Setting Up Grafana Dashboard Using API Endpoints

# 1. Prerequisites

Before you start using the API, make sure you have:

- Grafana installed and running.

- An API key (you can generate one from the Grafana UI under Configuration > API Keys).

- API access permissions (Editor or Admin for dashboard creation).

# 2. Creating an API Key

1. Log in to Grafana as an administrator.

2. Navigate to Configuration > API Keys.

3. Click on New API Key, give it a name, set the role to Editor or Admin, and click Add.

4. Copy and save the API key (you won’t be able to see it again).

# 3. Using Grafana API Endpoints

Grafana’s API works over HTTP(S), and the typical request URL format is:

http://<your-grafana-server>/api/<resource>

For dashboards, the relevant API endpoints are located at /api/dashboards.

## 4. Common Endpoints for Dashboards:

- Create/Update Dashboard: POST /api/dashboards/db

- Delete Dashboard: DELETE /api/dashboards/uid/:uid

- Get Dashboard by UID: GET /api/dashboards/uid/:uid

# 5. Create/Update a Dashboard Using API

You can use a POST request to create or update a dashboard.

## API Endpoint

POST /api/dashboards/db

## Request Headers

Authorization: Bearer <API\_KEY>  
Content-Type: application/json

## Payload (JSON)

{  
 "dashboard": {  
 "id": null,  
 "uid": null,  
 "title": "New Dashboard",  
 "tags": ["templated"],  
 "timezone": "browser",  
 "schemaVersion": 16,  
 "version": 0,  
 "refresh": "25s",  
 "panels": [  
 {  
 "type": "graph",  
 "title": "CPU Usage",  
 "gridPos": {  
 "h": 9,  
 "w": 12,  
 "x": 0,  
 "y": 0  
 },  
 "targets": [  
 {  
 "expr": "rate(node\_cpu\_seconds\_total[5m])",  
 "legendFormat": "{{cpu}}",  
 "refId": "A"  
 }  
 ]  
 }  
 ]  
 },  
 "folderId": 0,  
 "overwrite": false  
}

## Example curl Command

curl -X POST http://<grafana-server>/api/dashboards/db \  
 -H "Authorization: Bearer <API\_KEY>" \  
 -H "Content-Type: application/json" \  
 -d '{  
 "dashboard": {  
 "id": null,  
 "uid": null,  
 "title": "New Dashboard",  
 "tags": ["templated"],  
 "timezone": "browser",  
 "schemaVersion": 16,  
 "version": 0,  
 "refresh": "5s",  
 "panels": [  
 {  
 "type": "graph",  
 "title": "CPU Usage",  
 "gridPos": {  
 "h": 9,  
 "w": 12,  
 "x": 0,  
 "y": 0  
 },  
 "targets": [  
 {  
 "expr": "rate(node\_cpu\_seconds\_total[5m])",  
 "legendFormat": "{{cpu}}",  
 "refId": "A"  
 }  
 ]  
 }  
 ]  
 },  
 "folderId": 0,  
 "overwrite": false  
 }

# 6. Retrieve an Existing Dashboard

You can retrieve a dashboard by its UID with a GET request.

## API Endpoint

GET /api/dashboards/uid/:uid

## Example curl Command

curl -X GET http://<grafana-server>/api/dashboards/uid/<dashboard\_uid> \  
 -H "Authorization: Bearer <API\_KEY>"

# 7. Delete a Dashboard

To delete a dashboard, use the DELETE request with the dashboard's UID.

## API Endpoint

DELETE /api/dashboards/uid/:uid

## Example curl Command

curl -X DELETE http://<grafana-server>/api/dashboards/uid/<dashboard\_uid> \  
 -H "Authorization: Bearer <API\_KEY>"

# 8. Other Useful Endpoints

- Search Dashboards: GET /api/search?query=<dashboard\_name>

- List All Folders: GET /api/folders

- Create Folder: POST /api/folders

# 9. Handling Authentication

All API requests need to include the Authorization header with your API key:

Authorization: Bearer <API\_KEY>

# 10. Error Handling

If something goes wrong with your API request, Grafana will return a response with an error message. Check the response status code and body for debugging.

# 11. Conclusion

Using the Grafana API to create and manage dashboards allows you to automate many of your workflows. By following the examples above, you can integrate dashboard setup into your DevOps pipelines or dynamically create dashboards based on data or alerts.