Lab Report — Visualizations & Dashboards

Name: Dhruvish
Date: October 27, 2025
Platform: Splunk Cloud Trial

Objective

This lab focused on transforming raw search results into **effective visualizations** and combining them into a dynamic **multi-panel dashboard**.
You learned to:

- Select appropriate visualization types for different data stories
- Add dashboard **inputs (tokens)** for interactivity
- Configure **drilldowns** for deeper analysis
- Optimize dashboards using **base searches** for performance

These are key skills for building **real-world Splunk dashboards** that communicate system health and trends efficiently.

Tools Used

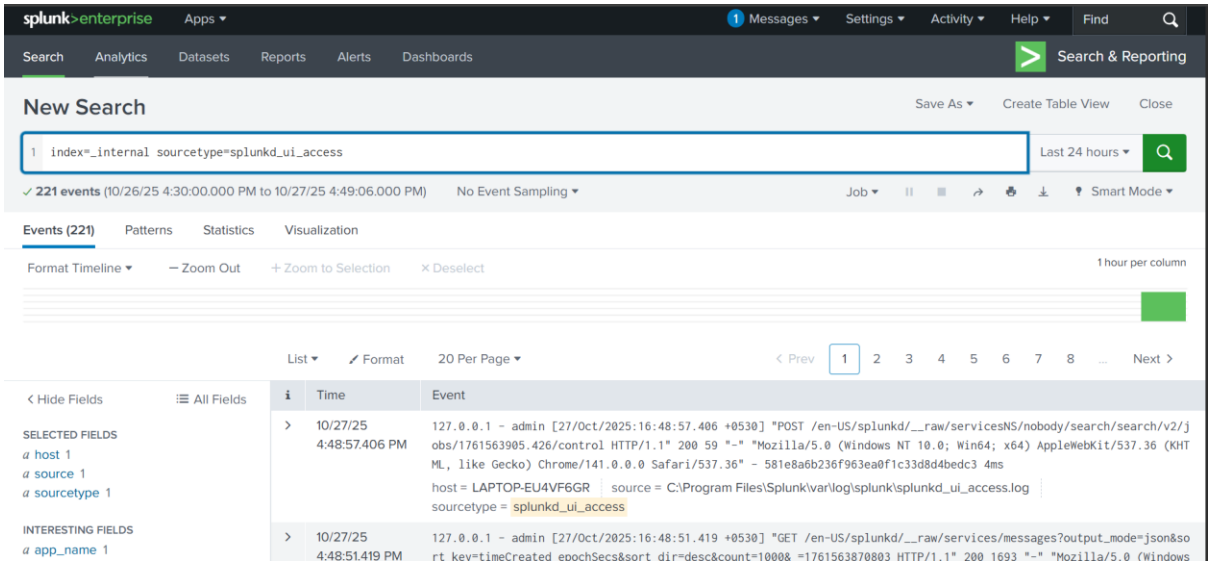
- **Splunk Cloud Trial** (or Splunk Free)
- **Web Browser**
- **Dashboard Studio** (for visual editing)

Procedure and Observations

Step 1: Prepare Dataset & Time Range

- Opened **Apps ▸ Search & Reporting**.
- Set **time range** to *Last 24 hours*.
- Used internal Splunk logs as dataset:

  index=_internal sourcetype=splunkd_ui_access
- Confirmed presence of fields: status, uri_path, user, method, and timestamps.



