# Fritz: the Zwicky Transient Facility Phase-II Marshal

Dmitry A. Duev <duev@caltech.edu>

on behalf of the Fritz Dev Team

# Fritz Core Dev Team

Joshua Bloom

Michael Coughlin

Arien Crellin-Quick

Dmitry Duev

Daniel Goldstein

Matthew Graham

Mansi Kasliwal

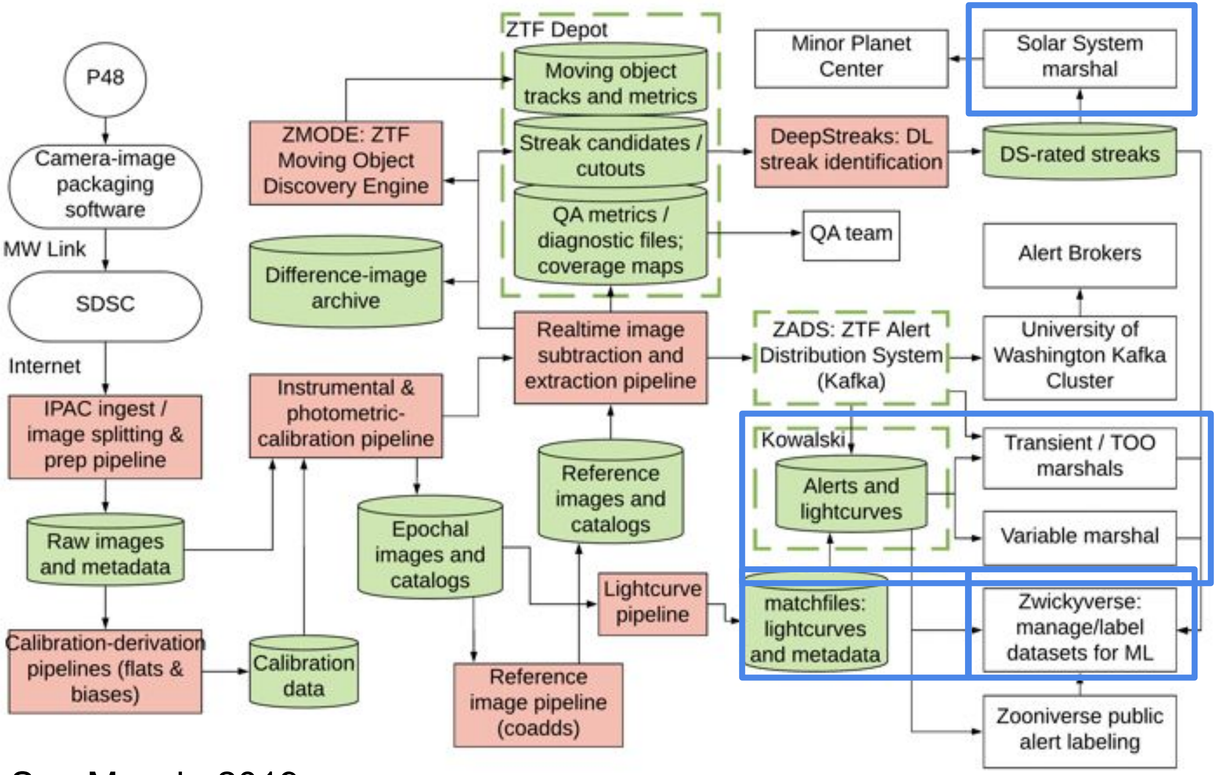
Guy Nir

Kyung Min Shin

Leo Singer

Stéfan van der Walt

# ZTF: data/processing flow



P48

Camera-image packaging software

MW Link

SDSC

Internet

IPAC ingest / image splitting & prep pipeline

Raw images and metadata

Calibration-derivation pipelines (flats & biases)

Calibration data

ZMODE: ZTF Moving Object Discovery Engine

Difference-image archive

Instrumental & photometric-calibration pipeline

Epochal images and catalogs

ZTF Depot

Moving object tracks and metrics

Streak candidates / cutouts

QA metrics / diagnostic files; coverage maps

Realtime image subtraction and extraction pipeline

Reference images and catalogs

Lightcurve pipeline

Reference image pipeline (coadds)

Minor Planet Center

DeepStreaks: DL streak identification

QA team

ZADS: ZTF Alert Distribution System (Kafka)

Kowalski

Alerts and lightcurves

matchfiles: lightcurves and metadata

Solar System marshal

DS-rated streaks

Alert Brokers

University of Washington Kafka Cluster

Transient / TOO marshals

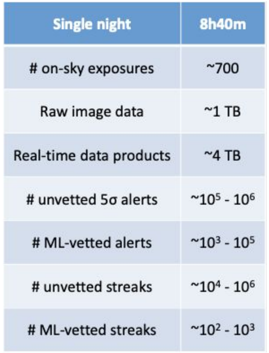
Variable marshal

Zwickyverse: manage/label datasets for ML

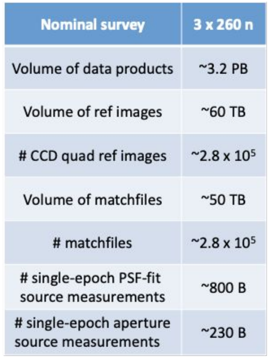
Zooniverse public alert labeling

See Masci+ 2019

ZTF acts as a discovery engine. Discoveries are followed-up using a wide range of instruments (including ZTF itself)



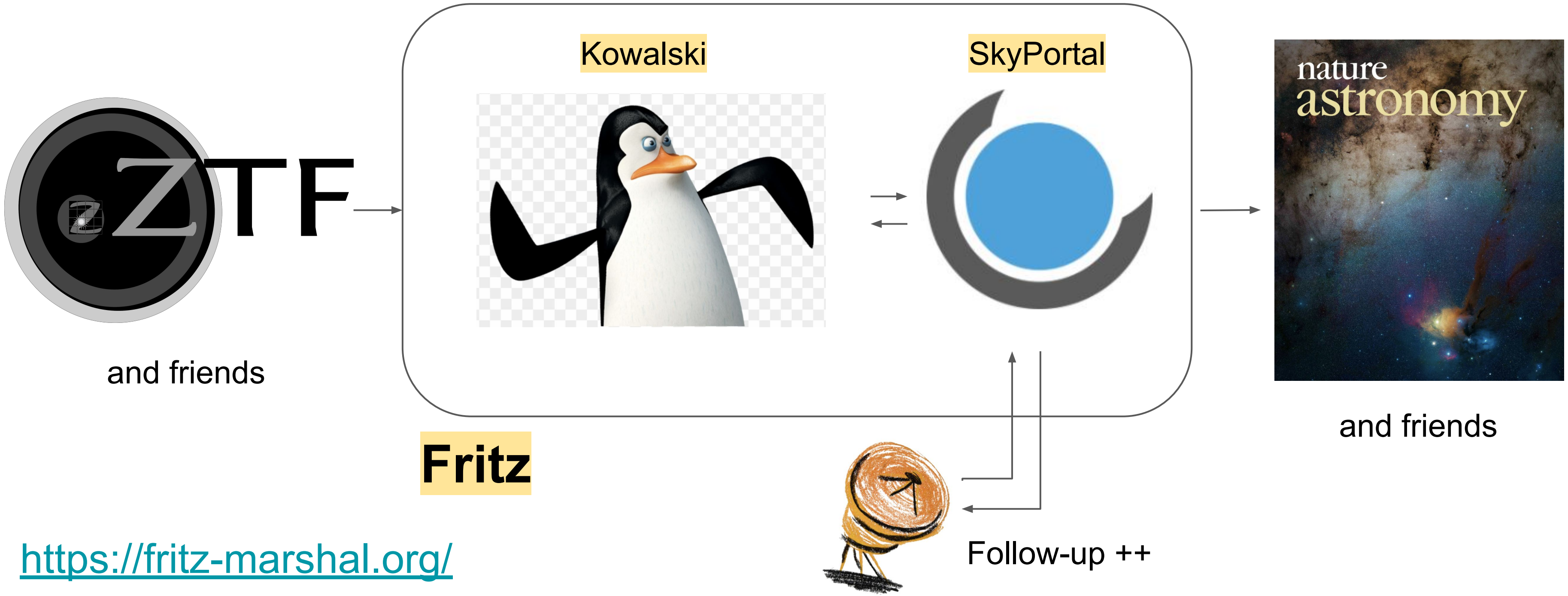
| Single night | 8h40m |
|---|---|
| # on-sky exposures | ~700 |
| Raw image data | ~1 TB |
| Real-time data products | ~4 TB |
| # unvetted 5σ alerts | $\sim 10^5 - 10^6$ |
| # ML-vetted alerts | $\sim 10^3 - 10^5$ |
| # unvetted streaks | $\sim 10^4 - 10^6$ |
| # ML-vetted streaks | $\sim 10^2 - 10^3$ |



| Nominal survey | 3 x 260 n |
|---|---|
| Volume of data products | ~3.2 PB |
| Volume of ref images | ~60 TB |
| # CCD quad ref images | $\sim 2.8 \times 10^5$ |
| Volume of matchfiles | ~50 TB |
| # matchfiles | $\sim 2.8 \times 10^5$ |
| # single-epoch PSF-fit source measurements | ~800 B |
| # single-epoch aperture source measurements | ~230 B |

