TWS Bridge Communication Fixes

Summary

Fixed critical Python bridge communication issues that were causing:
1. Account balance not displaying after connection
2. Order placement timing out (despite orders being placed successfully)
3. Positions not showing in the UI

Root Cause

The Python bridge (`tws_bridge.py`) was **not including the `requestId`** in responses sent back to the Electron app. This prevented the Electron app from matching responses to pending requests, causing all promises to timeout.

Fixes Implemented

1. Response Handling - `send_response()` Function

Before:

```python
def send_response(response):
    """Send JSON response to stdout"""
    print(json.dumps(response), flush=True)
```

After:

```python
def send_response(response, request_id=None):
    """Send JSON response to stdout"""
    if request_id is not None:
        response['requestId'] = request_id
    print(json.dumps(response), flush=True)
    log(f"Sent response: {json.dumps(response)}")
```

Impact: All responses now include the `requestId`, allowing the Electron app to resolve promises correctly.

---

2. Command Handling - `handle_command()` Function

Before:

```python
def handle_command(command):
    cmd_type = command.get('type')

    if cmd_type == 'place_order':
        # ... process order
        send_response(result)  # ❌ No requestId
```

After:

```
def handle_command(command):
    cmd_type = command.get('type')
    request_id = command.get('requestId')  # ✓ Extract requestId

    log(f"Handling command: {cmd_type} with requestId: {request_id}")

    try:
        if cmd_type == 'place_order':
            # ... process order
            send_response(result, request_id)  # ✓ Include requestId
    except Exception as e:
        log(f"Error handling command: {str(e)}")
        send_response({"success": False, "message": f"Error: {str(e)}"}, request_id)
```

Impact:
- RequestId is now extracted from incoming commands
- All responses include the requestId
- Comprehensive error handling ensures responses are always sent
- Debug logging added for troubleshooting

---

3. Balance Fetching Improvements

Added:
- Detailed logging of account values received from TWS
- Logging of NetLiquidation value extraction
- Warning when balance is 0
- Better error handling and reporting

Functions improved:
- `get_balance_ib_insync()`
- `get_balance_ibapi()`

Impact: Better visibility into why balance might not be displaying correctly.

---

4. Order Placement Response

Before:

```
def place_order_ib_insync(...):
    # Place order
    trade = ib.placeOrder(contract, order)
    ib.sleep(1)

    return {
        "success": True,
        "message": f"Order placed..."
    }
    # Response sent to Electron WITHOUT requestId ❌
```

After:

```
def place_order_ib_insync(...):
    # Place order
    trade = ib.placeOrder(contract, order)
    ib.sleep(1)

    return {
        "success": True,
        "message": f"Order placed..."
    }
    # handle_command() adds requestId and sends response ✓
```

Impact: Order placement responses now include requestId, eliminating timeout errors.

---

5. Positions Fetching Improvements

Added:
- Detailed logging of positions received from TWS
- Logging of position count before and after processing
- Better error handling for individual position processing
- Type conversion to ensure JSON serialization

Functions improved:
- `get_positions_ib_insync()`
- `get_positions_ibapi()`

Impact: Better visibility into why positions might not be showing correctly.

Technical Details

Communication Flow (After Fix)

```
Electron (main.js)
  ↓
  1. Generates unique requestId
  2. Sends command: {type: 'get_balance', requestId: 123.456}
  ↓
Python (tws_bridge.py)
  ↓
  3. Extracts requestId from command
  4. Processes command
  5. Includes requestId in response: {success: true, balance: 10000, requestId: 123.456}
  ↓
Electron (main.js)
  ↓
  6. Matches requestId to pending promise
  7. Resolves promise with response data
```

Logging Added

All operations now log:
- Command type and requestId received

- Processing steps
- Results being returned
- Errors encountered

This allows for easy debugging by checking:
- Electron console (press F12 in the app)
- Python stderr output

Testing Instructions

1. Prerequisites

- TWS or IB Gateway running (Paper Trading recommended)
- TWS configured to accept API connections on port 7496
- Enable ActiveX and Socket Clients in TWS

2. Start the Application

```
cd /home/ubuntu/tws_electron_app
npm start
```

3. Connect to TWS

Use the connection parameters from the screenshot:
- Host: 127.0.0.1
- Port: 7496
- Client ID: 1

4. Verify Fixes

Test 1: Balance Display

Expected: After successful connection, balance should display immediately (e.g., “$10,000.00”)

Check logs:

```
Python stderr: Getting balance...
Python stderr: Got 50 account values from TWS
Python stderr: Found NetLiquidation: 10000.0
```

Test 2: Order Placement

Expected: Clicking “Buy Market Order” should:
1. Show success message immediately
2. Order appears in TWS
3. No timeout error

Check logs:

```
Python stderr: Handling command: place_order with requestId: 1731614400.123
Python stderr: Placing order: {action: 'BUY', ticker: 'SPY', ...}
Python stderr: Sent response: {success: true, message: '...', requestId:
1731614400.123}
```

Test 3: Positions Display

Expected: If you have positions in TWS, they should appear in the positions table

Check logs:

```
Python stderr: Getting positions...
Python stderr: Got 3 positions from TWS
Python stderr: Returning 3 positions
```

Debugging Tips

If Balance Still Shows “$–“

1. Check Python stderr for “Getting balance…” message
2. Look for “Got X account values” - if 0, TWS may not be connected properly
3. Check if “NetLiquidation” is found in the account values
4. Verify TWS is in Paper Trading mode with simulated balance

If Orders Still Timeout

1. Check Python stderr for “Handling command: place_order”
2. Look for “Sent response” message with requestId
3. Verify the requestId in the response matches the one in Electron console
4. Check for any error messages in Python stderr

If Positions Still Don’t Show

1. Verify you actually have positions in TWS
2. Check Python stderr for “Got X positions”
3. Look for any error messages during position processing
4. Verify TWS API is enabled and accepting connections

Files Modified

- `tws_bridge.py` - Main bridge script with all fixes

Files Added

- `test_bridge.py` - Test script to verify fixes
- `BUGFIX_SUMMARY.md` - This document

Commit Message

```
fix: Add requestId to all Python bridge responses

- Modified send_response() to accept and include requestId
- Updated handle_command() to extract and pass requestId
- Added comprehensive error handling to ensure responses always sent
- Added detailed logging for debugging balance, orders, positions
- Fixes: balance not displaying, order timeout, positions not showing

Closes: Python bridge communication issues
```