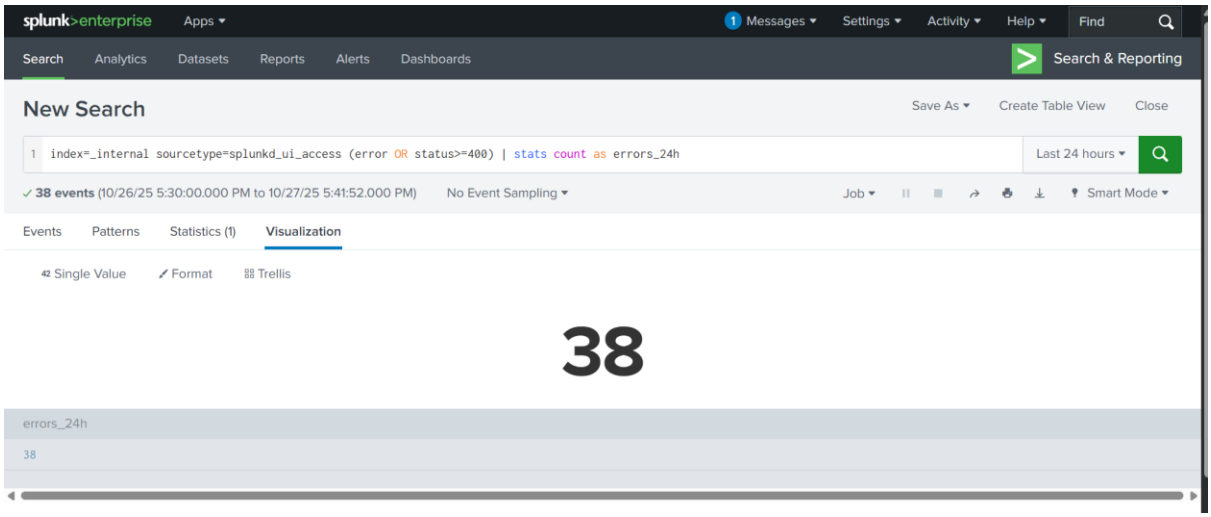
Step 2: Build Three Reusable Searches

1 KPI – Errors in Last 24h (Single Value)

index=_internal sourcetype=splunkd_ui_access (error OR status>=400)

| stats count as errors_24h

- Verified a **Single Value visualization** appeared in the Visualization tab.
- Displays total error count in past 24 hours.



2 Trend – Requests by Class (Line/Area Chart)

index=_internal sourcetype=splunkd_ui_access

| eval class=case(

status>=500,"5xx",

status>=400,"4xx",

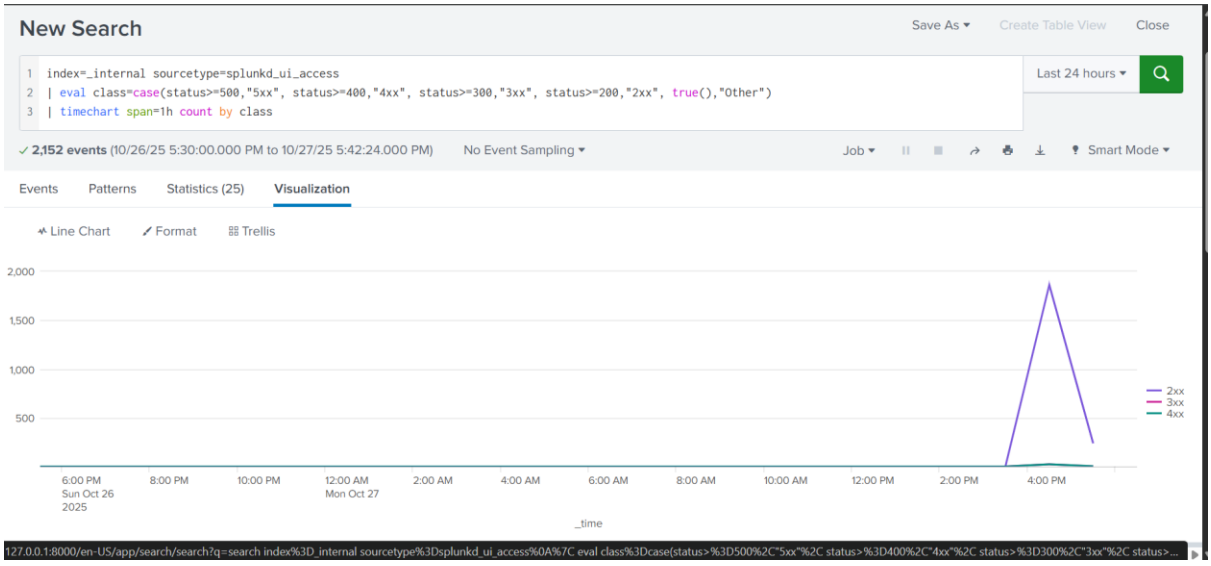
status>=300,"3xx",

status>=200,"2xx",

true(),"Other")

