Untitled-1

```
# Step 1: Define the data structure for fielding data
import pandas as pd

# Sample data structure
fielding_data = {
    'Match No.': [],
    'Innings': [],
    'Team': [],
    'Player Name': [],
    'Ballcount': [],
    'Position': [],
    'Short Description': [],
    'Pick': [],
    'Throw': [],
    'Runs': [],
    'Overcount': [],
    'Venue': []
}

# Step 2: Function to add a fielding event to the dataset
def add_fielding_event(match_no, innings, team, player_name, ballcount, position,
                       short_desc, pick, throw, runs, overcount, venue):
    fielding_data['Match No.'].append(match_no)
    fielding_data['Innings'].append(innings)
    fielding_data['Team'].append(team)
    fielding_data['Player Name'].append(player_name)
    fielding_data['Ballcount'].append(ballcount)
    fielding_data['Position'].append(position)
    fielding_data['Short Description'].append(short_desc)
    fielding_data['Pick'].append(pick)
    fielding_data['Throw'].append(throw)
    fielding_data['Runs'].append(runs)
    fielding_data['Overcount'].append(overcount)
    fielding_data['Venue'].append(venue)

# Step 3: Performance score calculation
def calculate_performance_score(player_name, df):
    # Weights for different actions
    weights = {
        'CP': 2,     # Clean Pick
        'GT': 3,     # Good Throw
        'C': 5,      # Catch
        'DC': -4,    # Drop Catch
        'ST': 4,     # Stumping
        'RO': 5,     # Run Out
        'MRO': -2,   # Missed Run Out
        'DH': 6,     # Direct Hit
        'RS': 1      # Runs Saved
```

```
    }

    # Filter data for the player
    player_data = df[df['Player Name'] == player_name]

    # Initialize performance metrics
    cp = len(player_data[player_data['Pick'] == 'Clean Pick'])
    gt = len(player_data[player_data['Pick'] == 'Good Throw'])
    c = len(player_data[player_data['Pick'] == 'Catch'])
    dc = len(player_data[player_data['Pick'] == 'Drop Catch'])
    st = len(player_data[player_data['Throw'] == 'Stumping'])
    ro = len(player_data[player_data['Throw'] == 'Run Out'])
    mro = len(player_data[player_data['Throw'] == 'Missed Run Out'])
    dh = len(player_data[player_data['Short Description'] == 'Direct Hit'])
    rs = player_data['Runs'].sum()  # Positive for runs saved, negative for runs conceded

    # Calculate the performance score
    ps = (cp * weights['CP']) + (gt * weights['GT']) + (c * weights['C']) + \
         (dc * weights['DC']) + (st * weights['ST']) + (ro * weights['RO']) + \
         (mro * weights['MRO']) + (dh * weights['DH']) + rs

    return ps

# Step 4: Collect and organize the data
# Example data for three players (you can expand this as needed)
add_fielding_event(1, 1, 'Team A', 'Player 1', '2.3', 'Cover', 'Fielded a drive', 'Clean Pick', 'Run Out', 2, 2, 'Stadium')
add_fielding_event(1, 1, 'Team A', 'Player 2', '4.5', 'Mid-off', 'Missed a catch', 'Drop Catch', '', -4, 4, 'Stadium')
add_fielding_event(1, 1, 'Team A', 'Player 3', '6.2', 'Point', 'Direct Hit', '', 'Run Out', 1, 6, 'Stadium')

# Convert the fielding data into a DataFrame
df = pd.DataFrame(fielding_data)

# Step 5: Analyze the data and calculate performance scores for each player
players = df['Player Name'].unique()
for player in players:
    ps = calculate_performance_score(player, df)
    print(f"Performance Score for {player}: {ps}")

# Optional: Save the data to a CSV file for further analysis
df.to_csv('fielding_analysis.csv', index=False)

```