# Fritz: Schematic overview

Observe → Mine/Discover → Study and Characterize → Profit!1



Kowalski

SkyPortal

and friends

Fritz

and friends

Follow-up ++

https://fritz-marshal.org/

# Fritz: dev

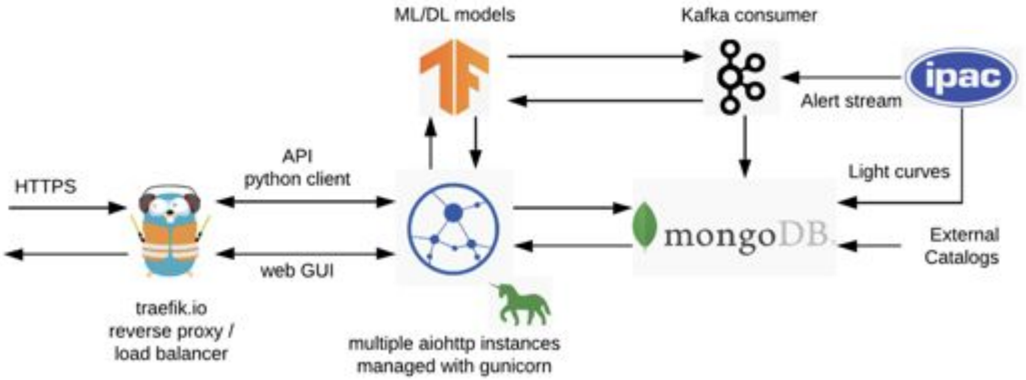
- Clubhouse:
  - https://app.clubhouse.io/astromarshal
- Github:
  - https://github.com/fritz-marshal/fritz
  - https://github.com/skyportal/skyportal
  - https://github.com/dmitryduev/kowalski
- Developer guidelines:
  - https://fritz-marshal.org/doc/developer.html

# Fritz: features

- Open source (free to use, modify, and distribute)
- Powerful alert stream enhancement/filtering capabilities
- Extendible & scalable design: async Python backends, React/Redux frontend
- Rich APIs for machine usage
- Fine-grained access control
- Authentication via Google (or other OAuth)
- Real-time Slack-like messaging, notifications
- Rich visualization capabilities
- Follow-up management
- Distributed computation via Dask
- Docker compose or Kubernetes deployment
- Well-tested, extensive docs, CI/CD

# Kowalski: a tool for TDA

- A sink for the ZTF alert stream
- ZTF light curves
- >30 external catalogs
- ML models
- Cross-matches



Beefy machines (50TB SSDs, 1TB memory) for local deployment

https://github.com/dmitryduev/kowalski

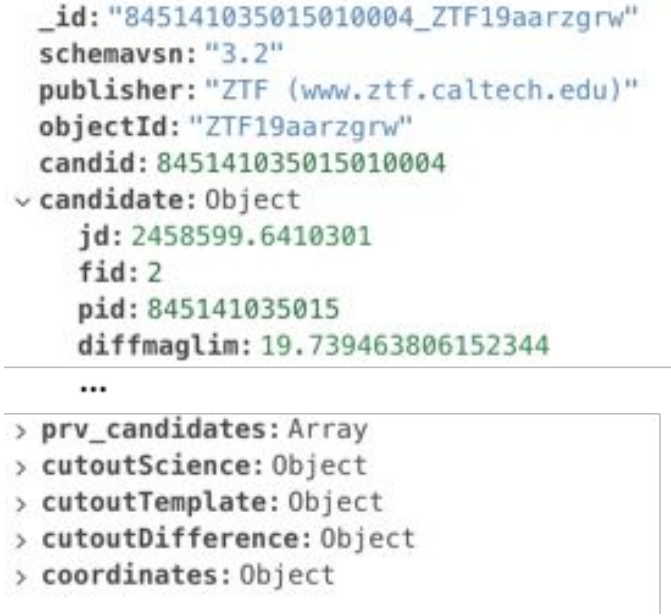
# Kowalski: some tech details

- traefik.io reverse proxy/load balancer
- aiohttp-based API layer
  - asyncio event loop with futures scheduling serves as a queue; simple, fast and robust
  - JWT tokens
- MongoDB
  - NoSQL: BSON documents in collections. Natural to store alerts (AVRO directly translates into BSON; evolving schema) and light curves (1 read per source to extract data points)
  - (Compound) indexes and covered queries for efficient access to most-in-demand data
  - Built-in GeoJSON support with 2D indexes on the sphere
  - MQL: cone and general searches, aggregation pipelines
  - Horizontal scale-out

Containerized; orchestration with docker-compose

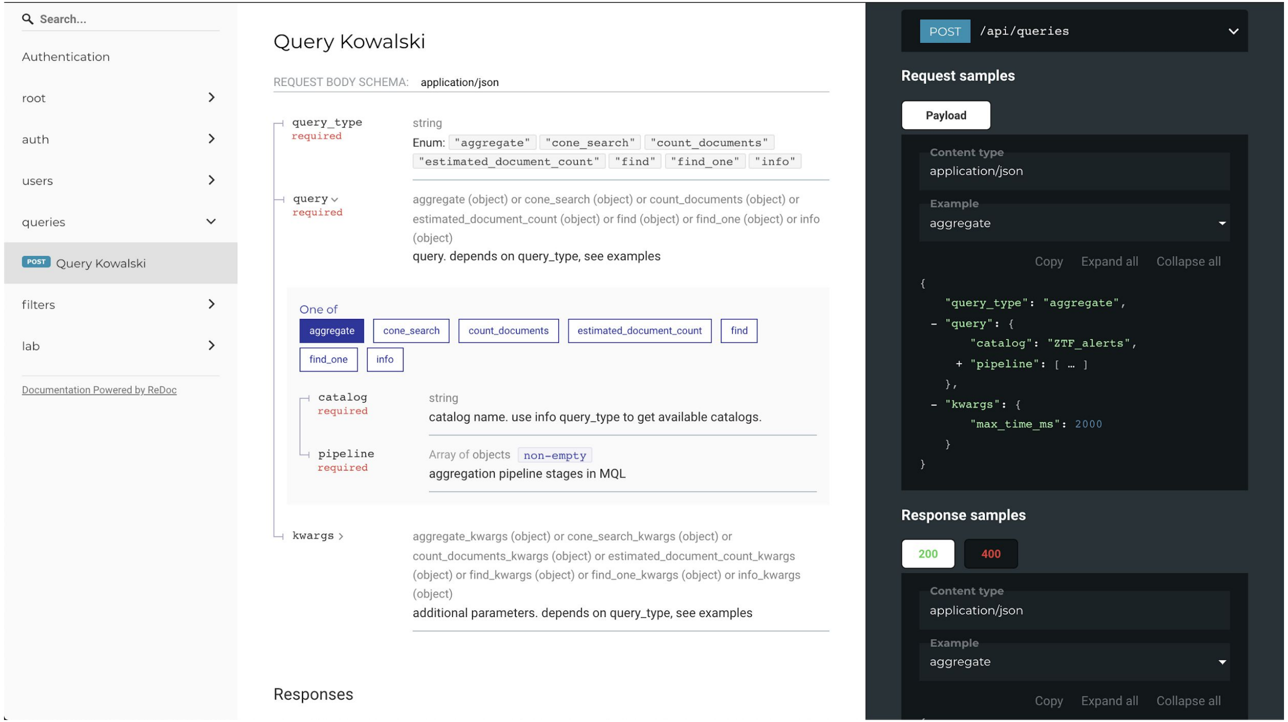
# Kowalski: stats

- 30+ databases (Gaia, PS1, 2MASS…) [~10 TB]
- 70+ active users including several services
- ~100M queries/day typical load
- Indices ~5% total db size
- Direct Kafka alert streams from IPAC
- >310 M ZTF alerts [~10 TB]
- >3.5 B ZTF light curves [~20 TB]



```
 _id: "845141035015010004_ZTF19aarzgrw"
 schemavsn: "3.2"
 publisher: "ZTF (www.ztf.caltech.edu)"
 objectId: "ZTF19aarzgrw"
 candid: 845141035015010004
⌄candidate: Object
   jd: 2458599.6410301
   fid: 2
   pid: 845141035015
   diffmaglim: 19.739463806152344
   ...
›prv_candidates: Array
›cutoutScience: Object
›cutoutTemplate: Object
›cutoutDifference: Object
›coordinates: Object
```

