.qMax - state.qNow); i++){

readInputs();

const lastLev = state.series.netLev[state.series.netLev.length-1] || 2.8;

if (lastLev > (state.company.covLev - 0.3)){

$('capexFlex').value = -2;

$('costOutBps').value = 100;

$('buyback').value = 0;

$('dividend').value = 10;

} else {

$('capexFlex').value = -0.5;

$('costOutBps').value = 50;

$('buyback').value = 0;

$('dividend').value = 20;

}

quarterStep();

}

}

// ---------- Events ----------

document.querySelectorAll('.preset').forEach(btn=>{

btn.addEventListener('click', (e)=>{ applyPreset(e.target.dataset.preset); });

});

$('sectorSel').addEventListener('change', (e)=>{ applySector(e.target.value); });

// Set initial sector

applySector('generic');

});

$('btnDefaults').addEventListener('click', ()=>{ document.querySelectorAll('input').forEach(inp=>inp.dispatchEvent(new Event('change'))); resetState(); });

$('btnStep').addEventListener('click', quarterStep);

$('btnAuto').addEventListener('click', autopilot);

$('btnUndo').addEventListener('click', undo);

$('btnSave').addEventListener('click', saveRun);

$('btnLoad').addEventListener('click', loadRun);

// Init

$('qMax').textContent = state.qMax;

$('year').textContent = new Date().getFullYear();

readInputs();

chartsInit();

resetState();

</script>

</body>

</html>