Let's Cache!

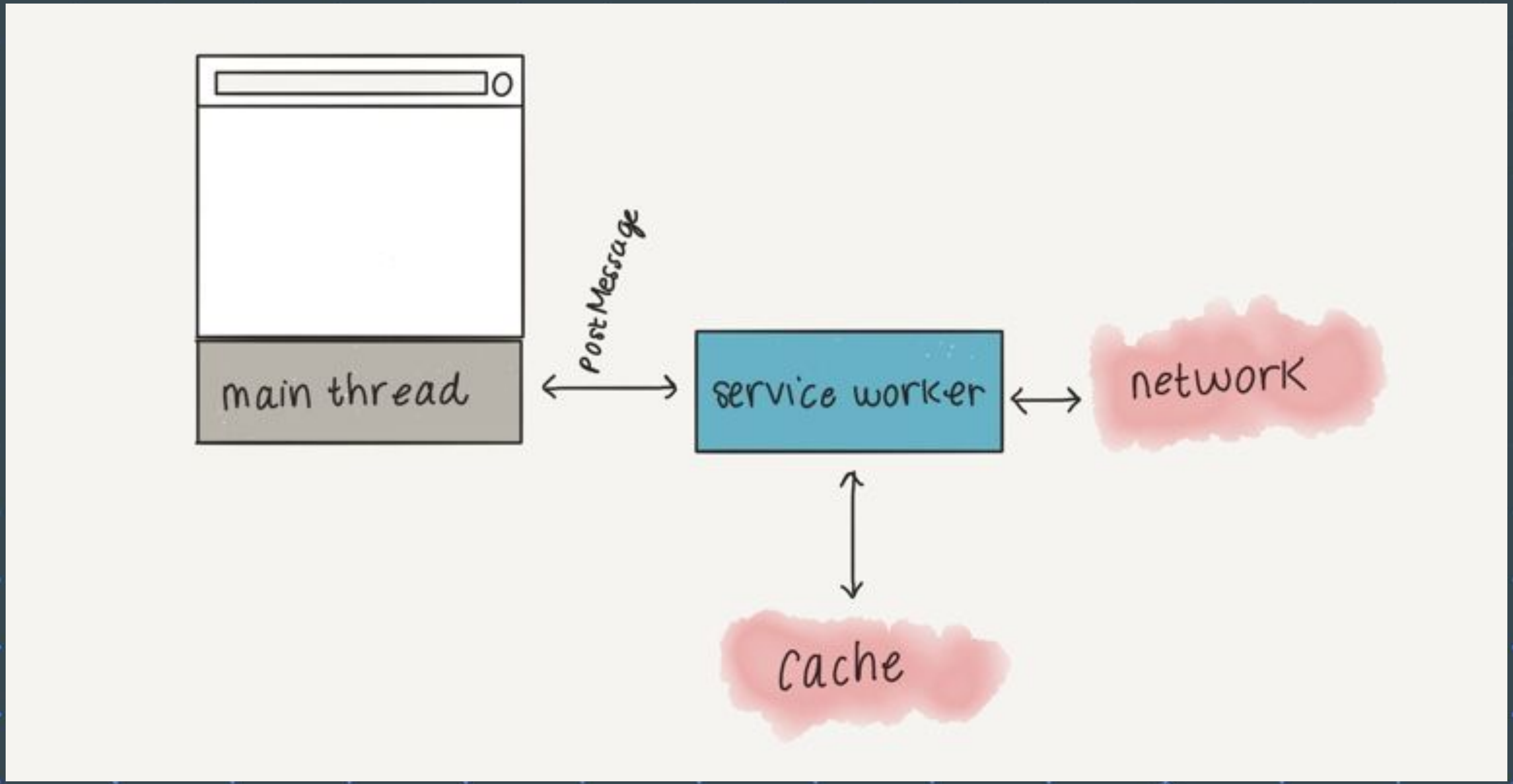
An introduction to service workers

Agenda

- Service worker?
- What do I need this?
- Access Strategies
  - Network only
  - Cache only
  - Cache falling back to network
  - Network falling back to cache

Service Workers?

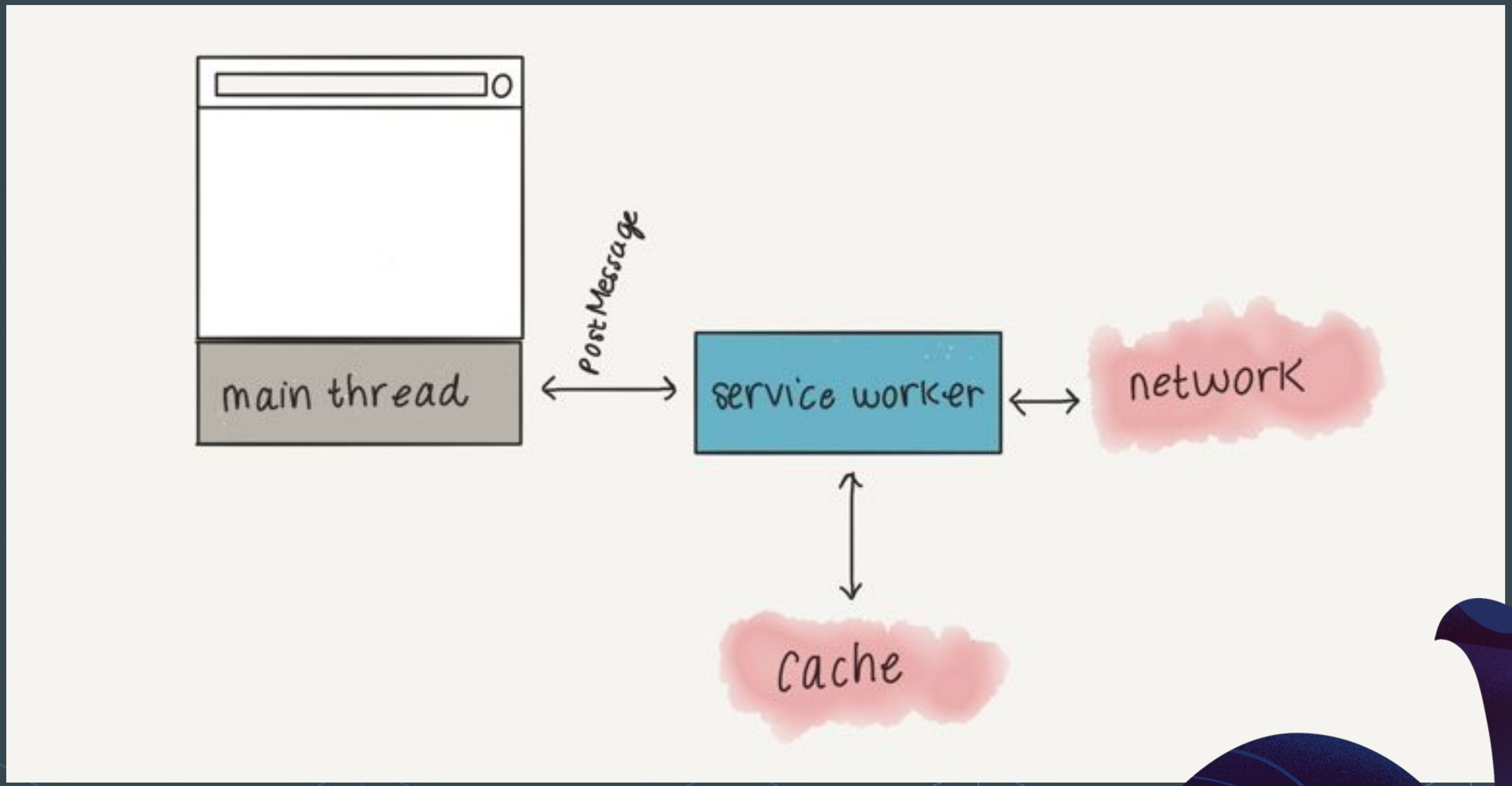
- Scripts that act like proxy servers by sitting between your app, network, and the cache
- Quietly waits for your web app request & jumps into action to intercept registered requests
- Retrieves resources from the browser Cache Storage



Credit: bitsofco

Why do I need this?

- Rich offline experience
  - Progressive Web Apps (PWA)
- Periodic background syncs
- Fast load time
- Reliable and consistent



Credit: bitsofco

Main Actors

Web Application

Network

Cache Storage

Service Worker

WARNING: Plan how you want each api to interact with service worker before implementing or else you might give your users an even worse UX