# Kowalski: API + python client



Query Kowalski

REQUEST BODY SCHEMA: application/json

query_type required — string
Enum: "aggregate" "cone_search" "count_documents" "estimated_document_count" "find" "find_one" "info"

query required — aggregate (object) or cone_search (object) or count_documents (object) or estimated_document_count (object) or find (object) or find_one (object) or info (object)
query. depends on query_type, see examples

One of

catalog required — string
catalog name. use info query_type to get available catalogs.

pipeline required — Array of objects non-empty
aggregation pipeline stages in MQL

kwargs — aggregate_kwargs (object) or cone_search_kwargs (object) or count_documents_kwargs (object) or estimated_document_count_kwargs (object) or find_kwargs (object) or find_one_kwargs (object) or info_kwargs (object)
additional parameters. depends on query_type, see examples

Responses

POST /api/queries

Request samples

Content type
application/json

Example
aggregate

```
{
    "query_type": "aggregate",
  - "query": {
        "catalog": "ZTF_alerts",
      + "pipeline": [ … ]
    },
  - "kwargs": {
        "max_time_ms": 2000
    }
}
```

Response samples

Content type
application/json

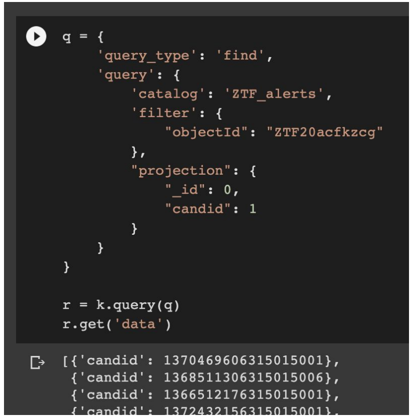
Example
aggregate

## OpenAPI specs:

https://kowalski.caltech.edu/docs/api

## Python client:

https://github.com/dmitryduev/penquins



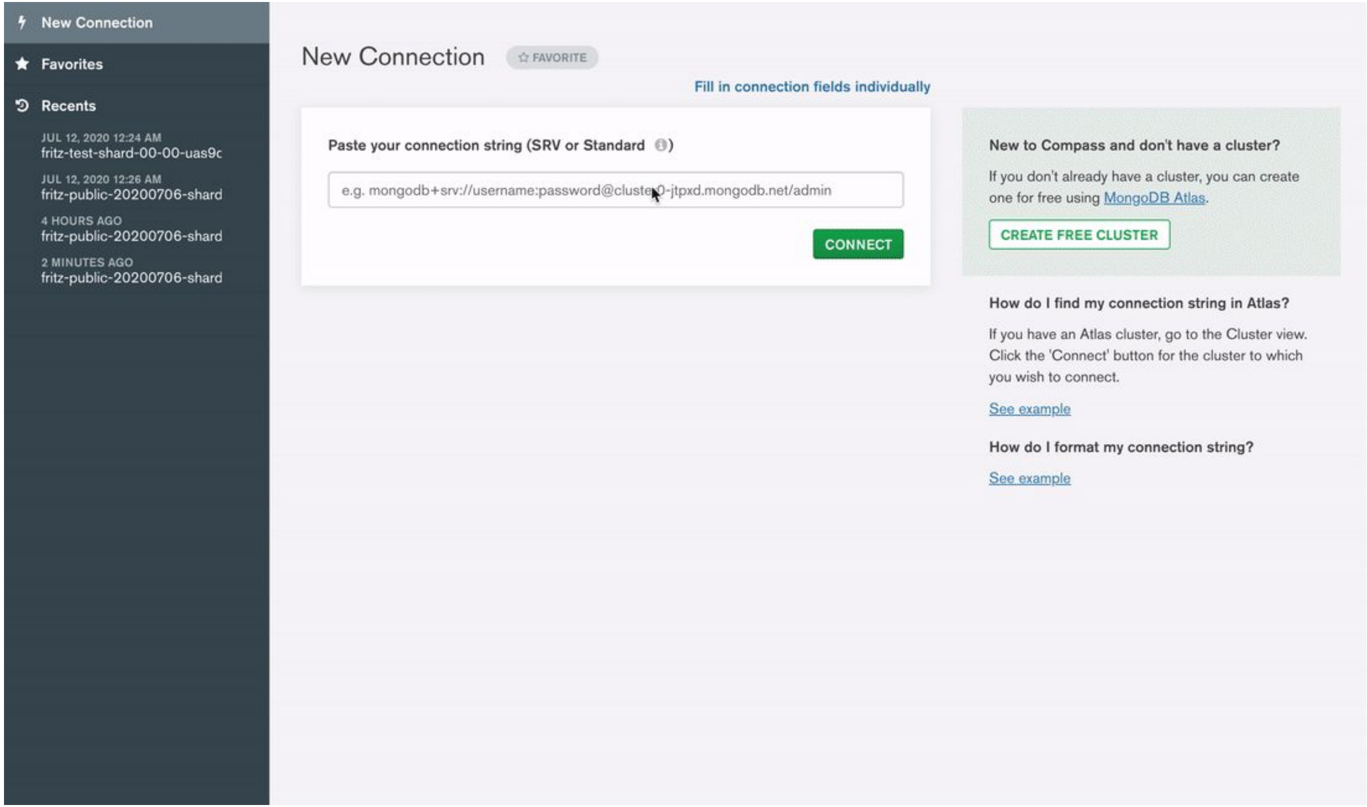
```
q = {
    'query_type': 'find',
    'query': {
        'catalog': 'ZTF_alerts',
        'filter': {
            "objectId": "ZTF20acfkzcg"
        },
        "projection": {
            "_id": 0,
            "candid": 1
        }
    }
}

r = k.query(q)
r.get('data')
```

```
[{'candid': 1370469606315015001},
 {'candid': 1368511306315015006},
 {'candid': 1366512176315015001},
 {'candid': 1372432156315015001},
```

# Kowalski as a part of Fritz

- Alert stream filters
  - Incoming alerts are cross-matched against the external catalogs
  - ML models are executed on alerts
  - Freshly-ingested *enhanced* alerts are passed through user-defined filters and posted to SkyPortal
  - Filters are implemented as MongoDB aggregation pipelines
  - https://fritz-marshal.org/doc/user_guide.html#alert-filters-in-fritz
- Queryable store of ZTF alerts and light curves++

# Kowalski: alert stream filters



JUL 12, 2020 12:24 AM
fritz-test-shard-00-00-uas9c

JUL 12, 2020 12:26 AM
fritz-public-20200706-shard

4 HOURS AGO
fritz-public-20200706-shard

2 MINUTES AGO
fritz-public-20200706-shard

Paste your connection string (SRV or Standard ⓘ)

e.g. mongodb+srv://username:password@cluster0-jtpxd.mongodb.net/admin

New to Compass and don't have a cluster?

If you don't already have a cluster, you can create one for free using MongoDB Atlas.

How do I find my connection string in Atlas?

If you have an Atlas cluster, go to the Cluster view. Click the 'Connect' button for the cluster to which you wish to connect.

See example

How do I format my connection string?

