🔧 Thread-Based Tracking Fix

🎯 Problem Identified

Original Issue:

When multiple bosses spawn simultaneously and admins verify attendance in random order:

```
8:00 AM: Larba spawns → Thread created
8:01 AM: Baron spawns → Thread created
8:02 AM: Milavy spawns → Thread created

8:05 AM: Admin verifies Milavy attendance → Creates MILAVY #1 column
8:06 AM: Admin verifies Larba attendance → Creates LARBA #1 column
8:07 AM: Admin verifies Baron attendance → Creates BARON #1 column
8:08 AM: Admin verifies another Larba → Creates LARBA #2 column ❌ (WRONG!)
```

The bot was creating columns **on first attendance verification**, not when the thread was created. This caused duplicate columns for the same spawn.

✅ Solution Implemented

New Approach: Thread ID Tracking

1. **Create column immediately when thread is created** (not on first attendance)
2. **Use unique thread ID** to track which column belongs to which spawn
3. **Store mapping in SpawnLog sheet** for persistence

Flow:

```
Timer detects spawn
       ↓
Creates Discord thread → Gets unique Thread ID
       ↓
IMMEDIATELY creates column in sheet
       ↓
Logs Thread ID → Column mapping in SpawnLog
       ↓
All future attendances use Thread ID to find correct column
```

📊 New SpawnLog Sheet

Purpose:

Tracks the relationship between Discord threads and Google Sheet columns.

Structure:

Timestamp	ThreadID	Boss	SpawnNum	Date	ColumnIndex	BossHeader
10/19 8:00	142929...	Larba	1	10/19/25	5	LARBA #1
10/19 8:01	142930...	Baron	1	10/19/25	6	BARON #1
10/19 8:02	142931...	Milavy	1	10/19/25	7	MILAVY #1

Benefits:

- ✅ Persistent mapping (survives bot restarts)
- ✅ Audit trail of all spawns
- ✅ Easy to debug column issues
- ✅ Can manually fix if needed

🔄 Updated Workflow

1. Spawn Detection:

javascript

```
// OLD: Only created threads
attThread.create() → Store thread ID

// NEW: Creates threads AND column
attThread.create() → Get thread ID
                ↓
    postToSheet({action: 'createColumn', threadId, boss, spawnNum})
                ↓
    Sheet creates column & logs Thread ID → Column mapping
```

2. Attendance Verification:

javascript

```
// OLD: Found/created column by boss name (caused duplicates)
findOrCreateColumn(boss) → Mark attendance

// NEW: Finds column by Thread ID
findColumnByThreadId(threadId) → Mark attendance
```

3. Spawn Closure:

javascript

```
// OLD: Found column by boss + spawn number
findColumn(boss, spawnNum) → Finalize

// NEW: Finds column by Thread ID
findColumnByThreadId(threadId) → Finalize
```

🎯 Benefits

1. Prevents Duplicate Columns

```
✅ BEFORE FIX:
Larba #1, Baron #1, Milavy #1, Larba #2 (wrong!)

✅ AFTER FIX:
Larba #1, Baron #1, Milavy #1 (correct!)
```

2. Handles Concurrent Spawns

Multiple bosses can spawn at the same time without confusion:

```
Thread A (Larba) → Column 5
Thread B (Baron) → Column 6
Thread C (Milavy) → Column 7
Thread D (Larba) → Column 8 (different spawn, correct #2)
```

3. Reliable Column Lookup

Thread ID is unique and immutable - guaranteed to find the right column every time.

4. Survives Bot Restarts

SpawnLog sheet persists even if bot crashes - can resume tracking correctly.