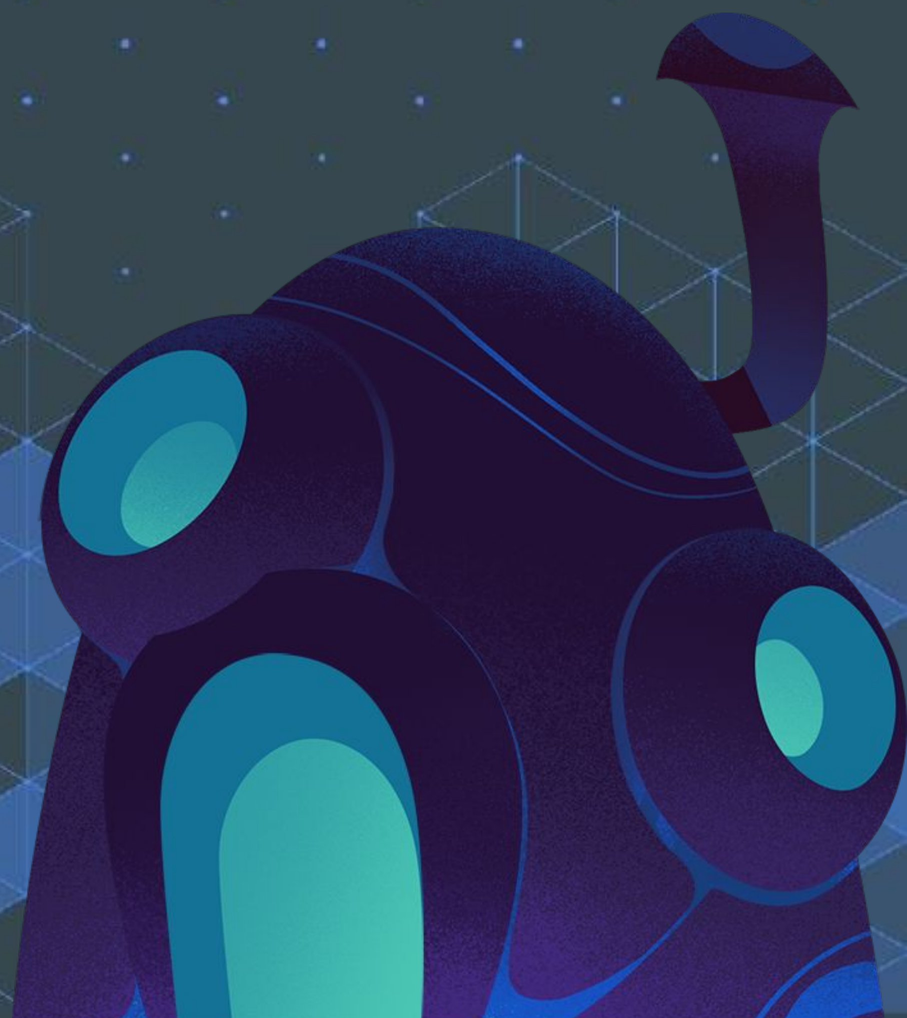
Caching strategies

Network-only

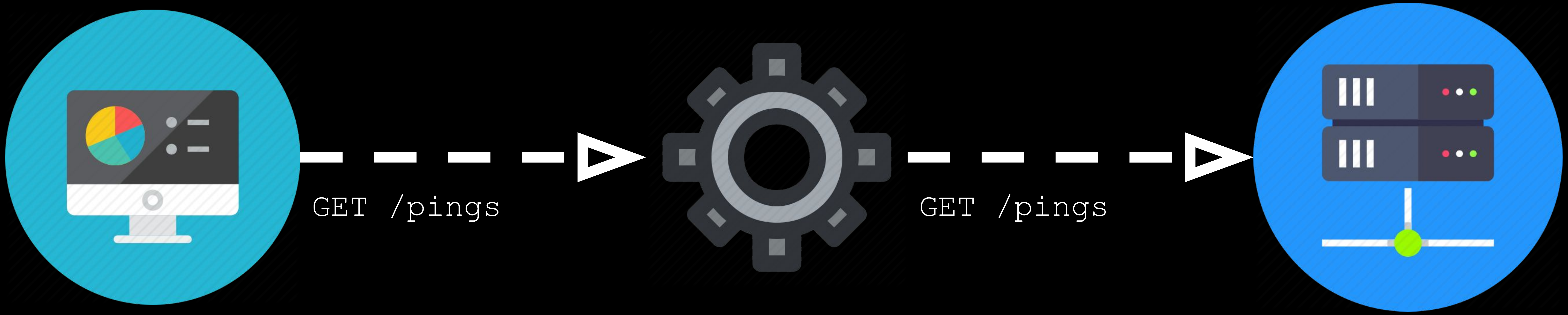
Cache-only

Cache falling back to network

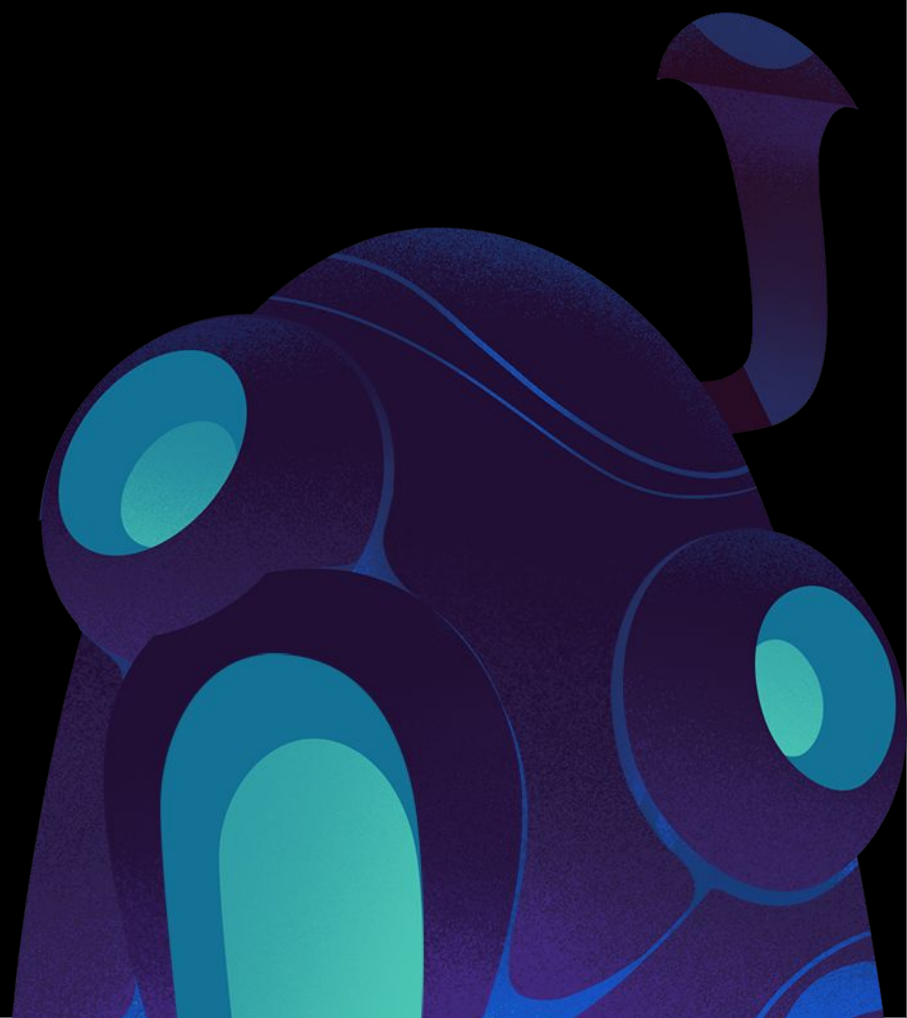
Network falling back to cache



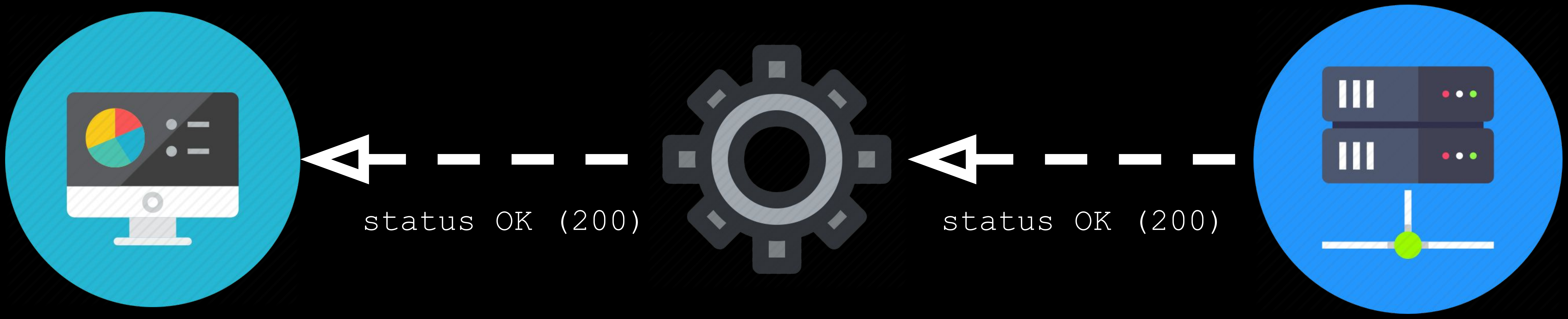
Network only



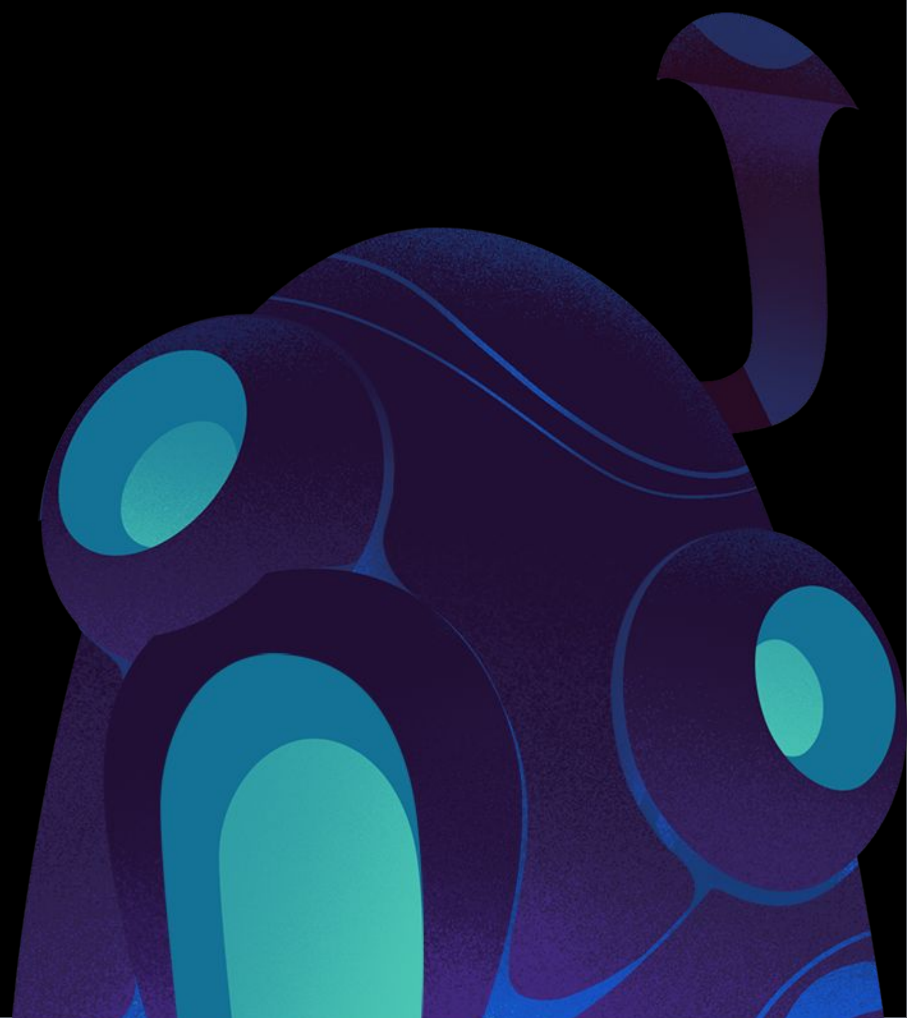
When to use this strategy:
Any data that you would not want persisted in your application offline (i.e analytics pings)