See example

Public databases for filter design and debugging

https://fritz-marshal.org/doc

# SkyPortal: vision

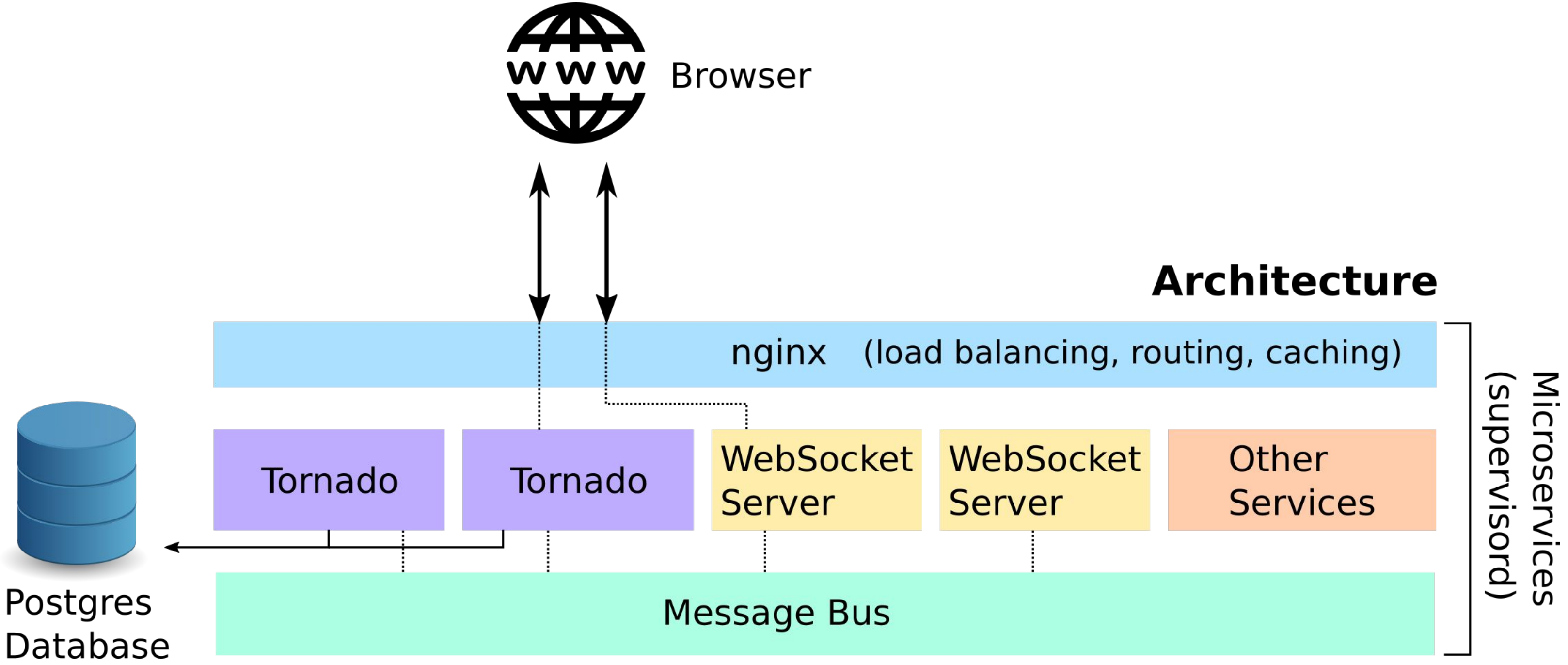
A **data portal**, inspired by the PTF/GROWTH marshal, utilizing secure **modern web technologies**, **scales** effectively, and is **highly customizable** and extensible so that it can be applied to various astronomy problems related to ZTF, LSST, and other surveys.

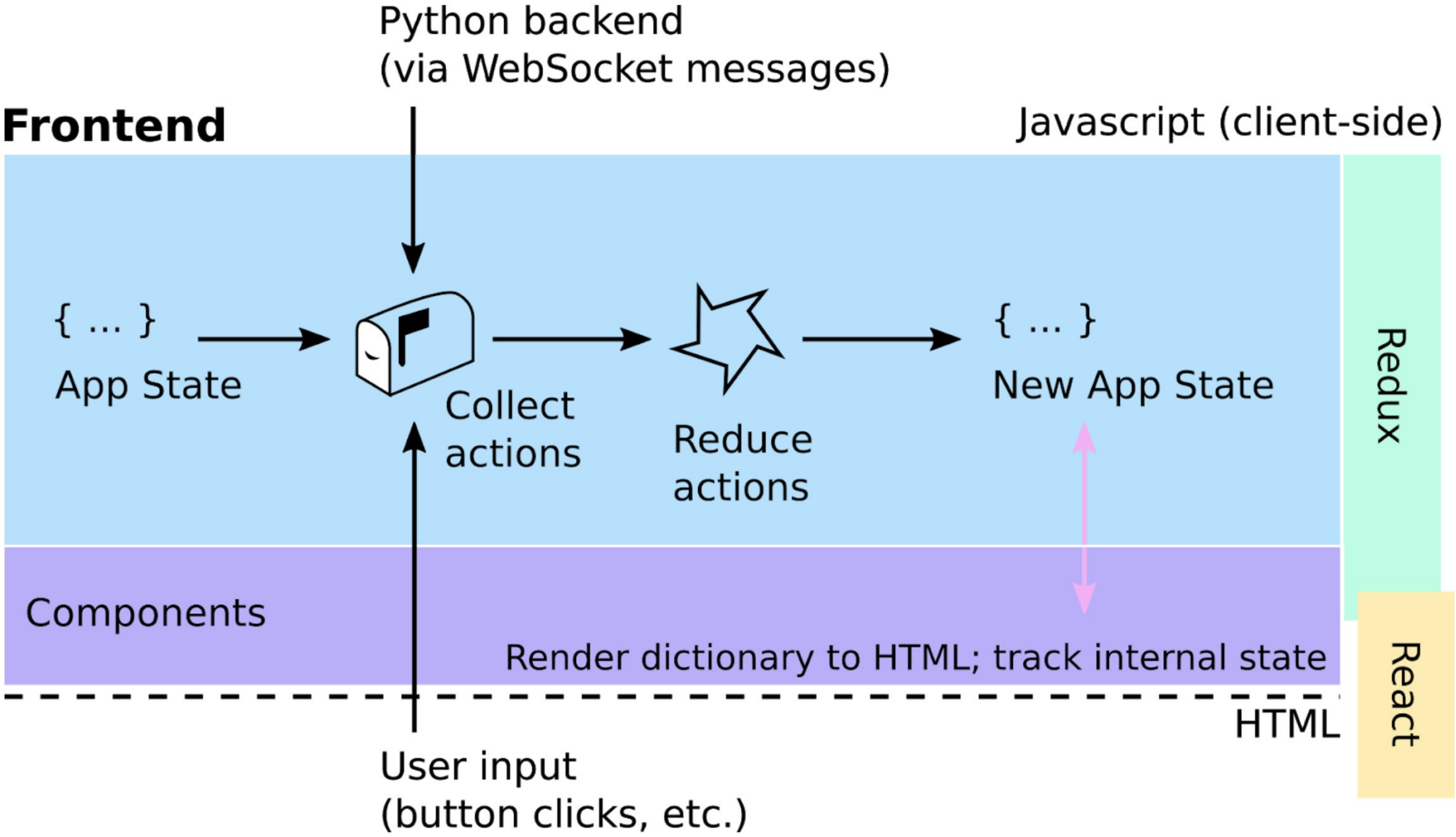
https://skyportal.io

# SkyPortal: architecture

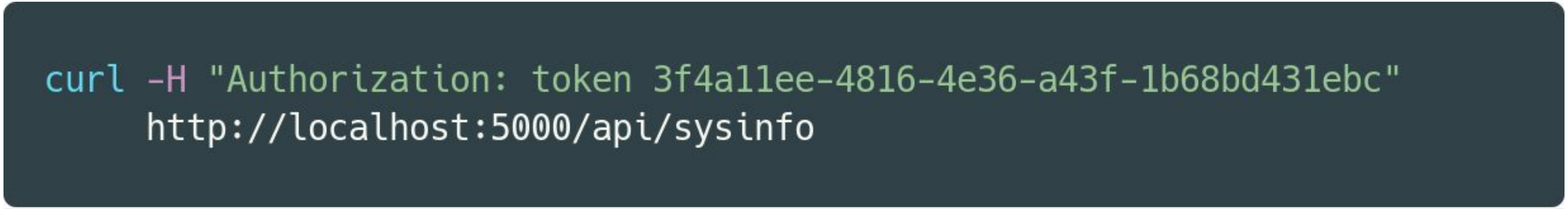
cesium-ml.org

Baselayer

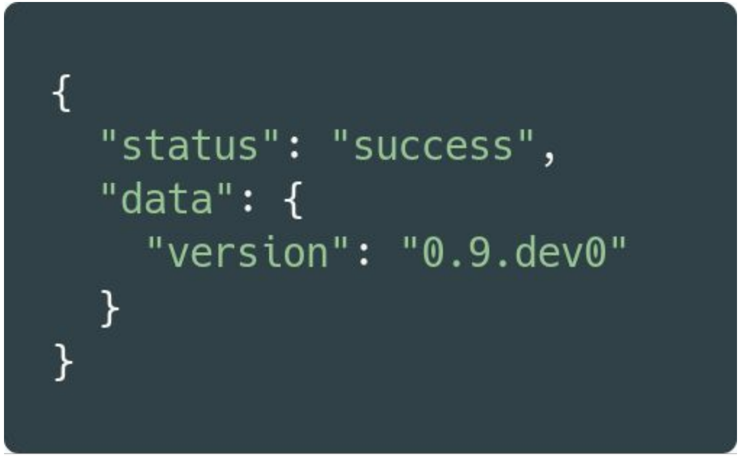



Python backend
(via WebSocket messages)
Frontend
Javascript (client-side)
{ ... }
App State
Collect actions
Reduce actions
{ ... }
New App State
Redux
Components
Render dictionary to HTML; track internal state
HTML
React
User input
(button clicks, etc.)