| timechart span=1h count by class

- Produced an hourly trend of request classes (Success, Client Error, Server Error, etc.).
- Ideal for spotting spikes or performance degradations.



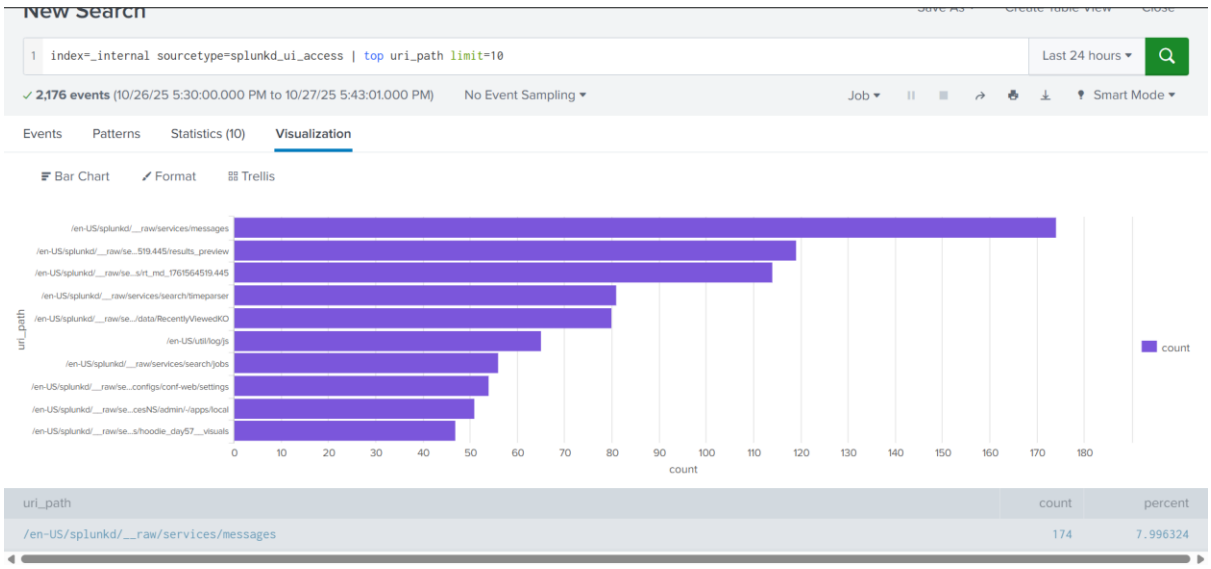
---

3 Top Paths (Bar Chart)

index=_internal sourcetype=splunkd_ui_access

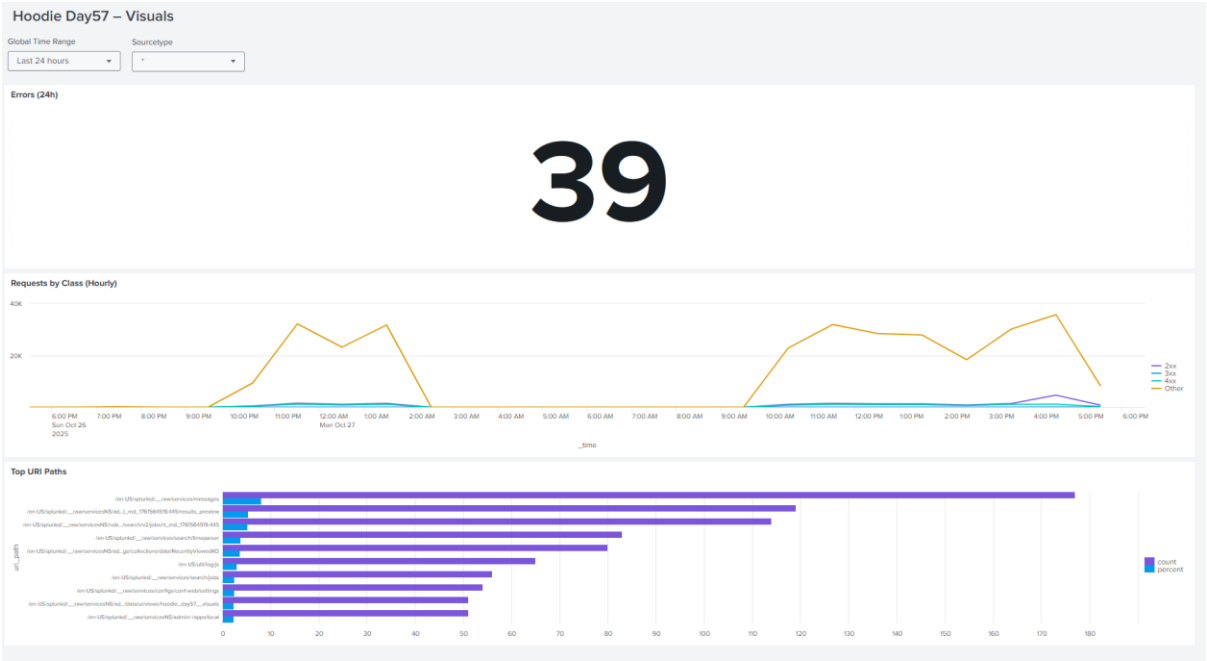
| top uri_path limit=10

- Displayed top accessed URI paths with corresponding counts.