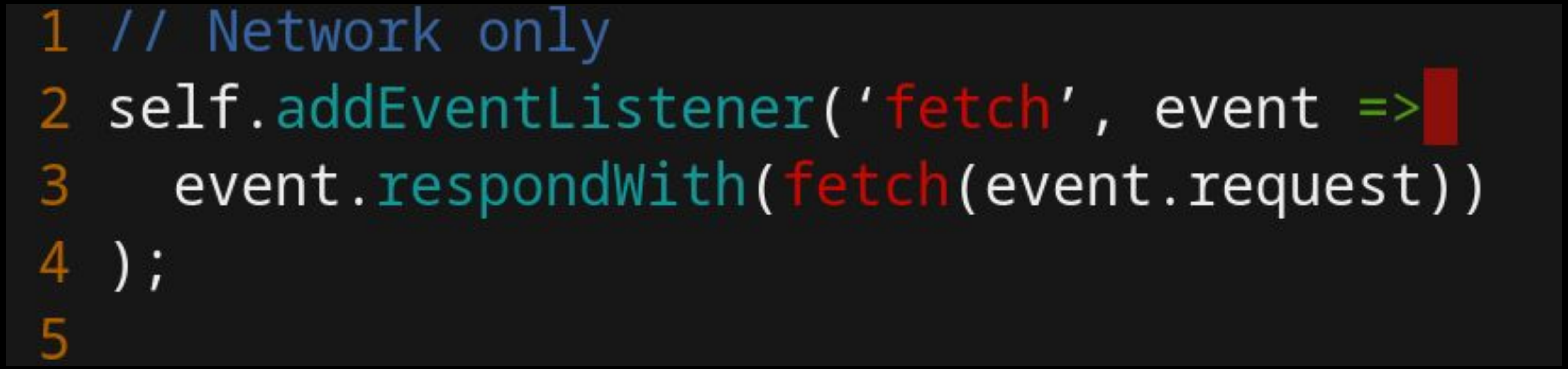
Network only



When to use this strategy:
Any data that you would not want persisted in your application offline (i.e analytics pings)

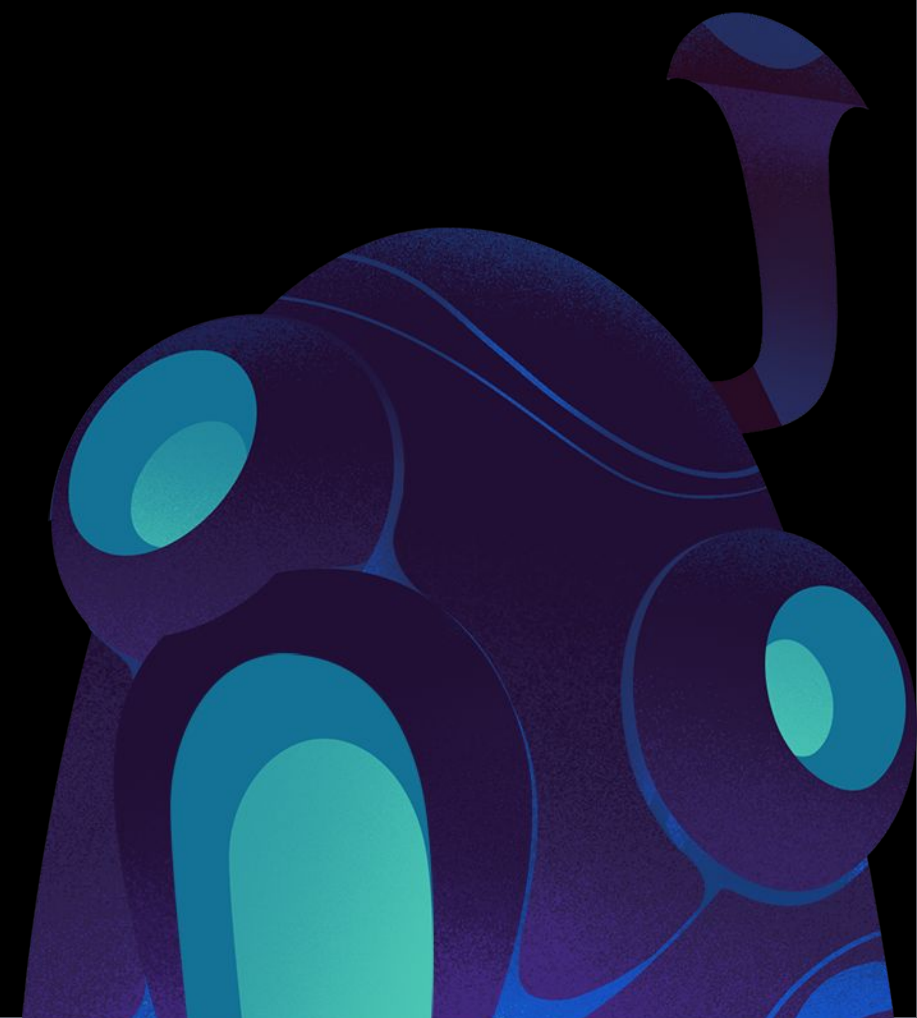


Network only



```
// Network only
self.addEventListener('fetch', event =>
  event.respondWith(fetch(event.request))
);
```

When to use this strategy:
Any data that you would not want persisted in your application offline (i.e analytics pings)