# SkyPortal: example API request

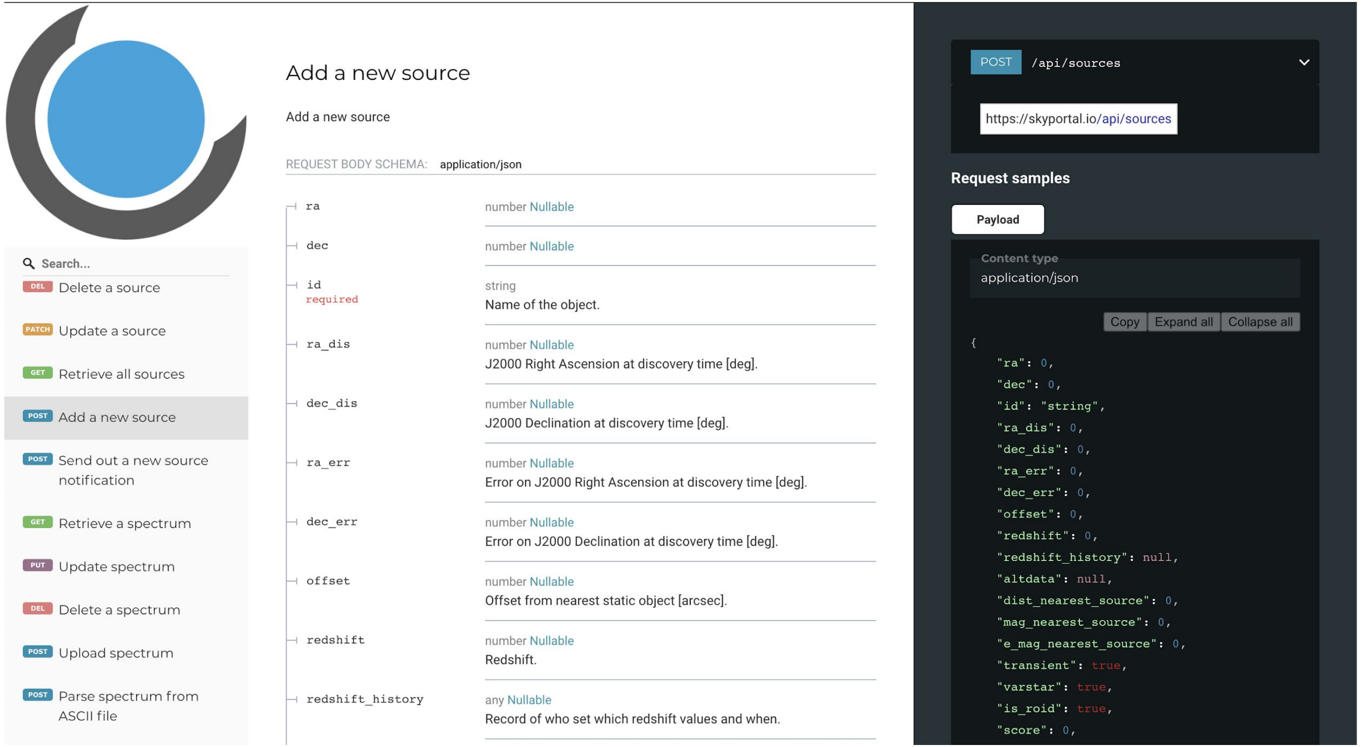


```
curl -H "Authorization: token 3f4a11ee-4816-4e36-a43f-1b68bd431ebc"
     http://localhost:5000/api/sysinfo
```



```
{
  "status": "success",
  "data": {
    "version": "0.9.dev0"
  }
}
```

# SkyPortal: API docs



Search...

- DEL Delete a source
- PATCH Update a source
- GET Retrieve all sources
- POST Add a new source
- POST Send out a new source notification
- GET Retrieve a spectrum
- PUT Update spectrum
- DEL Delete a spectrum
- POST Upload spectrum
- POST Parse spectrum from ASCII file

## Add a new source

Add a new source

REQUEST BODY SCHEMA: application/json

| | |
|---|---|
| ra | number Nullable |
| dec | number Nullable |
| id<br>required | string<br>Name of the object. |
| ra_dis | number Nullable<br>J2000 Right Ascension at discovery time [deg]. |
| dec_dis | number Nullable<br>J2000 Declination at discovery time [deg]. |
| ra_err | number Nullable<br>Error on J2000 Right Ascension at discovery time [deg]. |
| dec_err | number Nullable<br>Error on J2000 Declination at discovery time [deg]. |
| offset | number Nullable<br>Offset from nearest static object [arcsec]. |
| redshift | number Nullable<br>Redshift. |
| redshift_history | any Nullable<br>Record of who set which redshift values and when. |

POST /api/sources

https://skyportal.io/api/sources

**Request samples**

Payload

Content type
application/json

```
{
    "ra": 0,
    "dec": 0,
    "id": "string",
    "ra_dis": 0,
    "dec_dis": 0,
    "ra_err": 0,
    "dec_err": 0,
    "offset": 0,
    "redshift": 0,
    "redshift_history": null,
    "altdata": null,
    "dist_nearest_source": 0,
    "mag_nearest_source": 0,
    "e_mag_nearest_source": 0,
    "transient": true,
    "varstar": true,
    "is_roid": true,
    "score": 0,
```