- Converted to **Vertical Bar Chart** for better readability.



Step 3: Create the Dashboard

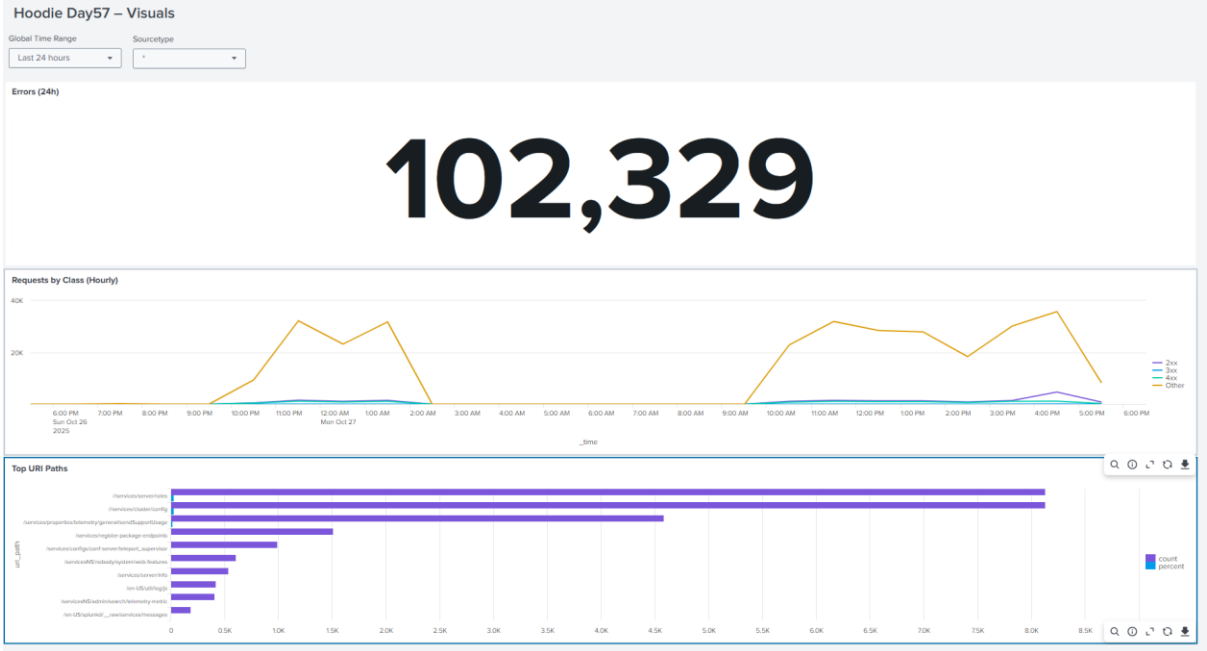
- Created a new dashboard:
  Name: *Hoodie Day57 – Visuals*
  Type: *Dashboard Studio*
- Added the three searches as panels:
  - *Errors (24h)* → Single Value
  - *Requests by Class (Hourly)* → Line Chart
  - *Top URI Paths* → Column Chart