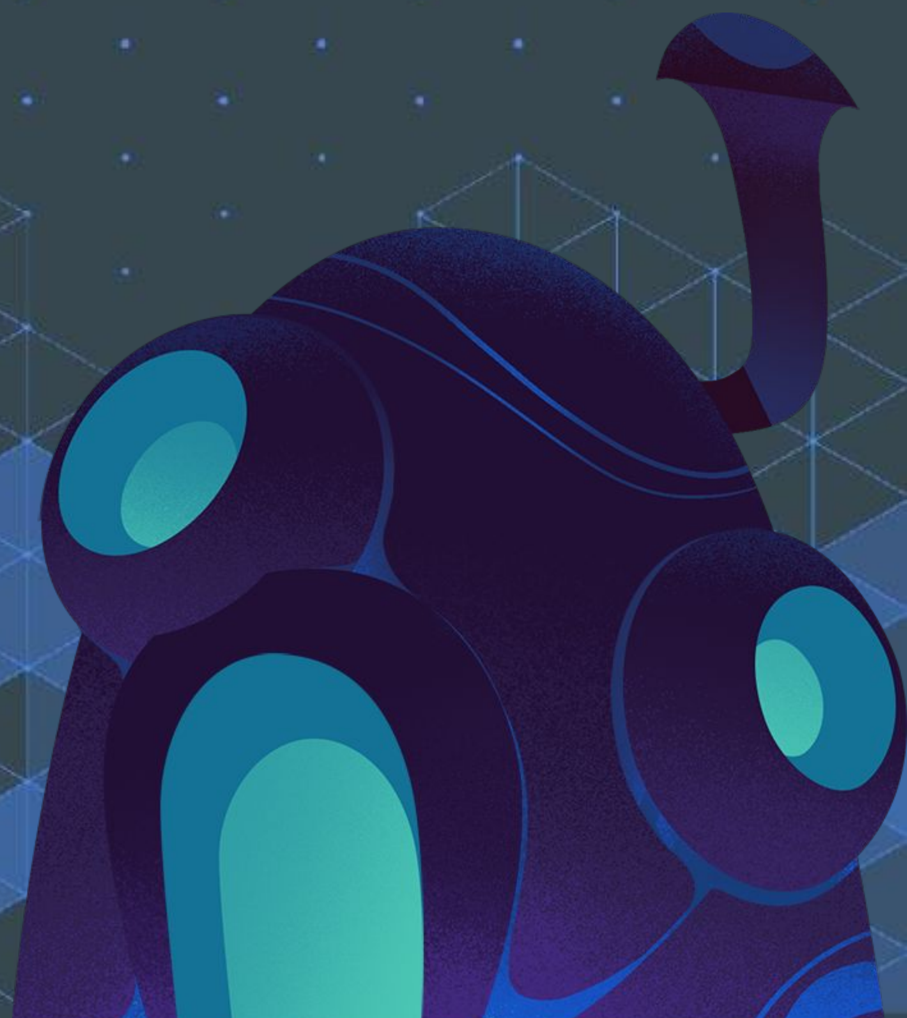
Caching strategies

Network-only

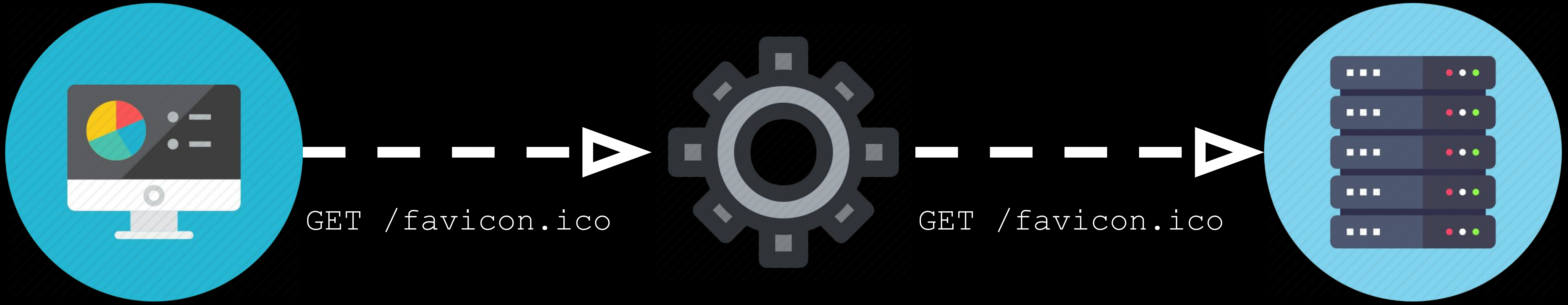
Cache-only

Cache falling back to network

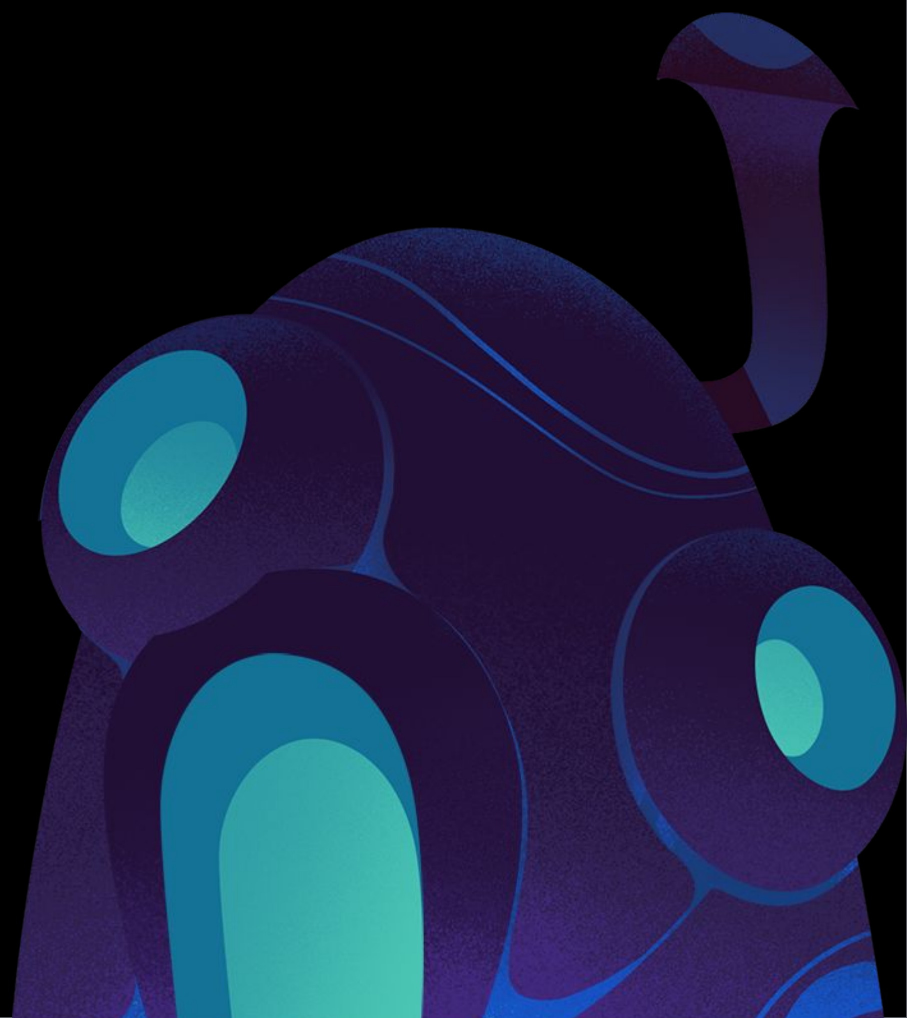
Network falling back to cache



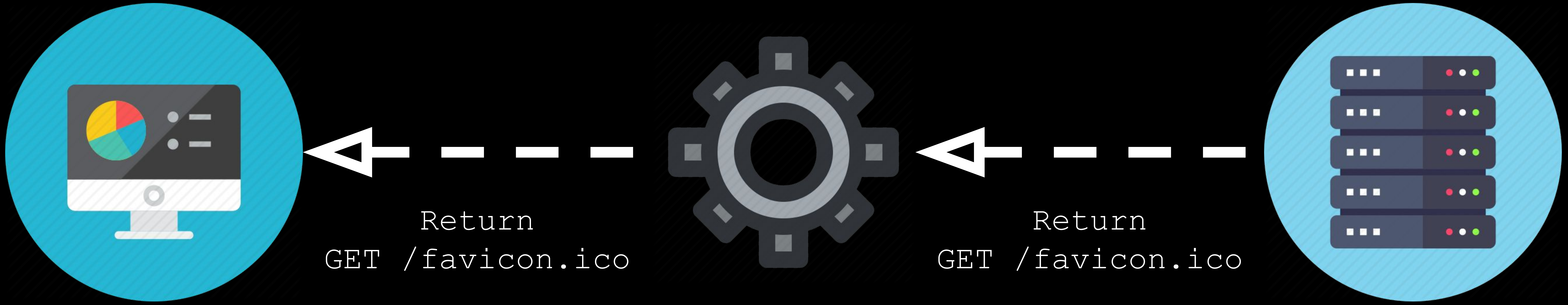
Cache <u>only</u>



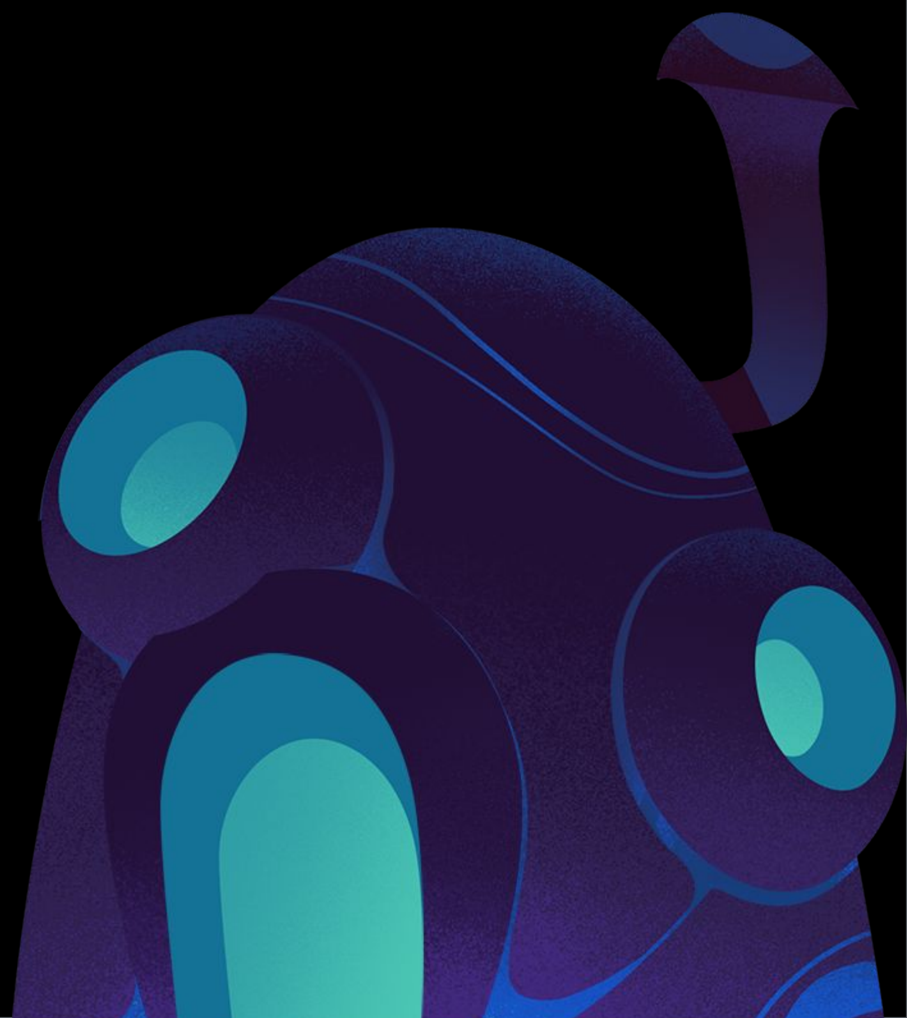
When to use this strategy:
This should only be used for assets (i.e images, favicons, logo) that will not be changing anytime soon