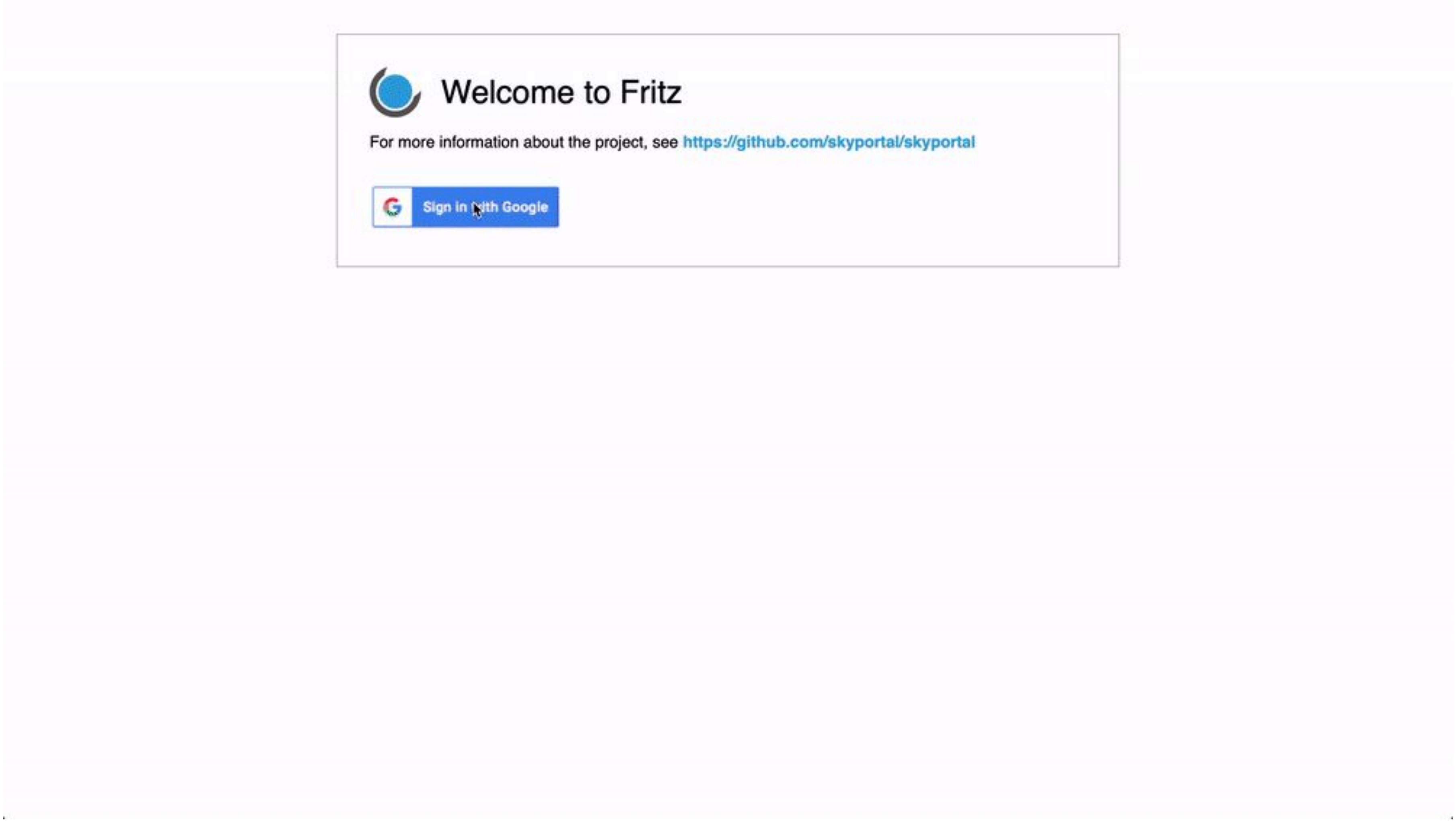
API-first system

https://skyportal.io/docs/api.html

# Fritz: misc

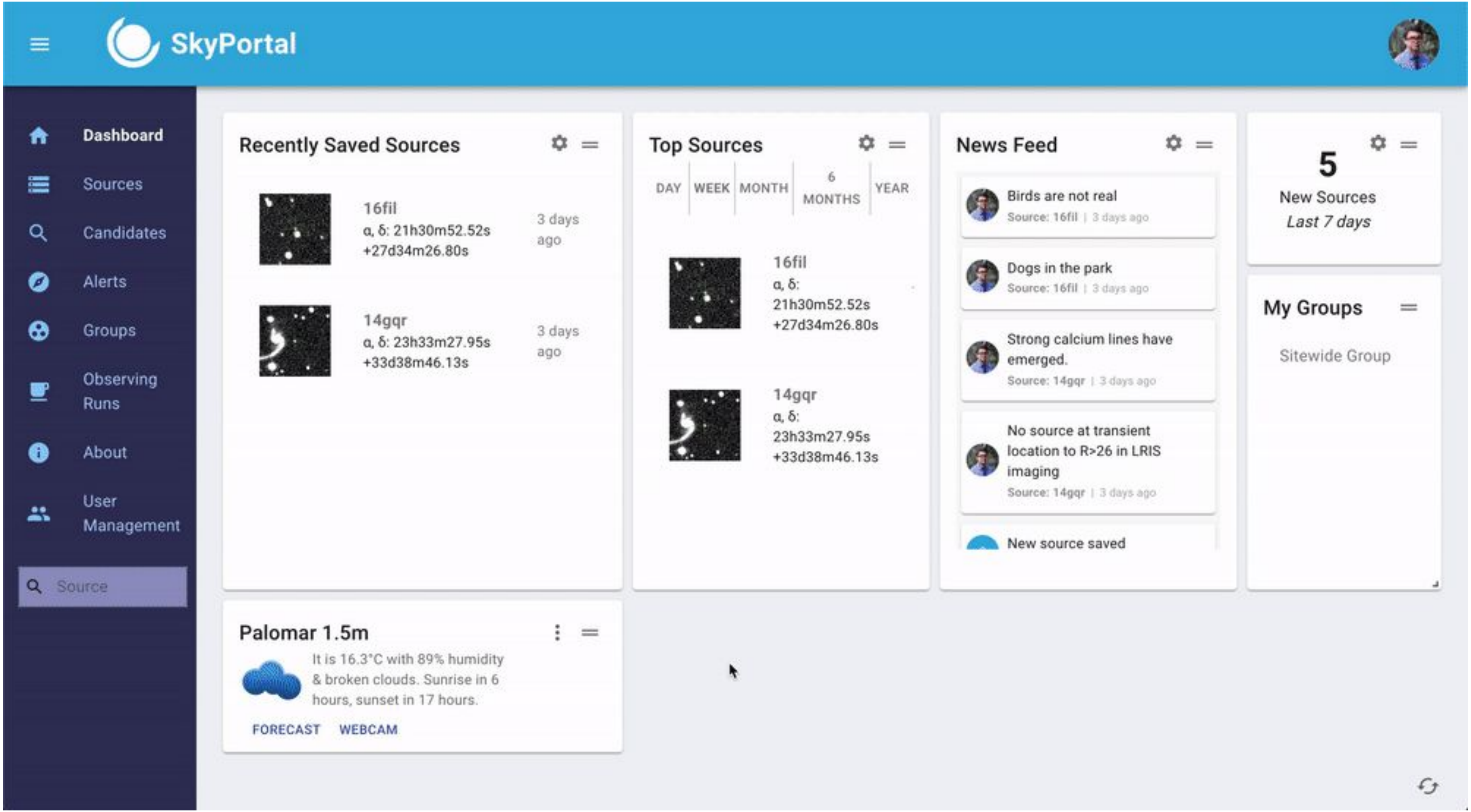
- Integrates and extends Kowalski & SkyPortal
  - ZTF-specific API endpoints and frontend components
- Deployments:
  - Local: orchestrated with docker-compose for beta-testers
  - Staging: GCP, orchestrated with k8s
  - Production: hybrid, GCP+local, orchestrated with docker-compose and k8s
- Current focus on transient science
  - Variable and Solar system science cases coming soon

**HUGE THANKS TO THE BETA-TESTERS!!!**



Welcome to Fritz

For more information about the project, see https://github.com/skyportal/skyportal



Recently Saved Sources

16fil
α, δ: 21h30m52.52s +27d34m26.80s
3 days ago

14gqr
α, δ: 23h33m27.95s +33d38m46.13s
3 days ago

Top Sources

16fil
α, δ: 21h30m52.52s +27d34m26.80s

14gqr
α, δ: 23h33m27.95s +33d38m46.13s

News Feed

Birds are not real
Source: 16fil | 3 days ago

Dogs in the park
Source: 16fil | 3 days ago

Strong calcium lines have emerged.
Source: 14gqr | 3 days ago

No source at transient location to R>26 in LRIS imaging
Source: 14gqr | 3 days ago

