# Survivor Tool — Handoff Pack, Troubleshooting & 3‑Day Sprint (Aug 13, 2025)

This page is the **single doc** a new teammate (or a new Chat) needs to take over the Survivor Tool. It includes the current architecture, runbook, tuning knobs, troubleshooting playbook, and PyCharm + project best practices. It ends with a focused **3‑day sprint** to reach “working order,” then we pivot to the Millions dashboard.

## 0) TL;DR — Golden Path (daily / weekly)

From the repo root (PowerShell on Windows):

# 1) Fetch odds → staging  
python -m scripts.fetch\_nfl\_moneylines `  
 --api-key "<KEY>" `  
 --bookmakers "draftkings,fanduel,betmgm,caesars" `  
 --week all `  
 --staging "picks/survivor/survivor\_schedule\_roadmap\_expanded.csv" `  
 --schedule "data/2025\_nfl\_schedule\_cleaned.csv" `  
 --set-proj-wp missing  
  
# 2) Rebuild roadmap (static flags + merges)  
python -m scripts.generate\_survivor\_roadmap  
  
# 3) Recompute spot values (ALL rows)  
python -m scripts.spot\_value\_updates  
  
# 4) Generate the pick matrix  
python -m scripts.generate\_survivor\_matrix

Sanity checks:

python -c "import pandas as pd; p=r'picks/survivor/survivor\_roadmap\_expanded.csv'; d=pd.read\_csv(p); print('rows:', len(d), '| blank spot\_value\_score:', int(d['spot\_value\_score'].isna().sum()))"  
# Expect: blank spot\_value\_score: 0

**Week‑scoped preview (no overwrite):**

python -m scripts.spot\_value\_updates --week 1 --dry-run --out "picks/survivor/spot\_preview\_wk1.csv"

## 1) Project Overview — Current State

We’re building the **NFL25 Agent** to support:

* **Circa Survivor**: one SU pick per week, no repeats; holiday constraints (Thanksgiving, **Black Friday**, Christmas) handled in matrix.
* **Circa Millions** (later): five ATS picks per week, contrarian/line‑value strategy.

Right now, Survivor is green and scores for all rows. Millions is next once we lock Survivor.

## 2) Repo Layout & Canonical Files

repo/  
├─ data/  
│ └─ 2025\_nfl\_schedule\_cleaned.csv # canonical schedule (vistm/hometm or team/opponent)  
├─ picks/  
│ └─ survivor/  
│ ├─ survivor\_schedule\_roadmap\_expanded.csv # staging: odds + weekly metrics  
│ ├─ survivor\_roadmap\_expanded.csv # SOURCE OF TRUTH for Survivor planning  
│ └─ survivor\_matrix.csv # weekly decision matrix  
└─ scripts/  
 ├─ generate\_survivor\_roadmap.py # stamps rest\_days/holiday, merges staging  
 ├─ spot\_value\_updates.py # recompute ALL spot\_value scores + buckets  
 ├─ generate\_survivor\_matrix.py # applies constraints, availability, holidays  
 ├─ fetch\_nfl\_moneylines.py # pulls odds → staging (ML → implied\_wp)  
 ├─ xlsx\_to\_staging\_csv.py # convert Excel inputs → staging CSV  
 ├─ utils\_io.py / utils\_read.py / utils\_schema.py / utils\_guardrails.py  
 └─ (others, optional helpers)

**Roles:**

* **Schedule** (data/…cleaned.csv): Stable fixtures. Season‑long; rarely changes.
* **Staging** (picks/survivor/…\_roadmap\_expanded.csv): Dynamic weekly metrics (odds, spreads, rating gaps, injuries…).
* **Roadmap** (picks/survivor/survivor\_roadmap\_expanded.csv): Single source of truth; combines static flags + dynamic metrics; feeds the matrix.
* **Matrix** (picks/survivor/survivor\_matrix.csv): Week view with constraints and availability.