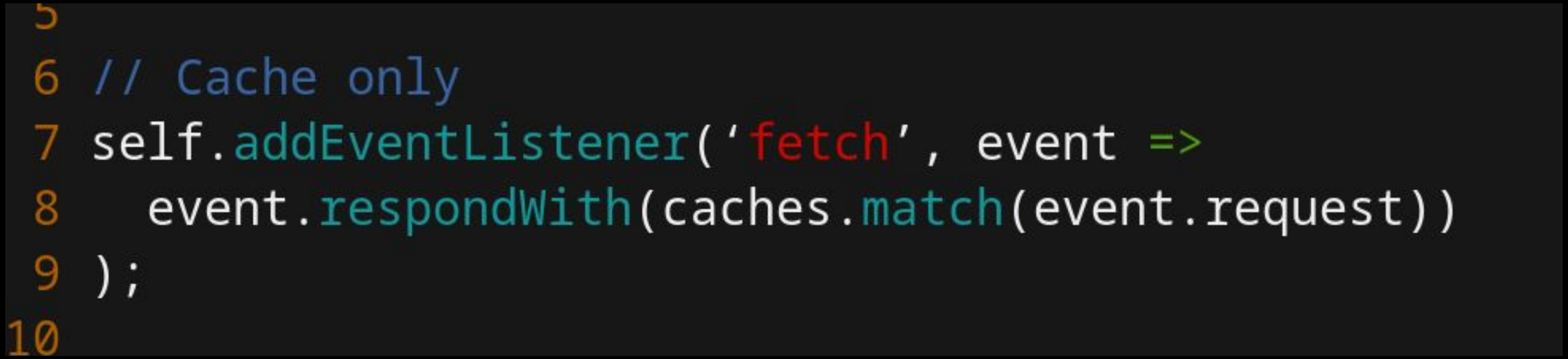
Cache only



When to use this strategy:
This should only be used for assets (i.e images, favicons, logo) that will not be changing anytime soon

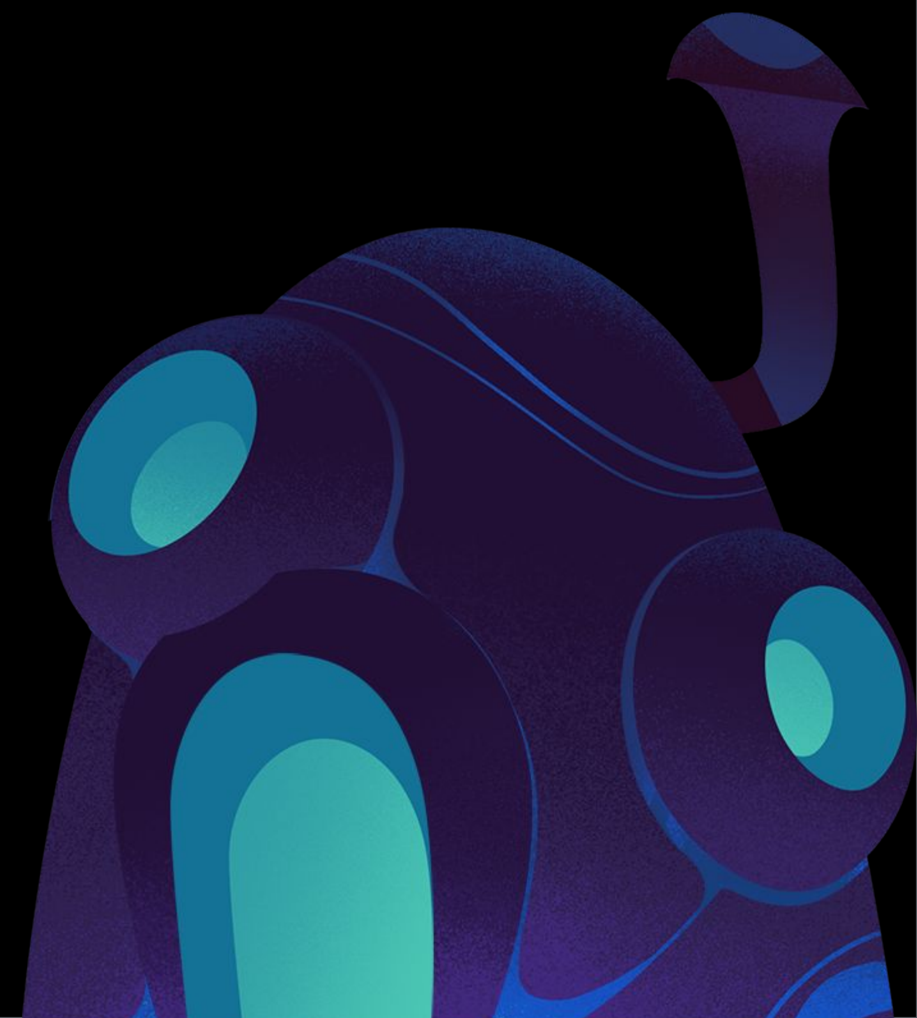


Cache <u>only</u>



```
// Cache only
self.addEventListener('fetch', event =>
  event.respondWith(caches.match(event.request))
);
```