Step 4: Add Inputs (Tokens)

- Added a **Dropdown Input** labeled *Sourcetype* in the dashboard editor.
- Created token: $tok_sourcetype$
- Options:
  - splunkd_ui_access *(default)*
  - splunkd
- Edited all search queries to include the tokenized filter:

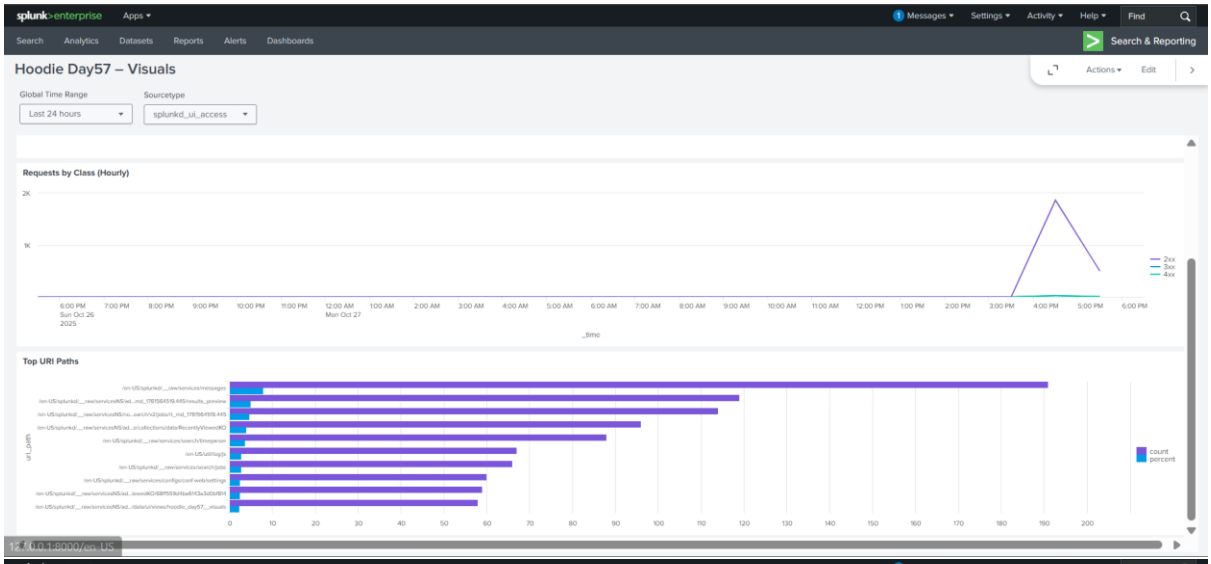
  index=_internal sourcetype=$tok_sourcetype$
- Tested dropdown—confirmed all panels updated dynamically when switching sourcetypes.