New source saved

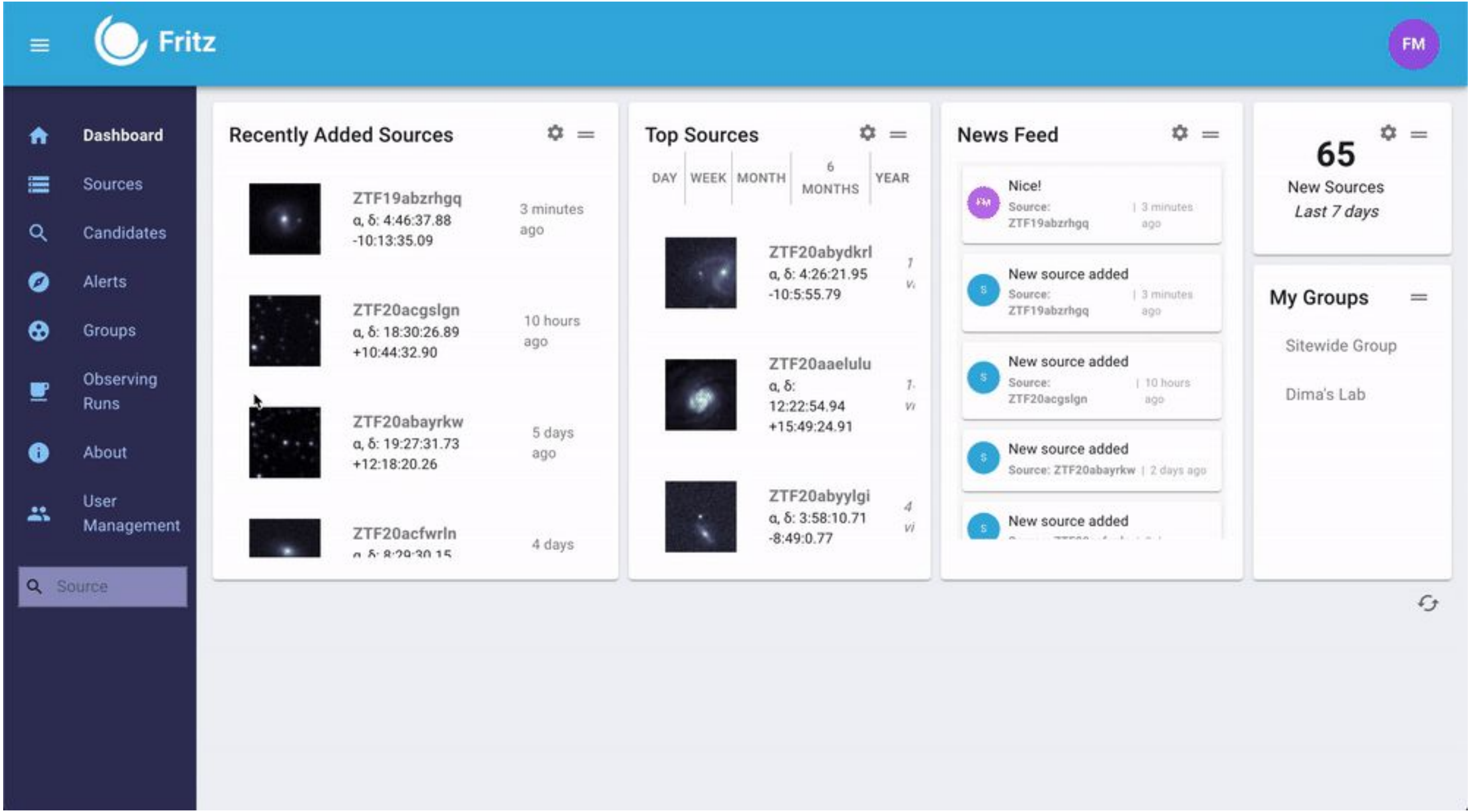
5
New Sources
*Last 7 days*

My Groups

Sitewide Group

Palomar 1.5m

It is 16.3°C with 89% humidity & broken clouds. Sunrise in 6 hours, sunset in 17 hours.