When to use this strategy:
This should only be used for assets (i.e images, favicons, logo) that will not be changing anytime soon



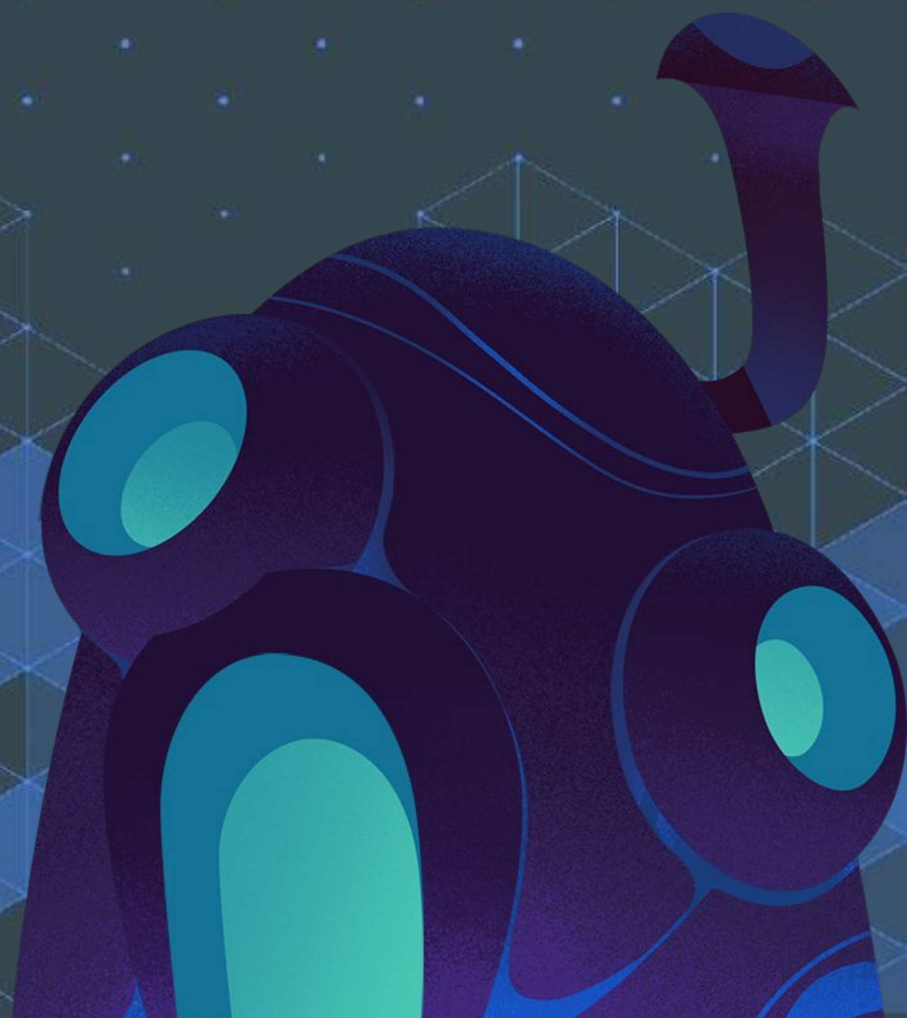
Caching strategies

Network-only

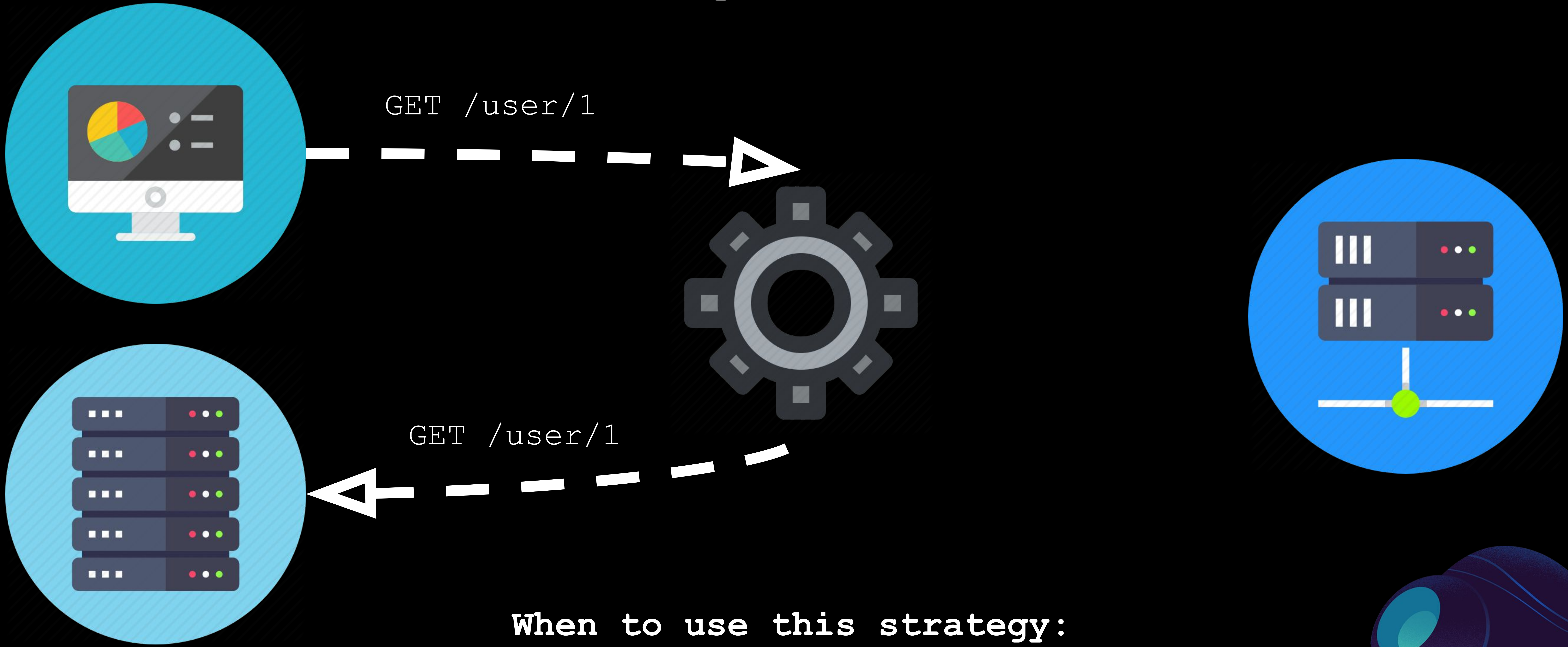
Cache-only

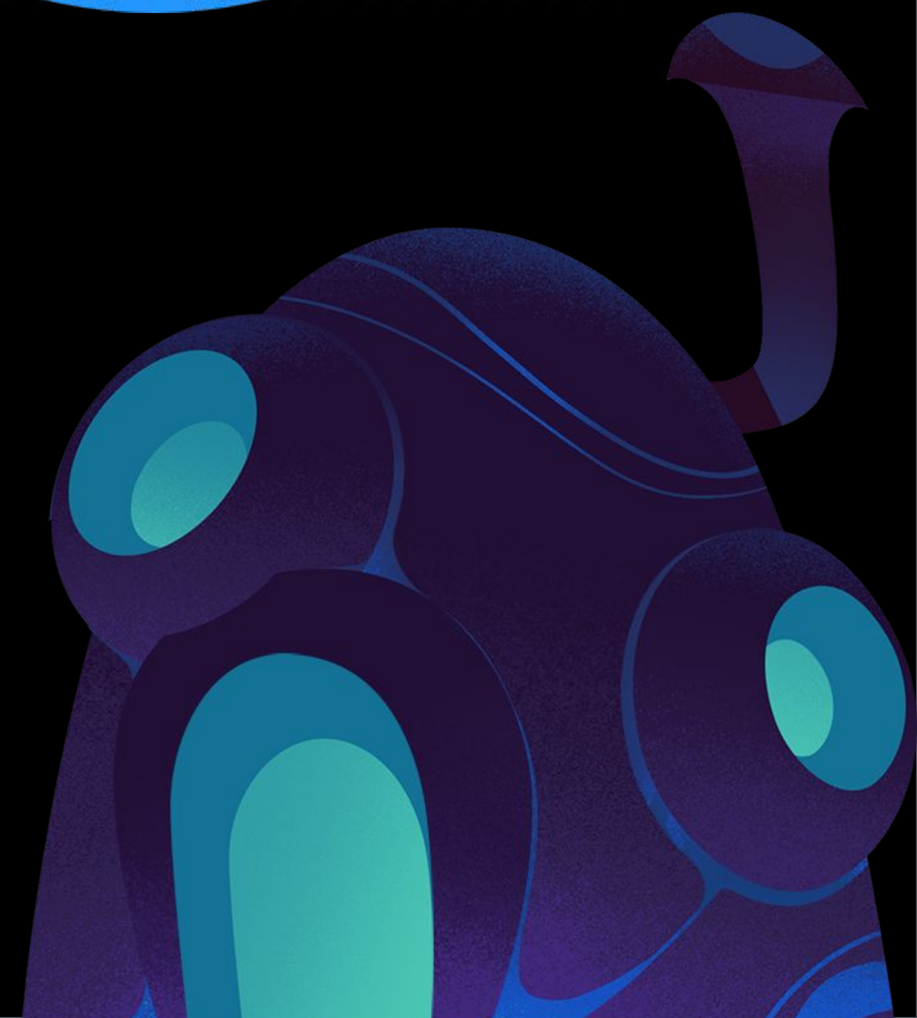
Cache falling back to network

Network falling back to cache



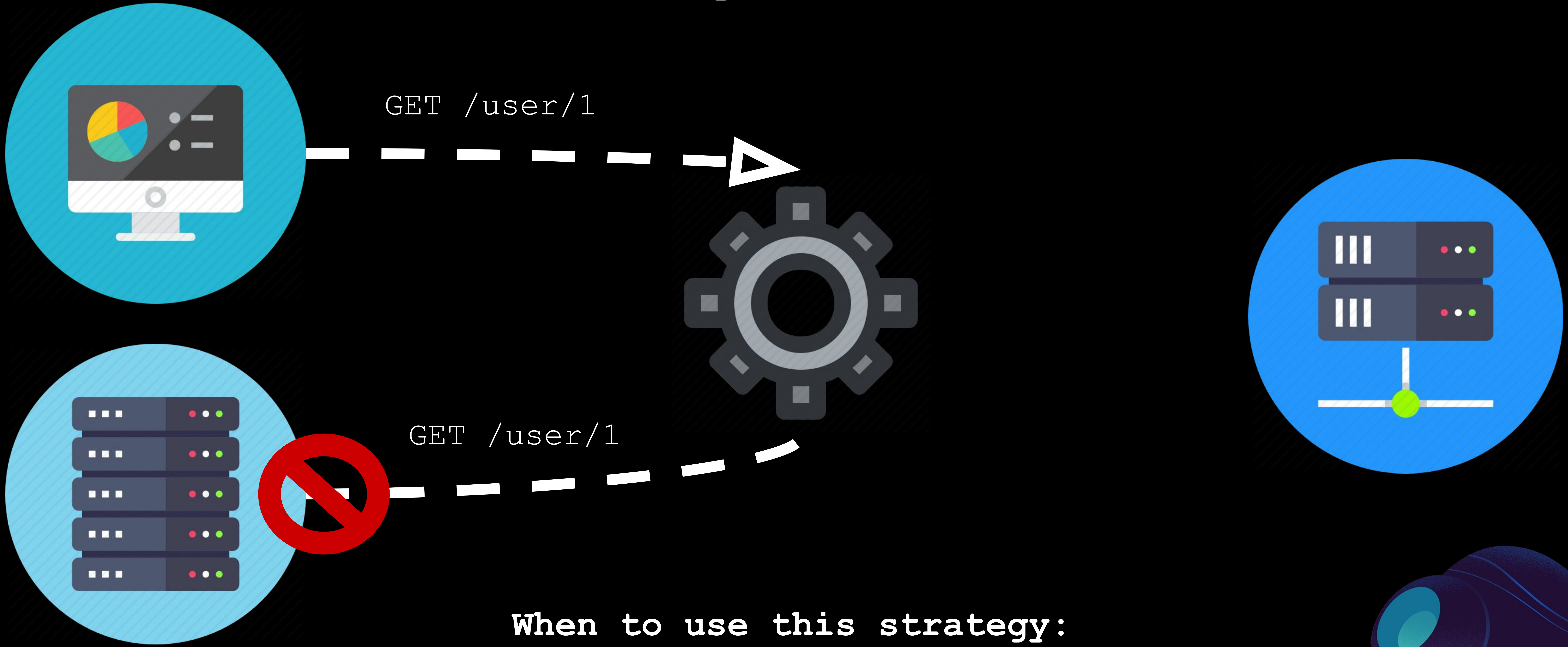
Cache falling back to network

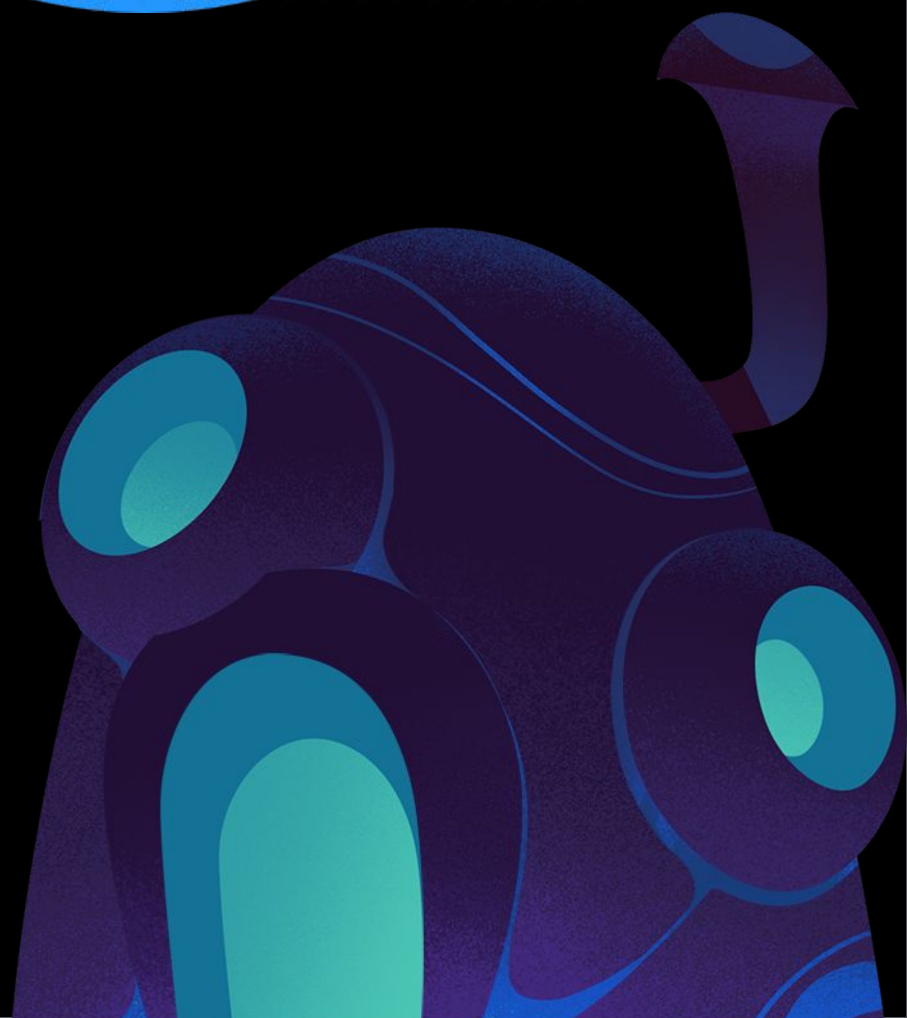




When to use this strategy:
For situations where you are building an application with offline mode in mind