🔍 Example Scenario

Multiple Spawns at Same Time:

```
8:00 AM - Timer posts 3 boss spawns simultaneously:
┌─────────────────────────────────────────────┐
│ ⚠️ Larba will spawn in 5 minutes!            │
│ ⚠️ Baron will spawn in 5 minutes!            │
│ ⚠️ Milavy will spawn in 5 minutes!           │
└─────────────────────────────────────────────┘

Bot detects all 3 → Creates threads & columns:
┌─────────────────────────────────────────────┐
│ Thread 1429... → LARBA #1 → Column 5        │
│ Thread 1430... → BARON #1 → Column 6        │
│ Thread 1431... → MILAVY #1 → Column 7       │
└─────────────────────────────────────────────┘

SpawnLog records:
┌─────────────────────────────────────────────┐
│ 1429... | Larba  | 1 | 10/19/25 | 5 | LARBA #1  │
│ 1430... | Baron  | 1 | 10/19/25 | 6 | BARON #1  │
│ 1431... | Milavy | 1 | 10/19/25 | 7 | MILAVY #1 │
└─────────────────────────────────────────────┘

8:05 AM - Members check in randomly:
┌─────────────────────────────────────────────┐
│ User A checks in Thread 1431 (Milavy)       │
│ → Looks up 1431 in SpawnLog → Column 7      │
│ → Marks attendance in Column 7 ✅           │
│                                             │
│ User B checks in Thread 1429 (Larba)        │
│ → Looks up 1429 in SpawnLog → Column 5      │
│ → Marks attendance in Column 5 ✅           │
│                                             │
│ User C checks in Thread 1430 (Baron)        │
│ → Looks up 1430 in SpawnLog → Column 6      │
│ → Marks attendance in Column 6 ✅           │
└─────────────────────────────────────────────┘

Result: Each spawn gets exactly ONE column ✅
```

🛠️ Implementation Details

Discord Bot Changes:

1. Immediate Column Creation:

javascript

```javascript
// Right after thread creation:
const createColumnPayload = {
  action: 'createColumn',
  threadId: attThread.id,
  boss: bossName,
  spawnNum,
  date: today
};
await postAttendanceToSheet(createColumnPayload);
```

2. Thread ID in Attendance:

javascript

```javascript
// When admin verifies:
const payload = {
  action: 'recordAttendance',
  threadId: msg.channel.id,  // ← Uses thread ID
  user: pending.author,
  boss: pending.boss,
  verifier: user.username
};
```

3. Thread ID in Finalization:

javascript

```javascript
// When closing spawn:
await postAttendanceToSheet({
  action: 'finalizeSpawn',
  threadId: msg.channel.id  // ← Uses thread ID
});
```

Apps Script Changes:

1. New SpawnLog Sheet:

javascript

```javascript
function getOrCreateSpawnLog() {
  // Creates SpawnLog sheet with structure:
  // ThreadID | Boss | SpawnNum | Date | ColumnIndex | BossHeader
  // Acts as persistent mapping between Discord threads and sheet columns
}
```

2. Column Creation Handler:

javascript

```javascript
function handleCreateColumn(data) {
  // 1. Check if thread already has column (prevent duplicates)
  // 2. If not, create new column
  // 3. Log thread ID → column mapping to SpawnLog
  // 4. Return column info
}
```

3. Attendance Handler:

javascript

```javascript
function handleAttendance(data) {
  // 1. Look up thread ID in SpawnLog
  // 2. Get column index from SpawnLog
  // 3. Mark attendance in that specific column
  // 4. No more finding/creating columns on attendance!
}
```

4. Finalization Handler:

javascript

```javascript
function handleFinalizeSpawn(data) {
  // 1. Look up thread ID in SpawnLog
  // 2. Get column index
  // 3. Mark all unchecked cells as absent
}
```

📋 Testing the Fix

Test Case 1: Multiple Simultaneous Spawns

Steps:

1. Post 3 spawn messages at same time
2. Check SpawnLog sheet - should have 3 entries
3. Check attendance sheet - should have 3 columns
4. Verify attendance in random order
5. Check that each attendance goes to correct column

