```
import streamlit as st
import mlbstatsapi
import pandas as pd
from streamlit_gsheets import GSheetsConnection
from datetime import datetime

# Initialize MLB API
mlb = mlbstatsapi.Mlb()

# --- CONFIGURATION ---
spreadsheet_url =
"https://docs.google.com/spreadsheets/d/1LloUGBIfXH9fidkt8gFYiFnHpLmP6NhORvqP3jnyyuw
/edit"
is_regular_season = datetime.now() >= datetime(2026, 3, 25)

if 'watchlist' not in st.session_state:
    st.session_state.watchlist = []

# --- API FUNCTIONS ---
@st.cache_data(ttl=3600)
def fetch_hr_count(player_name, season_type="2026REG"):
    try:
        players = mlb.get_people_id(player_name)
        if not players: return 0
        stats = mlb.get_player_stats(players[0], groups=['hitting'], types=['season'],
season=season_type)
        return stats['hitting']['season'].splits[0].stat.home_runs if 'hitting' in stats else 0
    except: return 0

@st.cache_data(ttl=3600)
def get_top_ten_by_position(pos_code, season_type="2026REG"):
    try:
        leaders = mlb.get_stats_leaders(leader_categories='homeRuns', stat_group='hitting',
season=season_type, limit=10, position=pos_code)
        data = [{"Player": l.person.fullname, "Team": l.team.name, "HR": l.value} for l in
leaders[0].statleaders]
        return pd.DataFrame(data)
    except: return pd.DataFrame(columns=["Player", "Team", "HR"])

# --- APP UI ---
st.set_page_config(page_title="2026 Homer Draft", layout="wide", page_icon="⚾")
st.title("⚾ 2026 Home Run League War Room")

# Sidebar
```

```
season_mode = st.sidebar.radio("Season Phase:", ["Spring Training", "Regular Season"],
index=1 if is_regular_season else 0)
api_season_code = "2026PRE" if season_mode == "Spring Training" else "2026REG"

if st.sidebar.button('🔄 Refresh All Data'):
    st.cache_data.clear()

# Main Tabs
tab1, tab2, tab3, tab4 = st.tabs(["🏆 Standings", "🔍 Leaders", "⚔️ Head-to-Head", "📋
Watchlist"])

# 1. Connect and Load Data
conn = st.connection("gsheets", type=GSheetsConnection)
df = conn.read(spreadsheet=spreadsheet_url, worksheet="TEAMS", header=0)
managers = {
    "Chris": df.iloc[1:10, 1].tolist(), "Rob": df.iloc[1:10, 4].tolist(),
    "Mike": df.iloc[1:10, 7].tolist(), "Kenyon": df.iloc[1:10, 10].tolist()
}
positions = ["C", "1B", "2B", "3B", "SS", "LF", "CF", "RF", "DH"]

# Cache team stats for the whole run
all_team_data = {}
for m, roster in managers.items():
    p_data = [{"Position": positions[i], "Player": p, "HR": fetch_hr_count(p, api_season_code)} for
i, p in enumerate(roster)]
    all_team_data[m] = pd.DataFrame(p_data)

# --- TAB 1: STANDINGS ---
with tab1:
    standings = [{"Manager": m, "Points": df_m['HR'].sum()} for m, df_m in all_team_data.items()]
    st.table(pd.DataFrame(standings).sort_values(by="Points", ascending=False))

# --- TAB 2: POSITION LEADERS ---
with tab2:
    pos_map = {"C": "Catcher", "1B": "1st Base", "2B": "2nd Base", "3B": "3rd Base", "SS":
"Shortstop", "OF": "Outfield", "DH": "DH"}
    sel_p = st.selectbox("Position:", list(pos_map.keys()), format_func=lambda x: pos_map[x])
    l_df = get_top_ten_by_position(sel_p, api_season_code)
    st.dataframe(l_df, use_container_width=True, hide_index=True)
    to_watch = st.selectbox("Add to Watchlist:", ["None"] + l_df['Player'].tolist())
    if st.button("Add"):
        if to_watch != "None": st.session_state.watchlist.append(to_watch); st.rerun()

# --- TAB 3: HEAD-TO-HEAD ---
```

```
with tab3:
    col1, col2 = st.columns(2)
    m1 = col1.selectbox("Manager 1", list(managers.keys()), index=0)
    m2 = col2.selectbox("Manager 2", list(managers.keys()), index=1)

    # Side-by-side comparison
    comparison_df = all_team_data[m1].merge(all_team_data[m2], on="Position", suffixes=(f"
({m1})", f" ({m2})"))
    st.dataframe(comparison_df, use_container_width=True, hide_index=True)

    # Score Summary
    s1, s2 = all_team_data[m1]['HR'].sum(), all_team_data[m2]['HR'].sum()
    st.subheader(f"Score: {m1} ({s1}) vs {m2} ({s2})")
    if s1 != s2:
        st.success(f"{m1 if s1 > s2 else m2} is leading by {abs(s1-s2)} HRs!")

# --- TAB 4: WATCHLIST ---
with tab4:
    st.subheader("Players to Watch")
    if st.session_state.watchlist:
        for p in st.session_state.watchlist:
            hr = fetch_hr_count(p, api_season_code)
            st.write(f"**{p}**: {hr} HRs this season")
        if st.button("Clear All"): st.session_state.watchlist = []; st.rerun()
    else:
        st.info("Watchlist is empty. Add players from the Leaders tab.")
```