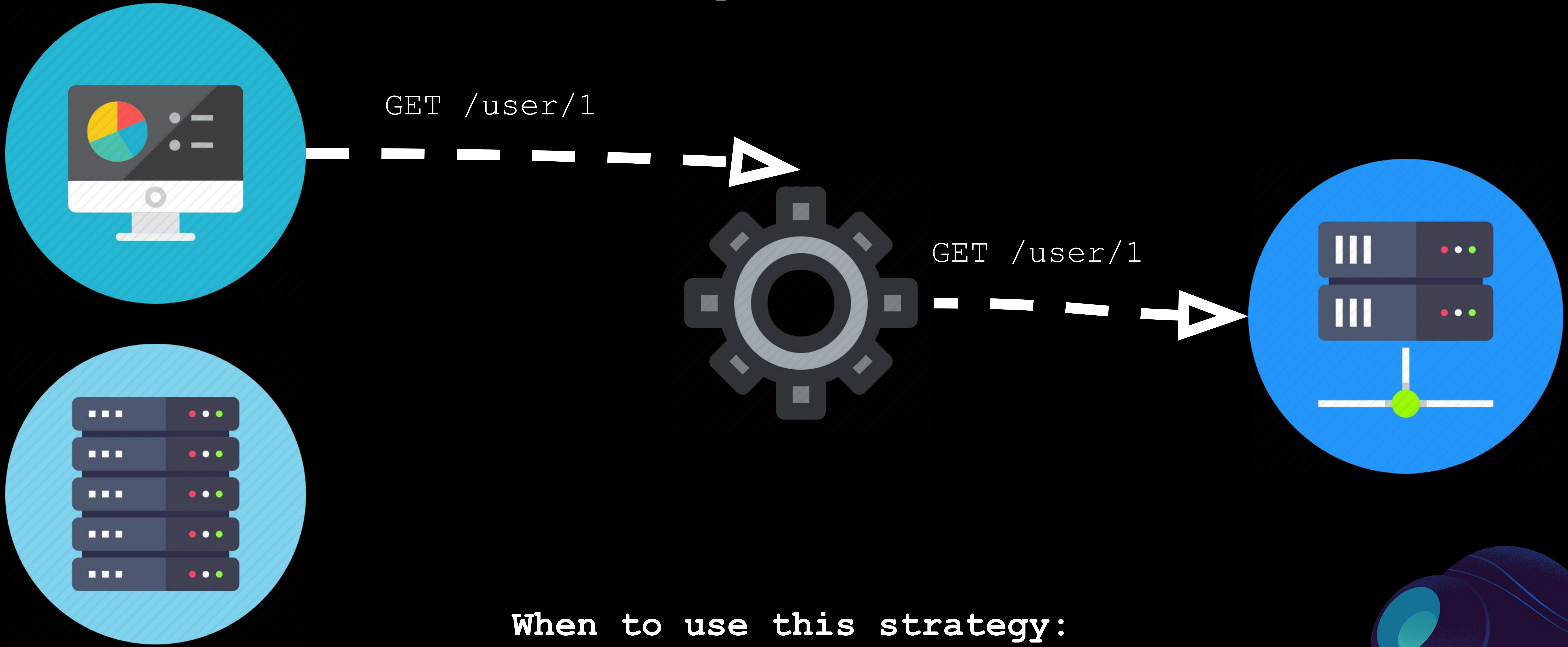
Cache falling back to network





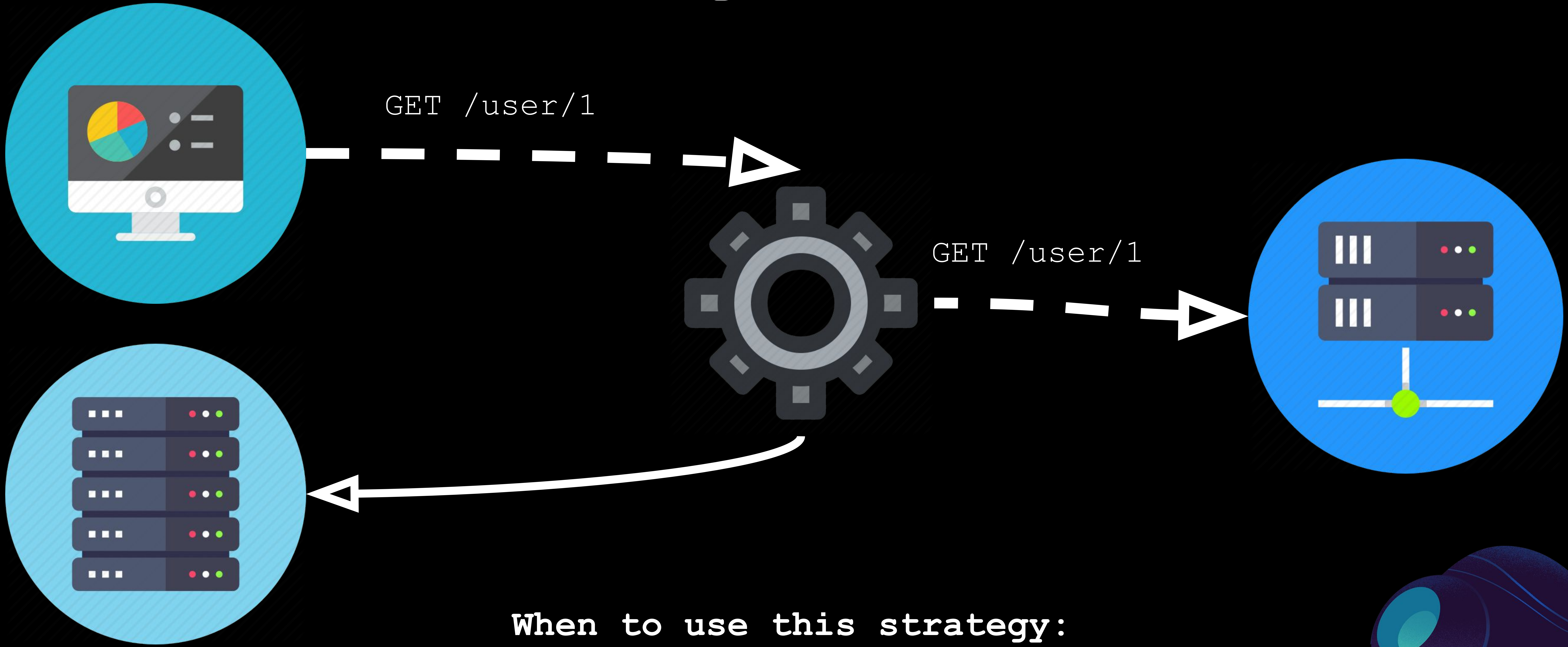
When to use this strategy:
For situations where you are building an application with offline mode in mind

Cache falling back to network

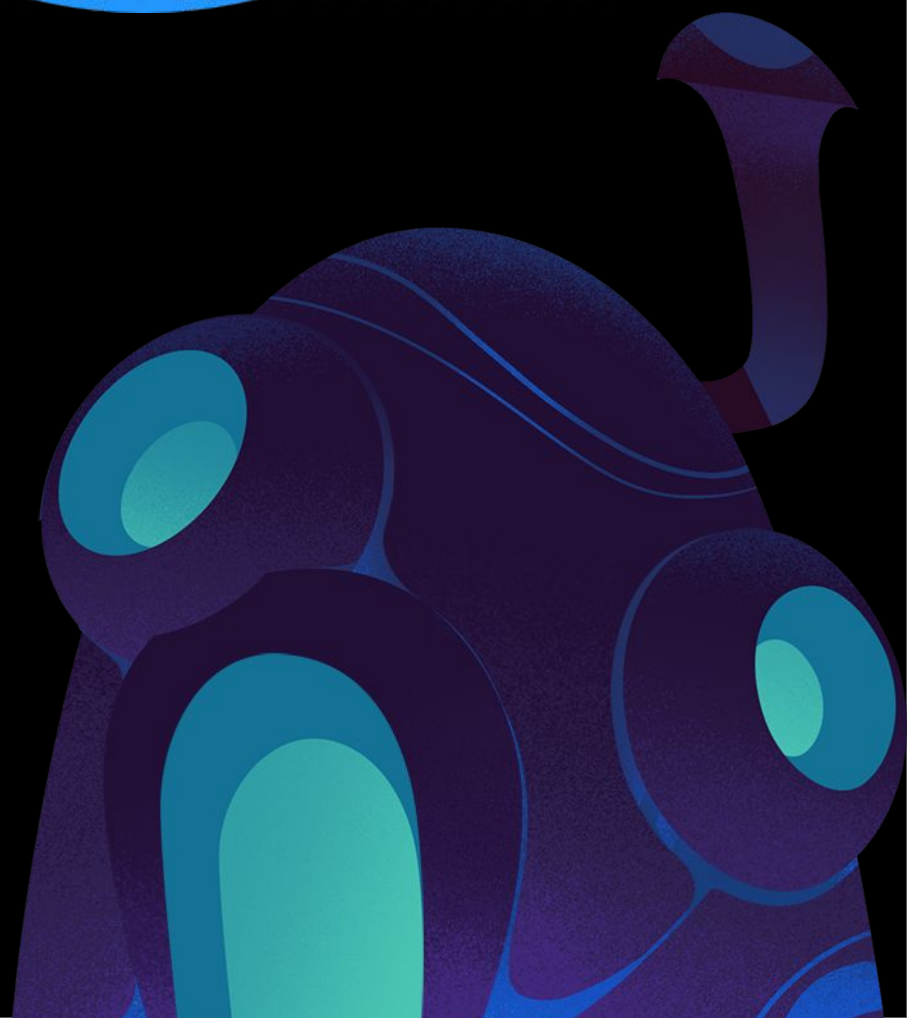


When to use this strategy:
For situations where you are building an application with offline mode in mind

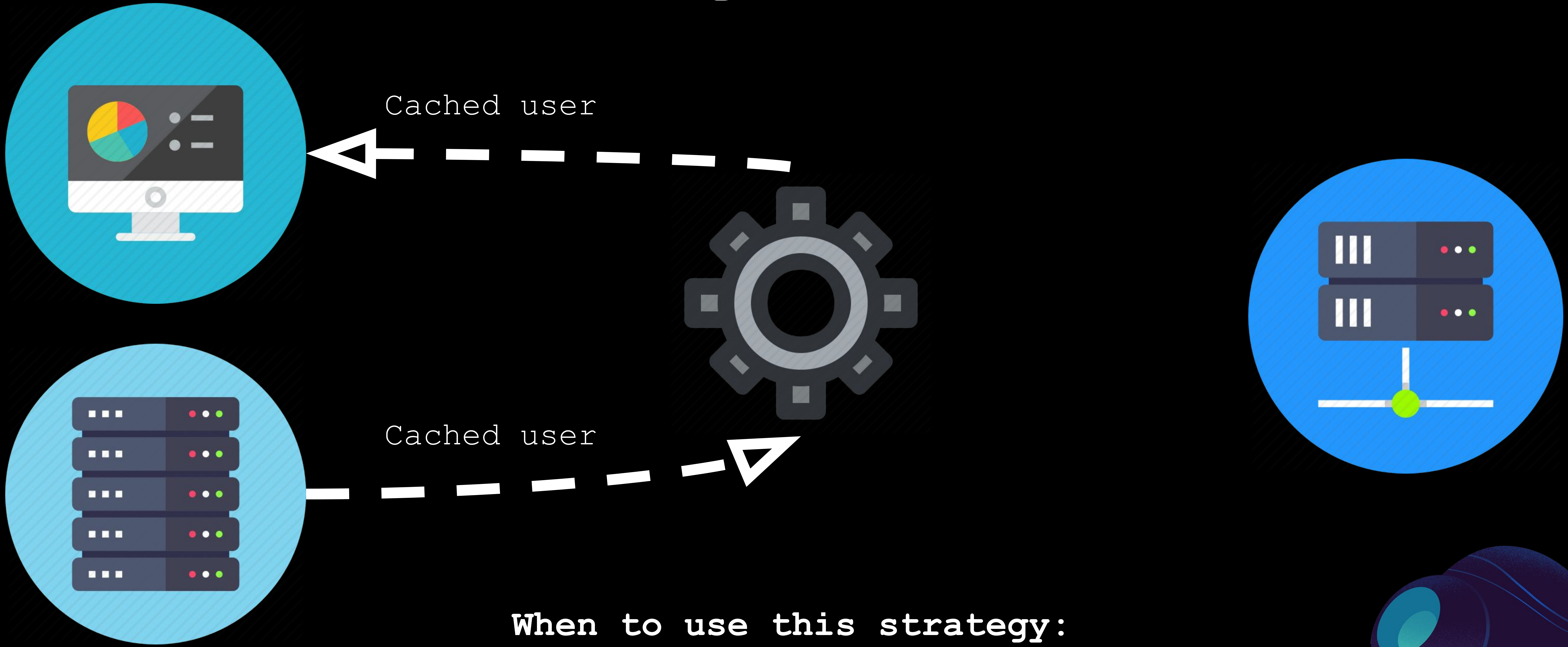
Cache falling back to network



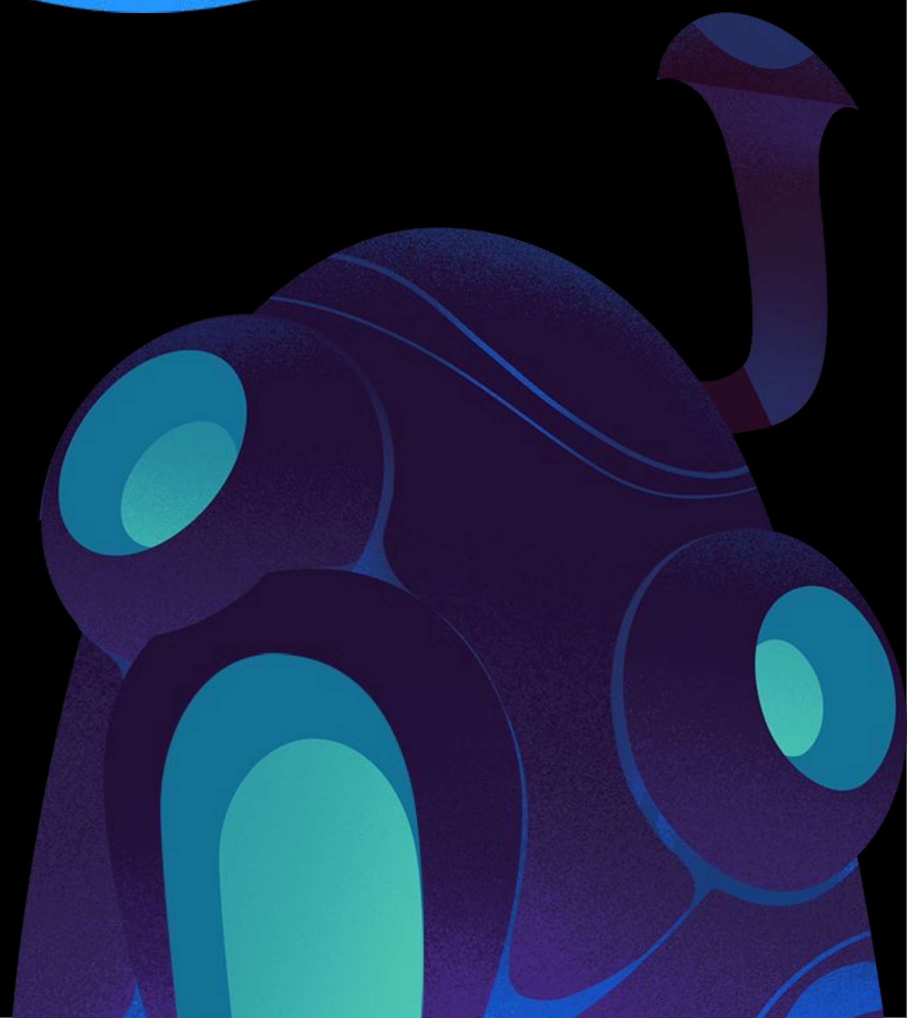
When to use this strategy:
For situations where you are building an application with offline mode in mind