Expected Result:

```
SpawnLog:
1429... | Larba  | 1 | 5 | LARBA #1
1430... | Baron  | 1 | 6 | BARON #1
1431... | Milavy | 1 | 7 | MILAVY #1

Sheet Columns:
E: LARBA #1 | F: BARON #1 | G: MILAVY #1

No duplicate columns! ✅
```

Test Case 2: Same Boss Multiple Spawns

Steps:

1. Post Larba spawn message (creates thread A)
2. Wait 30 seconds
3. Post Larba spawn message again (creates thread B)
4. Verify attendance in both threads

Expected Result:

```
SpawnLog:
1429... | Larba | 1 | 5 | LARBA #1
1432... | Larba | 2 | 6 | LARBA #2

Sheet Columns:
E: LARBA #1 | F: LARBA #2

Correct numbering! ✅
```

Test Case 3: Bot Restart Recovery

Steps:

1. Create several spawns
2. Restart bot
3. Verify attendance for old spawns

Expected Result:

- SpawnLog persists in Google Sheets
- Bot can still look up columns by thread ID
- Attendance marks correctly even after restart ✅

🔧 Manual Fixes (If Needed)

If Duplicate Columns Were Created Before Fix:

Option 1: Merge Columns Manually

1. Open Google Sheet
2. Find duplicate columns (e.g., two LARBA #1 columns)
3. Manually copy data from duplicate to original
4. Delete duplicate column

Option 2: Update SpawnLog

1. Open SpawnLog sheet
2. Find entries for duplicate spawns
3. Update ColumnIndex to point to correct column
4. Future verifications will use correct column

Option 3: Start Fresh

1. Create new weekly sheet
2. Old week remains as historical data
3. New week uses fixed system

📊 Monitoring & Debugging

Check SpawnLog Sheet:

```
If column not found error:
1. Open SpawnLog sheet
2. Search for thread ID
3. Check if entry exists
4. If missing, column wasn't created properly
```

Apps Script Logs:

```
Look for:
✅ "Created column for thread 1429..."
✅ "Attendance recorded for thread 1429..."
✅ "Finalized spawn for thread 1429..."

Errors:
❌ "Spawn column not found for this thread"
  → Thread not in SpawnLog
  → Column wasn't created when thread was made
```

Discord Bot Logs:

```
Look for:
✅ "Column created for Larba spawn #1"
✅ "Admin verifying user for Larba"

Errors:
❌ "Failed to create column: ..."
  → Check webhook URL
  → Check Apps Script deployment
```

🎯 Summary

What Changed:

Aspect	Before	After
Column Creation	On first attendance	On thread creation
Column Lookup	By boss name	By thread ID
Duplicate Prevention	Unreliable	Guaranteed
Concurrent Spawns	Problematic	Handled perfectly
Restart Recovery	Lost state	Persists in SpawnLog

Key Files Updated:

1. **index.js (Discord Bot)**
   - Immediately creates column when thread created
   - Uses thread ID for all sheet operations
   - Sends thread ID with every webhook call
2. **Code.gs (Apps Script)**
   - New SpawnLog sheet for thread tracking

    - handleCreateColumn() for thread creation
    - findColumnByThreadId() for lookups
    - All operations use thread ID

Result:

✅ **No more duplicate columns** ✅ **Handles any number of concurrent spawns** ✅ **Reliable column tracking** ✅ **Survives bot restarts** ✅ **Easy to debug and fix issues**

🚀 Deployment Steps

1. **Update Discord Bot:**
    - Replace index.js with updated version
    - Restart bot
2. **Update Apps Script:**
    - Replace Code.gs with updated version
    - Re-deploy as Web App
    - Update webhook URL in config.json if changed
3. **Test:**
    - Post test spawn messages
    - Check SpawnLog sheet created
    - Verify attendance works
    - Test concurrent spawns
4. **Monitor:**
    - Check SpawnLog for entries
    - Watch for any "column not found" errors
    - Verify no duplicate columns created

System is now production-ready! 🎉