API Activity Card - Implementation Summary

Overview

A real-time rolling text card has been added to the right side of every page, displaying live API activity logs with visual feedback and statistics.

🎨 Visual Design

Card Features:

- **Dark terminal theme** - Black gradient background with green accent colors
- **Sticky positioning** - Stays visible while scrolling
- **Rolling log display** - Shows latest 50 API activities
- **Activity statistics** - Success/Error/Total counters
- **Animated entries** - Slide-in animation for new logs
- **Pulse indicator** - Animated green dot showing active status
- **Responsive design** - Hides on screens < 1200px width

Color Coding:

- 🟢 **Success** - Green (#00ff88) - API calls that succeeded

- 🔴 **Error** - Red (#ff4757) - Failed API calls
- 🔵 **Info** - Blue (#3498db) - General information
- 🟡 **Warning** - Orange (#ffa502) - Warnings and notices

📁 Files Modified

---

1. templates/base.html

Layout Changes:

- Added two-column layout with flex display
- Main content area (flex: 1)
- Activity sidebar (fixed 350px width)

New CSS Classes:

```
.main-layout          /* Flex container */
.main-content         /* Left column - main content */
.activity-sidebar     /* Right column - activity card */
.api-activity-card    /* Dark card container */
.api-activity-header  /* Card header with pulse indicator */
.activity-log         /* Scrollable log container */
.log-entry           /* Individual log entry */
.log-timestamp       /* Time of log entry */
.log-type            /* Badge showing log type */
.log-message         /* Log message text */
.activity-stats      /* Statistics grid at bottom */
.stat-item           /* Individual stat */
```

Activity Card HTML Structure:

```html
<div class="activity-sidebar">
    <div class="api-activity-card">
        <!-- Header with pulse indicator -->
        <div class="api-activity-header">
            <div class="status-indicator"></div>
            <h3>API Activity Log</h3>
        </div>

        <!-- Scrollable log area -->
        <div class="activity-log" id="activityLog">
            <!-- Logs populated from session -->
        </div>

        <!-- Statistics -->
        <div class="activity-stats">
            <!-- Success/Error/Total counters -->
        </div>
    </div>
</div>
```

JavaScript Added:

- Form submission monitoring
- Log entry management (keeps last 50)
- Auto-scroll to latest entries
- Session storage persistence
- Dynamic stat updates

---

2. app.py

New Function:

```python
def log_api_activity(activity_type, message):
    """Log API activity to session for display in frontend"""
    if 'api_activity' not in session:
        session['api_activity'] = []

    session['api_activity'].append({
        'type': activity_type,      # 'success', 'error', 'info', 'warning'
        'message': message,          # Description of activity
        'timestamp': datetime.now().strftime('%H:%M:%S')
    })

    # Keep only last 50 entries
    if len(session['api_activity']) > 50:
        session['api_activity'] = session['api_activity'][-50:]

    session.modified = True
```

Activity Logging Added To:

Authentication Phase: - `log_api_activity('info', 'Starting data fetch for X data types')` - `log_api_activity('info', 'Authenticating with Commvault API...')` - `log_api_activity('success', 'Authenticated as: username')` - `log_api_activity('error', 'Authentication failed - invalid credentials')`

Data Fetching Phase: - `log_api_activity('info', 'Fetching Clients data...')` - `log_api_activity('success', 'Retrieved 3,604 clients')` - `log_api_activity('info', 'Fetching Jobs (last 24h)...')` - `log_api_activity('success', 'Retrieved 120 jobs')` - `log_api_activity('info', 'Fetching MediaAgents...')` - `log_api_activity('success', 'Retrieved 10 MediaAgents')` - `log_api_activity('info', 'Fetching Storage Pools...')` - `log_api_activity('success', 'Retrieved 8 storage pools')` - `log_api_activity('error', 'Clients fetch failed: HTTP 404')`

🎯 Features

1. Real-Time Activity Logging

- Every API call is logged with timestamp
- Shows what's being fetched
- Displays success/failure status
- Shows count of items retrieved

2. Visual Feedback

- Color-coded log entries by type
- Animated slide-in for new entries
- Pulsing status indicator
- Border colors match log type

3. Statistics Dashboard

- **Success Count** - Total successful API calls
- **Error Count** - Total failed API calls
- **Total Count** - All API activities logged

4. Session Persistence

- Logs stored in Flask session
- Persists across page navigation
- Automatically cleared when session ends
- Shows historical activity

5. Auto-Scrolling

- Latest entries appear at top
- Auto-scrolls to show new activity
- Scrollbar for viewing older entries