## 3) Scripts — What They Do

* generate\_survivor\_roadmap.py
  + Normalizes schedule to long format; stamps **rest\_days** from kickoff deltas; flags **Thanksgiving, Black Friday, Christmas**.
  + Merges in staging columns (odds, spreads, rating\_gap, injuries, DVOA, etc.).
  + **Preserves planner flags only** (reserved, expected\_avail, preferred, must\_use, save\_for\_later, notes\*, spot\_quality). **Does NOT preserve** spot\_value or spot\_value\_score.
  + Validates & writes atomically with snapshots.
* spot\_value\_updates.py (**pandas‑compatible build**)
  + ALWAYS recomputes spot\_value\_score for **all rows** using components:
    - Base: sv\_win (projected\_win\_prob), sv\_home (home/away), sv\_rest (rest\_days)
    - Layers: sv\_rating (rating\_gap/power\_gap with NaNs→0), sv\_dvoa, sv\_injury, sv\_holiday (small penalty), sv\_scarcity (future opportunity logic)
  + Buckets into spot\_value ∈ {High, Medium, Low}
  + Supports --week scoped updates and --dry-run previews.
  + Guard: fails if any spot\_value\_score remain NaN.
* generate\_survivor\_matrix.py
  + Builds week matrix with availability rules, locks, **holiday** slots (TG/BF/Christmas), and byes.
* fetch\_nfl\_moneylines.py
  + Pulls moneylines for selected books; computes implied win probability; merges to staging.
  + Consensus spread supported in the canonical odds wrapper (Circa is **backlog**).
* xlsx\_to\_staging\_csv.py
  + Converts Excel inputs into staging CSV with sensible column inference & type coercion.

## 4) Environment & PyCharm Best Practices

**Virtualenv**

* Create once and reuse: Python 3.11+ recommended.
* In PyCharm: *Settings → Project → Python Interpreter* → add your venv.

**Run Configurations** (PyCharm → Add Configuration…)

* Set **Working directory** to repo root.
* Create configs for each core script with parameters:
  + *Fetch odds*: -m scripts.fetch\_nfl\_moneylines --api-key <KEY> --bookmakers "draftkings,fanduel,betmgm,caesars" --week all --staging picks/survivor/survivor\_schedule\_roadmap\_expanded.csv --schedule data/2025\_nfl\_schedule\_cleaned.csv --set-proj-wp missing
  + *Roadmap*: -m scripts.generate\_survivor\_roadmap
  + *Spot values*: -m scripts.spot\_value\_updates
  + *Matrix*: -m scripts.generate\_survivor\_matrix

**Editor/tooling**

* Enable *“Emulate terminal in output console”* for parity with shell.
* Turn on *Black* or *Ruff* (optional) for consistent style.
* Use breakpoints + *Debug* for fast variable inspection.

**Secrets**

* API key: for now pass via --api-key. Backlog: env var ODDS\_API\_KEY and .env.

**Windows shell tips**

* PowerShell ≠ Bash. Use back‑ticks (`) for line‑continuations or prefer one‑line commands.
* Heredocs like <<'TAG' are **not** supported in PowerShell.

## 5) Spot Value Model — Tuning Map

Weights (in spot\_value\_updates.py):

* Base: W\_WIN, W\_HOME, W\_REST
* Layers: W\_RATING + RATING\_WIDTH, MAX\_DVOA\_ADJ + DVOA\_WIDTH, W\_INJURY + INJURY\_CAP, W\_HOLIDAY, W\_SCARCITY\_TOTAL + scarcity params
* Buckets: HI\_THRESH, MED\_THRESH

**Workflow to tune:**

1. Adjust constants → save.
2. python -m scripts.spot\_value\_updates → python -m scripts.generate\_survivor\_matrix.
3. Compare Week N deltas vs prior run; sanity‑check top spots (home favs with rest should rise).

**Add a new signal** (e.g., weather\_severity or qb\_status):

1. Ensure the column is present in **staging** (and thus merged into the roadmap).
2. In ensure\_columns, add a safe default (e.g., 0.0).
3. Implement add\_<signal>\_component(df) to compute df["sv\_<signal>"] and add to spot\_value\_score with bounding.
4. Re-run the pipeline.

