<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8" />

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

<title>Fore Bros — Leaderboard</title>

<script src="https://cdnjs.cloudflare.com/ajax/libs/axios/1.6.2/axios.min.js"></script>

<style>

body{font-family:system-ui,Arial;margin:18px;}

table{width:100%;border-collapse:collapse;margin-top:12px;}

th,td{border:1px solid #ddd;padding:8px;text-align:center;}

th{background:#f7f7f7;}

.top3{background:#ffeaa7;}

.warn{color:#b03;margin-left:8px;}

a.player{cursor:pointer;color:#0074d9;text-decoration:none;}

.popup{display:none;position:fixed;left:50%;top:50%;transform:translate(-50%,-50%);

background:#fff;padding:18px;border:2px solid #222;max-height:80vh;overflow:auto;z-index:999;}

.close{float:right;cursor:pointer;color:#900;}

</style>

</head>

<body>

<h1>Fore Bros — Leaderboard</h1>

<p>Season: up to 25+ weeks. Website sorts by <strong>Rounds Played (desc)</strong> then <strong>Avg Gross (Best 12) (asc)</strong>.</p>

<table id="leaderboard">

<thead>

<tr>

<th>Rank</th>

<th>Player</th>

<th>Rounds Played</th>

<th>Total Gross (Selected)</th>

<th>Avg Gross (Selected)</th>

<th>Total Net (Selected)</th>

<th>Avg Net (Selected)</th>

<th>Birdies (Selected)</th>

<th>Avg Handicap</th>

</tr>

</thead>

<tbody></tbody>

</table>

<div id="playerPopup" class="popup">

<span class="close" onclick="document.getElementById('playerPopup').style.display='none'">✕</span>

<h2 id="popupName"></h2>

<table id="popupTable">

<thead><tr><th>Week</th><th>Gross</th><th>Handicap</th><th>Net</th><th>Birdies</th><th>Include</th></tr></thead>

<tbody></tbody>

</table>

</div>

<script>

const SHEET\_JSON\_URL = "https://script.google.com/macros/s/AKfycbxqWd5MwQpYUZ6z4t5g4dZiw7-lizxulNuH6\_bHXTkq9P2YyBQPTwM9eZ13NuoLMxTmHw/exec?sheet=Scores"; // <- REPLACE THIS

let rows = [];

async function fetchSheet(){

try {

const res = await axios.get(SHEET\_JSON\_URL, { timeout: 20000 });

rows = res.data.map(r => ({

Week: r.Week,

Player: r.Player,

Gross: Number(r.Gross) || 0,

Handicap: Number(r.Handicap) || 0,

Net: (r.Net !== undefined && r.Net !== '') ? Number(r.Net) : (Number(r.Gross)||0) - (Number(r.Handicap)||0),

Birdies: Number(r.Birdies) || 0,

Include: String(r.Include||'').toUpperCase() === 'Y'

}));

buildLeaderboard();

} catch (err) {

console.error('Failed to fetch sheet:', err);

document.querySelector('#leaderboard tbody').innerHTML = '<tr><td colspan="9">Error loading data — check sheet/Web App URL</td></tr>';

}

}

function buildLeaderboard(){

// aggregate by player

const players = {};

rows.forEach(r => {

if(!r.Player) return;

if(!players[r.Player]) players[r.Player] = [];

players[r.Player].push(r);

});

const table = [];

for(const name in players){

const rounds = players[name].sort((a,b) => Number(a.Week) - Number(b.Week)); // keep sheet order by week asc

const roundsPlayed = rounds.length;

// included = only rows the admin has marked Y

let included = rounds.filter(x => x.Include);

// If more than 12 included, take the first 12 (sheet order). Also flag a warning.

let overLimit = false;

if(included.length > 12){

overLimit = true;

included = included.slice(0,12);

}

const totals = included.reduce((acc,r) => {

acc.totalGross += r.Gross;

acc.totalNet += r.Net;

acc.totalBirdies += r.Birdies;

return acc;

}, { totalGross:0, totalNet:0, totalBirdies:0 });

const avgGross = included.length ? (totals.totalGross / included.length) : null;

const avgNet = included.length ? (totals.totalNet / included.length) : null;

const avgHandicap = roundsPlayed ? (rounds.reduce((s,r)=>s+r.Handicap,0) / roundsPlayed) : null;

table.push({

Player: name,

RoundsPlayed: roundsPlayed,

TotalGross: totals.totalGross, // used for display

TotalGrossSortKey: included.length ? totals.totalGross : Infinity, // used for sorting

AvgGross: avgGross,

TotalNet: totals.totalNet,

AvgNet: avgNet,

Birdies: totals.totalBirdies,

AvgHandicap: avgHandicap,

OverLimit: overLimit,

IncludedCount: included.length

});

}

// sort: primary = rounds played (desc), secondary = total gross of selected rounds (asc)

table.sort((a,b)=>{

if (b.RoundsPlayed !== a.RoundsPlayed) return b.RoundsPlayed - a.RoundsPlayed;

if (a.TotalGrossSortKey !== b.TotalGrossSortKey) return a.TotalGrossSortKey - b.TotalGrossSortKey;

return a.Player.localeCompare(b.Player);

});

// render

const tbody = document.querySelector('#leaderboard tbody');

tbody.innerHTML = '';

table.forEach((p,i)=>{

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

if(i < 3) tr.classList.add('top3');

tr.innerHTML = `

<td>${i+1}</td>

<td><a class="player" onclick="showPopup('${escapeHtml(p.Player)}')">${escapeHtml(p.Player)}</a>${p.OverLimit?'<span class="warn">⚠️ more than 12 included</span>':''}</td>

<td>${p.RoundsPlayed}</td>

<td>${p.TotalGross || '-'}</td>

<td>${p.AvgGross ? p.AvgGross.toFixed(1) : '-'}</td>

<td>${p.TotalNet || '-'}</td>

<td>${p.AvgNet ? p.AvgNet.toFixed(1) : '-'}</td>

<td>${p.Birdies || 0}</td>

<td>${p.AvgHandicap ? p.AvgHandicap.toFixed(1) : '-'}</td>

`;

tbody.appendChild(tr);

});

}

function showPopup(playerName){

const popup = document.getElementById('playerPopup');

document.getElementById('popupName').innerText = playerName;

const tbody = document.querySelector('#popupTable tbody');

tbody.innerHTML = '';

rows.filter(r => r.Player === playerName).sort((a,b)=>Number(a.Week)-Number(b.Week)).forEach(r => {

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

tr.innerHTML = `<td>${r.Week}</td><td>${r.Gross}</td><td>${r.Handicap}</td><td>${r.Net}</td><td>${r.Birdies}</td><td>${r.Include? 'Y':'-'}</td>`;

tbody.appendChild(tr);

});

popup.style.display = 'block';

}

function escapeHtml(s){ return String(s).replace(/&/g,'&amp;').replace(/</g,'&lt;').replace(/>/g,'&gt;').replace(/"/g,'&quot;'); }

// initial fetch

fetchSheet();

// optional: auto-refresh every 10 minutes

setInterval(fetchSheet, 10\*60\*1000);

</script>

</body>

</html>