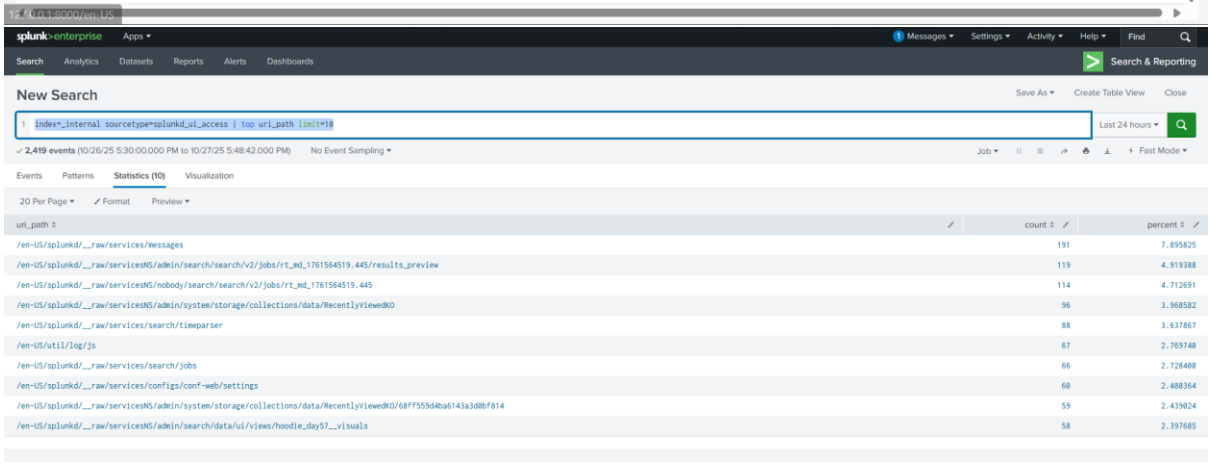


Step 5: Enable Drilldowns

- Configured **drilldown** for the “Top URI Paths” panel:
  - **Action:** *Link to New Search*
  - **Target SPL:**

    index=_internal sourcetype=$tok_sourcetype$ uri_path="$click.value$" earliest=$time.earliest$ latest=$time.latest$
- Clicking a bar now opens a detailed search filtered by the selected uri_path.





Step 6: (Optional) Base Search for Performance

For Classic Simple XML dashboards, you can define a **base search** and reuse it for all panels:

Base: index=_internal sourcetype=$tok_sourcetype$ earliest=$time.earliest$ latest=$time.latest$

Post-process:

| timechart span=1h count by class

| stats count as errors_24h

| top uri_path limit=10

This reduces redundant data scans, improving performance on larger datasets.

Step 7: Polish the Dashboard

- Set dashboard auto-refresh interval: **60 seconds**
- Applied **threshold coloring** to Single Value panel:
    - Green < 50

    - Yellow 50–200
    - Red > 200
- Added **Trellis visualization** for request trends by host:

  index=_internal sourcetype=$tok_sourcetype$

  | timechart span=1h count by host

Enabled *Trellis = by series* to create small multiples per host.

Reflection

- **Which panel communicated change fastest?**
  The *Single Value KPI (Errors 24h)* immediately highlighted anomalies, ideal for quick health checks.
- **How did tokens improve efficiency?**
  Tokens allowed reusing the same dashboard logic for multiple sourcetypes, reducing duplication and simplifying maintenance.
- **What extra panel would add value?**
  A *"4xx Errors by URI"* or *"Response Time Trend"* panel would provide actionable insight for troubleshooting user-facing issues.

Summary

This lab demonstrated how to:

- Transform searches into meaningful visuals
- Combine panels into an interactive, token-driven dashboard
- Implement drilldowns for context
- Optimize performance with base searches and refresh intervals

These techniques enable you to craft **interactive, efficient Splunk dashboards** that drive fast decision-making and effective incident analysis.