## 6) Troubleshooting — Best Practices (Copy/Paste ready)

**A) Scores aren’t populating (NaNs)**

* Check for duplicates or stale spot columns:

@"  
import pandas as pd  
p=r'picks/survivor/survivor\_roadmap\_expanded.csv'  
df=pd.read\_csv(p)  
print('dupe cols?', df.columns.duplicated().any())  
print([c for c in df.columns if c.startswith('sv\_')])  
print('blank scores:', int(df['spot\_value\_score'].isna().sum()))  
"@ | python -

If needed, clean once:

@"  
import pandas as pd  
p=r'picks/survivor/survivor\_roadmap\_expanded.csv'  
df=pd.read\_csv(p)  
df.drop(columns=['spot\_value\_score','spot\_value'], errors='ignore', inplace=True)  
df=df.loc[:, ~df.columns.duplicated(keep='last')]  
df.to\_csv(p, index=False)  
print('Roadmap cleaned; re-run the updater')  
"@ | python -

Then re-run spot\_value\_updates.

**B) Pandas** ``\*\* error\*\*

* Use the pandas‑compatible spot\_value\_updates.py (no include\_groups in groupby). The current repo version is fixed.

**C) Roadmap won’t write (reindex duplicate labels)**

* Your on‑disk CSV has duplicate column names. Deduplicate and re-run:

@"  
import pandas as pd  
p=r'picks/survivor/survivor\_roadmap\_expanded.csv'  
df=pd.read\_csv(p)  
df=df.loc[:, ~df.columns.duplicated(keep='last')]  
df.to\_csv(p, index=False)  
print('Deduped columns written back to', p)  
"@ | python -

**D) Odds exist but** ``\*\* missing\*\*

* Ensure the fetch step ran with --set-proj-wp missing and that moneylines include both sides for each game.

**E) Imported wrong script copy**

python -c "import inspect, scripts.spot\_value\_updates as s; print('Using file:', s.\_\_file\_\_)"

Should point to **this repo**.

**F) Holiday classification looks off**

* Confirm **Black Friday** game and Thanksgiving triple‑header are flagged in the generator (we’ve added BF already). Re-run generate\_survivor\_roadmap.

## 7) How to Spin Up a New Chat (Handoff Quickstart)

1. Paste this document for context (or summarize with the sections: Overview → Runbook → Troubleshooting).
2. Confirm environment: Python 3.11+, venv active, working dir = repo root.
3. Run the **Golden Path** (Section 0) to verify local health.
4. If anything fails, follow **Troubleshooting** (Section 6) and paste outputs for fast diagnosis.

## 8) 3‑Day Sprint to Survivor “Working Order”

**Day 1 – Inputs & Integrity**

**Day 2 – Scoring Polish & UX**

**Day 3 – Guardrails & Backfill**

## 9) Backlog (post‑Survivor)

* Circa spreads ingestion (alternate source or manual CSV import flag).
* Injury and weather feeds (simple numeric inputs first, model later).
* Line move tracking: open vs close deltas.
* Backtesting harness for spot buckets (attach to historical seasons; ROC / calibration by bucket).
* API key handling via env and .env; add --dry-run to odds fetcher.
* Simple UI (streamlit or HTML report) shared by Survivor & Millions.

## 10) Acceptance Criteria — “Working Order”

* Roadmap regenerates with **0 missing** in projected\_win\_prob and spot\_value\_score.
* Matrix reflects **Thanksgiving / Black Friday / Christmas** constraints correctly.
* Spot Value updater is deterministic from inputs; re‑runs produce identical results for identical staging.
* Week report highlights coherent top spots; manual inspection agrees ≥ 80% of the time.

### Notes for Future Us

* Treat **staging** as the editable inputs layer; treat **roadmap** and **matrix** as generated artifacts.
* Avoid hand‑editing survivor\_roadmap\_expanded.csv unless for planner flags/notes.
* If something seems odd, run the Golden Path and paste the integrity checks — we’ll be back to green quickly.