## Recently Added Sources

ZTF19abzrhgq
α, δ: 4:46:37.88 -10:13:35.09
3 minutes ago

ZTF20acgslgn
α, δ: 18:30:26.89 +10:44:32.90
10 hours ago

ZTF20abayrkw
α, δ: 19:27:31.73 +12:18:20.26
5 days ago

ZTF20acfwrln
α, δ: 8:29:30.15
4 days

## Top Sources

ZTF20abydkrl
α, δ: 4:26:21.95 -10:5:55.79

ZTF20aaelulu
α, δ: 12:22:54.94 +15:49:24.91

ZTF20abyylgi
α, δ: 3:58:10.71 -8:49:0.77

## News Feed

Nice!
Source: ZTF19abzrhgq | 3 minutes ago

New source added
Source: ZTF19abzrhgq | 3 minutes ago

New source added
Source: ZTF20acgslgn | 10 hours ago

New source added
Source: ZTF20abayrkw | 2 days ago

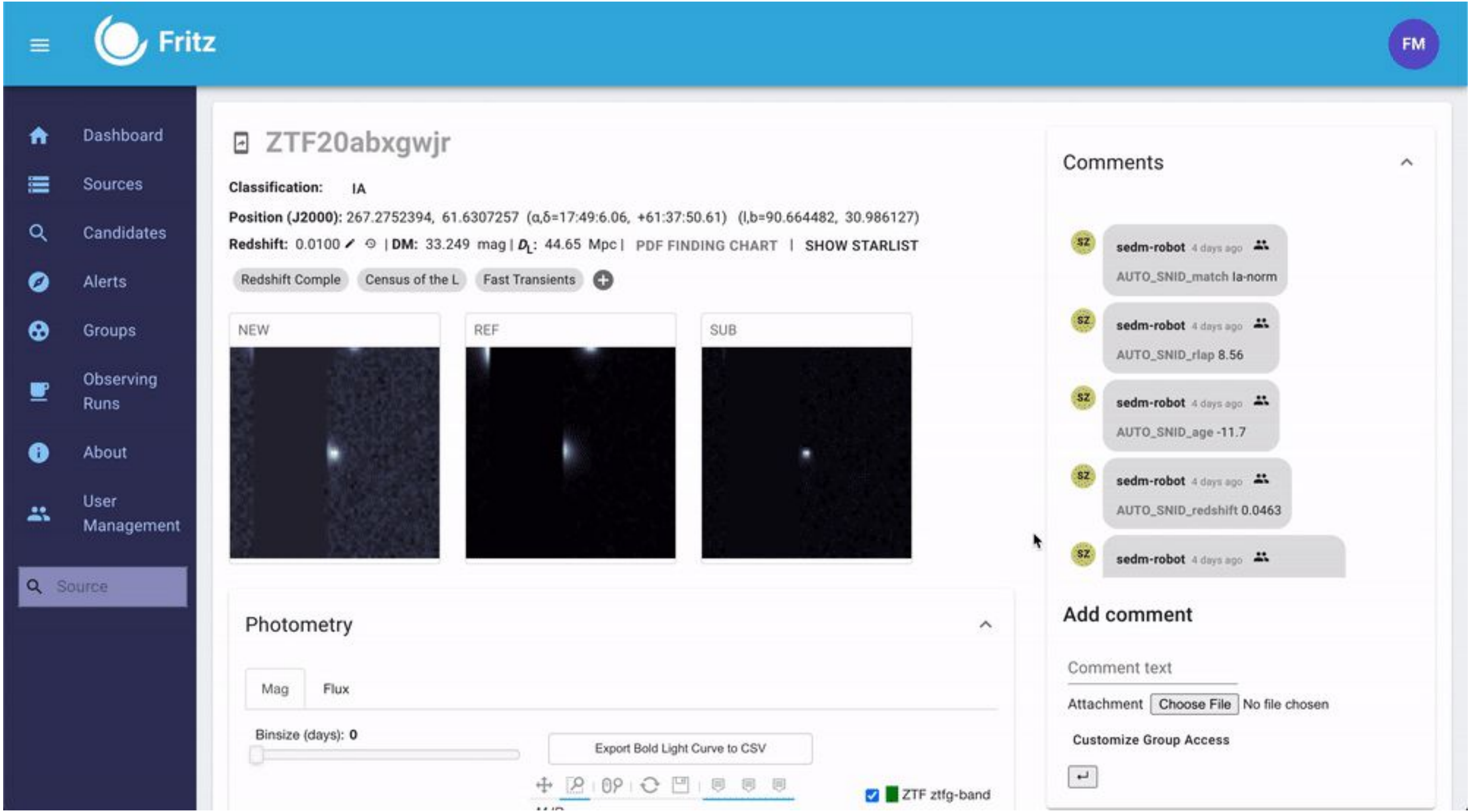
New source added

65
New Sources
*Last 7 days*

## My Groups

Sitewide Group

Dima's Lab



ZTF20abxgwjr

**Classification:** IA

**Position (J2000):** 267.2752394, 61.6307257 (α,δ=17:49:6.06, +61:37:50.61) (l,b=90.664482, 30.986127)

**Redshift:** 0.0100 | **DM:** 33.249 mag | $D_L$: 44.65 Mpc | PDF FINDING CHART | SHOW STARLIST

Redshift Comple | Census of the L | Fast Transients

**sedm-robot** 4 days ago
AUTO_SNID_match Ia-norm

**sedm-robot** 4 days ago
AUTO_SNID_rlap 8.56

**sedm-robot** 4 days ago
AUTO_SNID_age -11.7

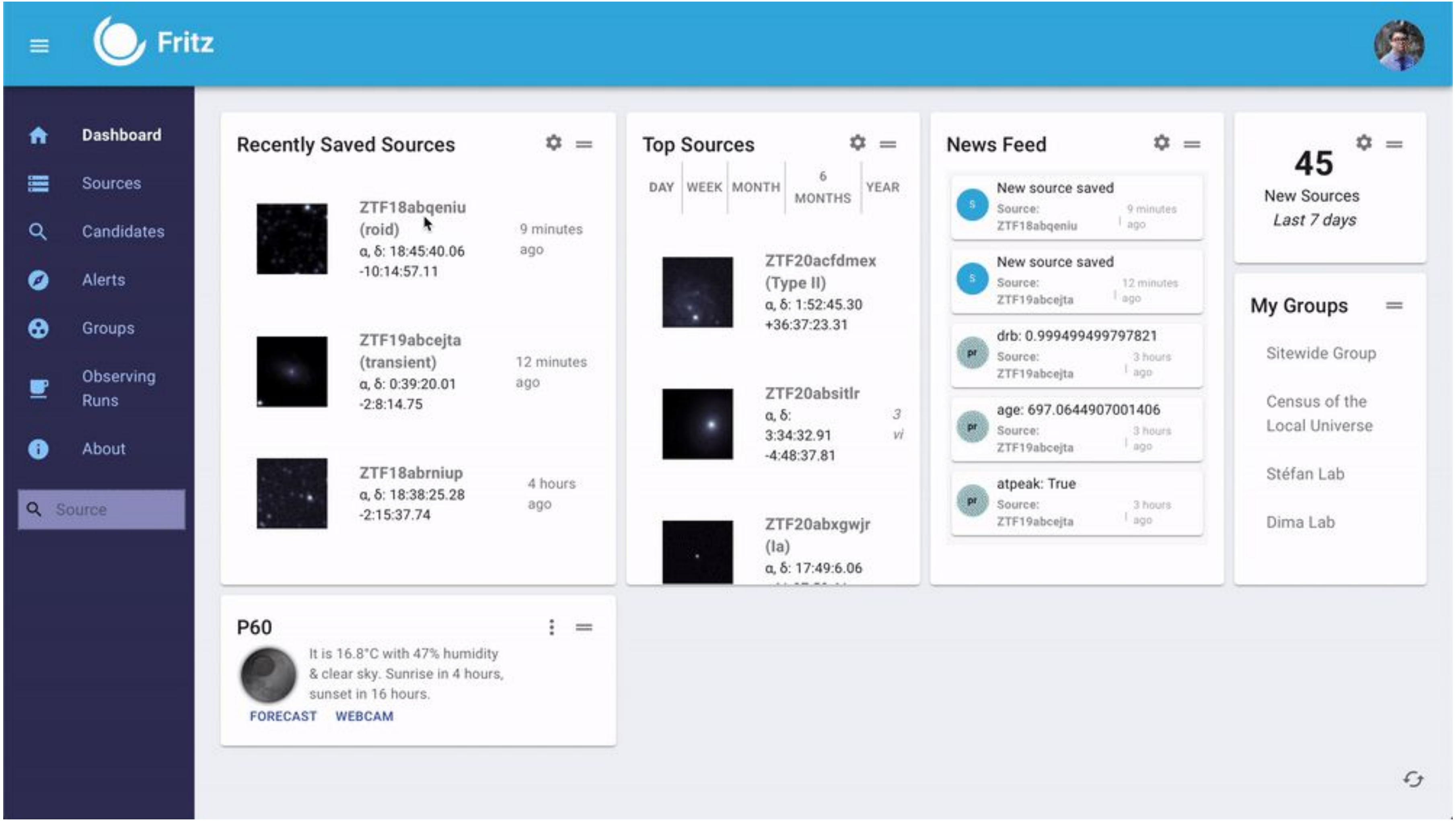
**sedm-robot** 4 days ago
AUTO_SNID_redshift 0.0463

**sedm-robot** 4 days ago

Attachment: No file chosen

Binsize (days): **0**

☑ ZTF ztfg-band



Recently Saved Sources

ZTF18abqeniu (roid)
α, δ: 18:45:40.06 -10:14:57.11
9 minutes ago

ZTF19abcejta (transient)
α, δ: 0:39:20.01 -2:8:14.75
12 minutes ago

ZTF18abrniup
α, δ: 18:38:25.28 -2:15:37.74
4 hours ago

Top Sources

ZTF20acfdmex (Type II)
α, δ: 1:52:45.30 +36:37:23.31

ZTF20absitlr
α, δ: 3:34:32.91 -4:48:37.81
3 vi

ZTF20abxgwjr (Ia)
α, δ: 17:49:6.06

News Feed

New source saved
Source: ZTF18abqeniu
9 minutes ago

New source saved
Source: ZTF19abcejta
12 minutes ago

drb: 0.999499499797821
Source: ZTF19abcejta
3 hours ago

age: 697.0644907001406
Source: ZTF19abcejta
3 hours ago

atpeak: True
Source: ZTF19abcejta
3 hours ago

45
New Sources
*Last 7 days*

My Groups

Sitewide Group

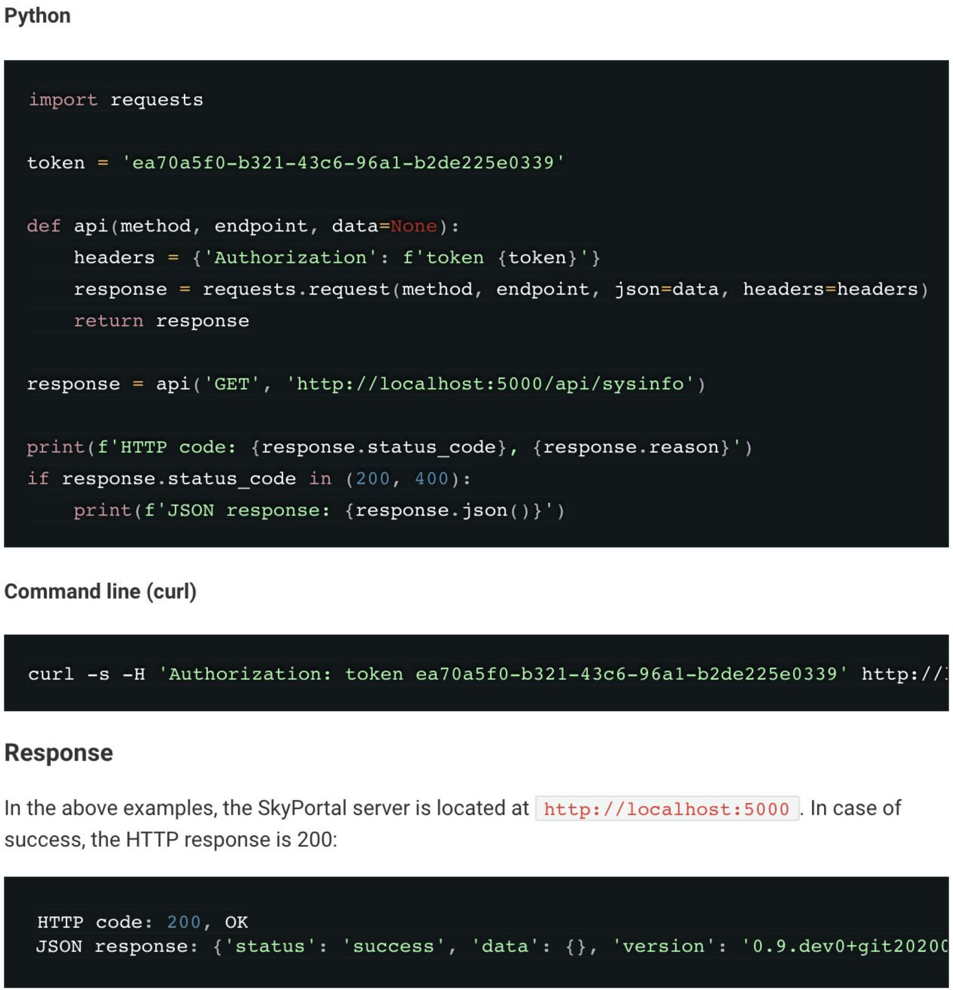
Census of the Local Universe

Stéfan Lab

Dima Lab

P60

It is 16.8°C with 47% humidity & clear sky. Sunrise in 4 hours, sunset in 16 hours.



### Python

```python
import requests

token = 'ea70a5f0-b321-43c6-96a1-b2de225e0339'

def api(method, endpoint, data=None):
    headers = {'Authorization': f'token {token}'}
    response = requests.request(method, endpoint, json=data, headers=headers)
    return response

response = api('GET', 'http://localhost:5000/api/sysinfo')

print(f'HTTP code: {response.status_code}, {response.reason}')
if response.status_code in (200, 400):
    print(f'JSON response: {response.json()}')
```

### Command line (curl)

```
curl -s -H 'Authorization: token ea70a5f0-b321-43c6-96a1-b2de225e0339' http://
```

## Response

In the above examples, the SkyPortal server is located at `http://localhost:5000`. In case of success, the HTTP response is 200:

```
HTTP code: 200, OK
JSON response: {'status': 'success', 'data': {}, 'version': '0.9.dev0+git2020
```

All of these actions (and many, many more) can be carried out programmatically using the API (which is what the frontend is actually doing)