- Keeps last 50 entries

📊 Example Activity Flow

Typical Data Fetch Session:

```
10:15:23 INFO    Starting data fetch for 4 data types
10:15:23 INFO    Target: commvaultweb01.jhb.seagatestoragecloud.co.za
10:15:23 INFO    Authenticating with Commvault API...
10:15:24 SUCCESS Authenticated as: guys@storvault.co.za
10:15:24 INFO    Fetching Clients data...
10:15:26 SUCCESS Retrieved 3,604 clients
10:15:26 INFO    Fetching Jobs (last 24h)...
10:15:27 SUCCESS Retrieved 120 jobs
10:15:27 INFO    Fetching MediaAgents...
10:15:28 SUCCESS Retrieved 10 MediaAgents
10:15:28 INFO    Fetching Storage Pools...
10:15:29 SUCCESS Retrieved 8 storage pools
```

Statistics: - Success: 5 - Errors: 0 - Total: 9

🔧 Usage

For Users:

1. Navigate to any page (Home, Dashboard, etc.)
2. Look at the right sidebar
3. Watch logs appear as you fetch data
4. Monitor success/error rates

5. Review recent API activity

For Developers:

Add activity logs anywhere in app.py:

```
# Success
log_api_activity('success', 'Operation completed successfully')

# Error
log_api_activity('error', f'Failed with HTTP {status_code}')

# Info
log_api_activity('info', 'Processing data...')

# Warning
log_api_activity('warning', 'Deprecated endpoint used')
```

🎨 Customization

Change Colors:

Edit in templates/base.html CSS:

```
/* Success color */
.log-entry.success { border-left-color: #00ff88; }
.log-type.success { background: #00ff88; }

/* Error color */
.log-entry.error { border-left-color: #ff4757; }
.log-type.error { background: #ff4757; }

/* Info color */
.log-entry.info { border-left-color: #3498db; }
.log-type.info { background: #3498db; }
```

```
/* Warning color */
.log-entry.warning { border-left-color: #ffa502; }
.log-type.warning { background: #ffa502; }
```

Change Card Size:

```
.activity-sidebar {
    width: 350px;  /* Change width */
}

.api-activity-card {
    height: calc(100vh - 200px);  /* Change height */
}
```

Change Log Limit:

In app.py:

```
# Keep last 50 entries (change to any number)
if len(session['api_activity']) > 50:
    session['api_activity'] = session['api_activity'][-50:]
```

📱 Responsive Behavior

Desktop (> 1200px):

- Activity card visible on right side
- Two-column layout
- Sticky positioning

Tablet/Mobile (< 1200px):

- Activity card hidden
- Full-width main content
- Better mobile experience

🎯 Benefits

For Users:

✅ **Transparency** - See exactly what's happening ✅ **Debugging** - Identify failed API calls immediately ✅ **Monitoring** - Track success rates in real-time ✅ **History** - Review recent activity

For Developers:

✅ **Easy logging** - Simple function call ✅ **Persistent** - Session-based storage ✅ **Visual** - Beautiful terminal-style display ✅ **Flexible** - Easy to extend

---

🔍 Technical Details

Session Storage:

```
session['api_activity'] = [
    {
        'type': 'success',
        'message': 'Retrieved 3,604 clients',
        'timestamp': '10:15:26'
    },
    ...
]
```

Frontend Rendering:

```
{% for activity in session.get('api_activity')[::-1] %}
<div class="log-entry {{ activity.type }}">
    <span class="log-timestamp">{{ activity.timestamp }}</span>
    <span class="log-type {{ activity.type }}">
        {{ activity.type.upper() }}
    </span>
    <div class="log-message">{{ activity.message }}</div>
</div>
{% endfor %}
```

Statistics Calculation:

```
{% set success_count = session.get('api_activity', [])
    |selectattr('type', 'equalto', 'success')
    |list|length %}
```

✨ Future Enhancements

Potential improvements:

1. **Export Logs** - Download activity log as CSV
2. **Filter by Type** - Show only success/error/info/warning
3. **Search** - Search through log messages
4. **Time Range** - Filter by time range
5. **Clear Logs** - Button to clear current session logs
6. **Real-time Updates** - WebSocket for live updates

7. **Notification Badges** - Show error count in nav
8. **Sound Alerts** - Audio notification for errors

📝 Summary

Files Modified: 2 - templates/base.html (~200 lines added) - app.py (~30 lines added)

Features Added: - ✅ Rolling activity log display - ✅ Real-time statistics - ✅ Color-coded log types - ✅ Session persistence - ✅ Auto-scrolling - ✅ Animated entries - ✅ Responsive design

Ready to use! Start the Flask app and watch the activity log in action as you fetch data from Commvault.