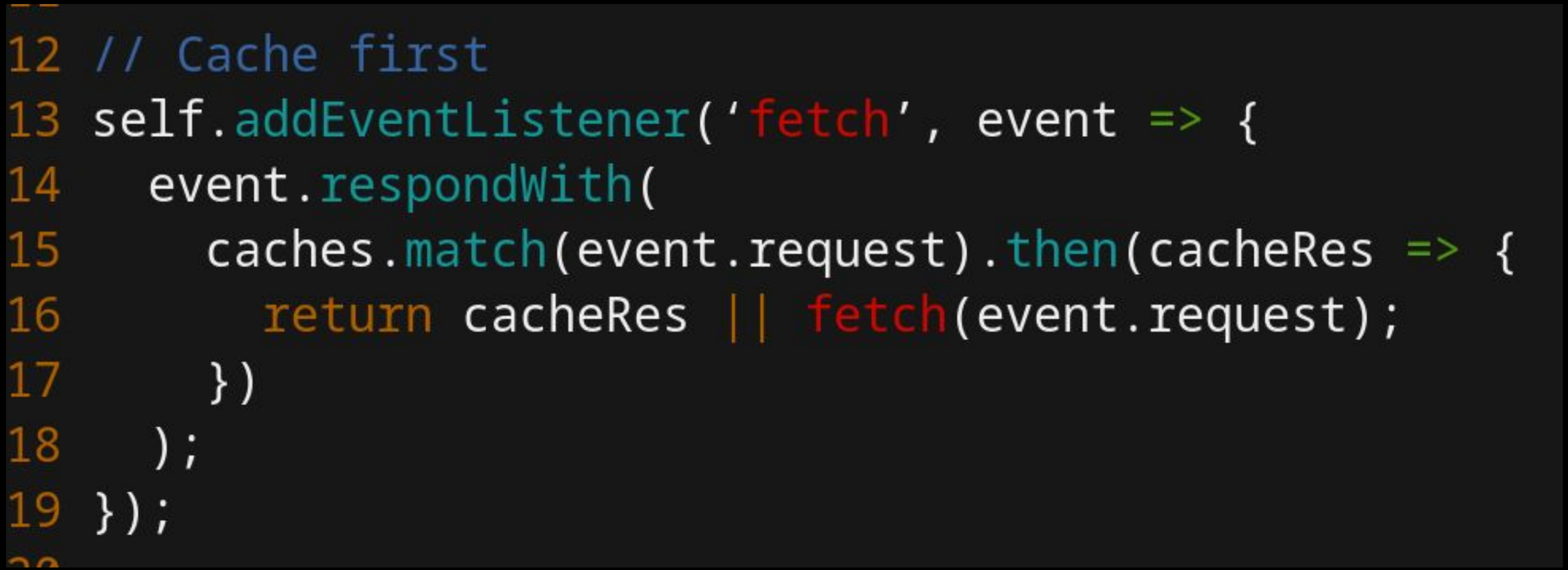
Cache falling back to network



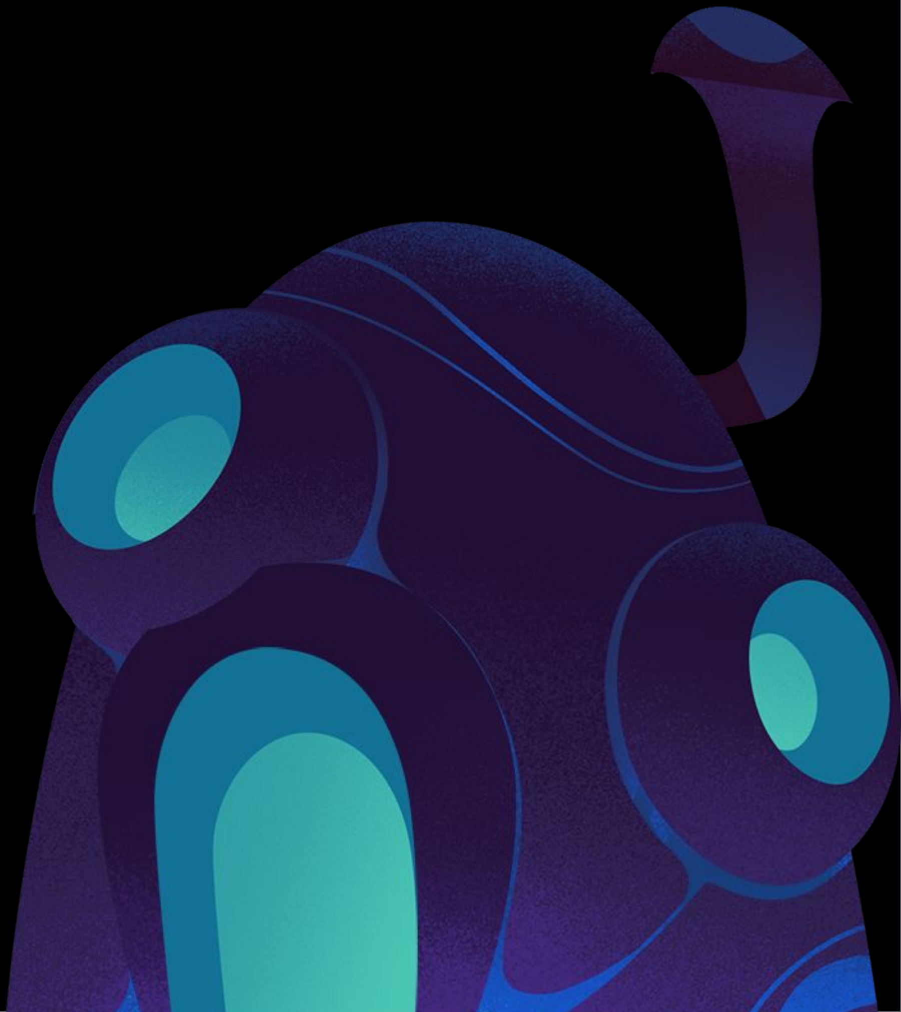
When to use this strategy:
For situations where you are building an application with offline mode in mind



Cache falling back to network



```
// Cache first
self.addEventListener(‘fetch’, event => {
  event.respondWith(
    caches.match(event.request).then(cacheRes => {
      return cacheRes || fetch(event.request);
    })
  );
});
```



When to use this strategy:
For situations where you are building an application with offline mode in mind

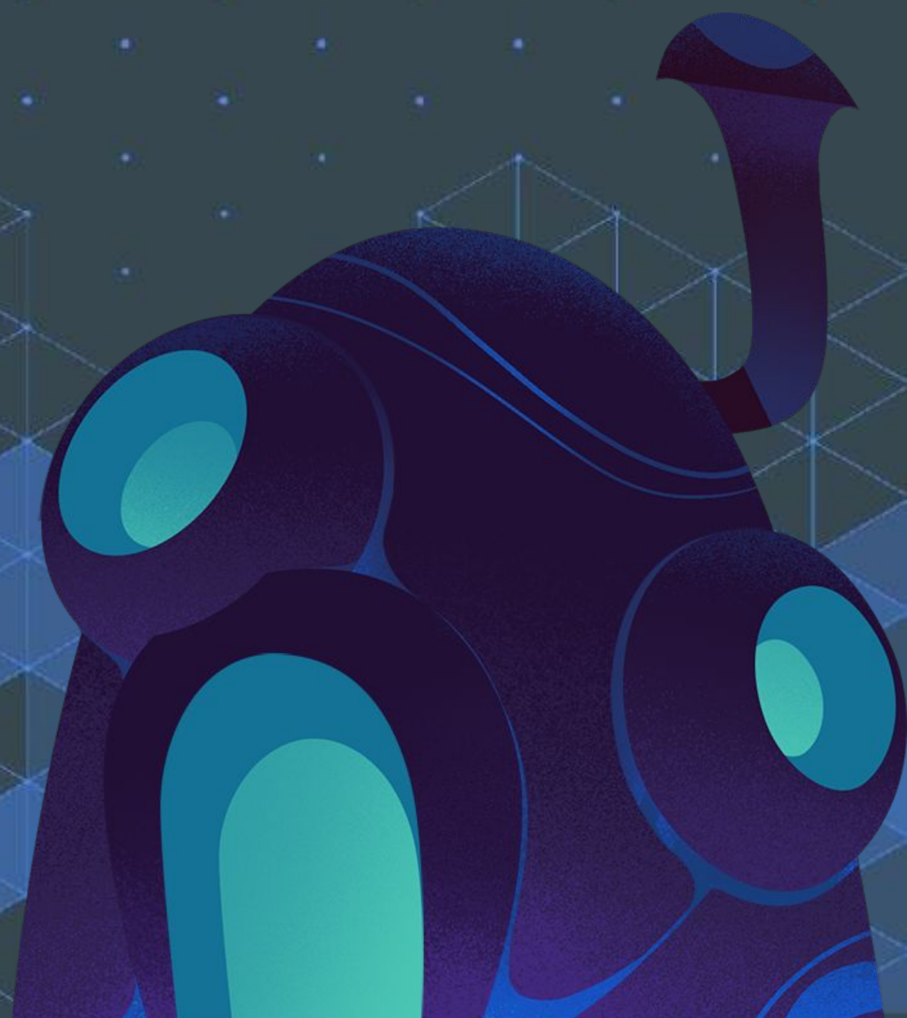
Caching strategies

Network-only

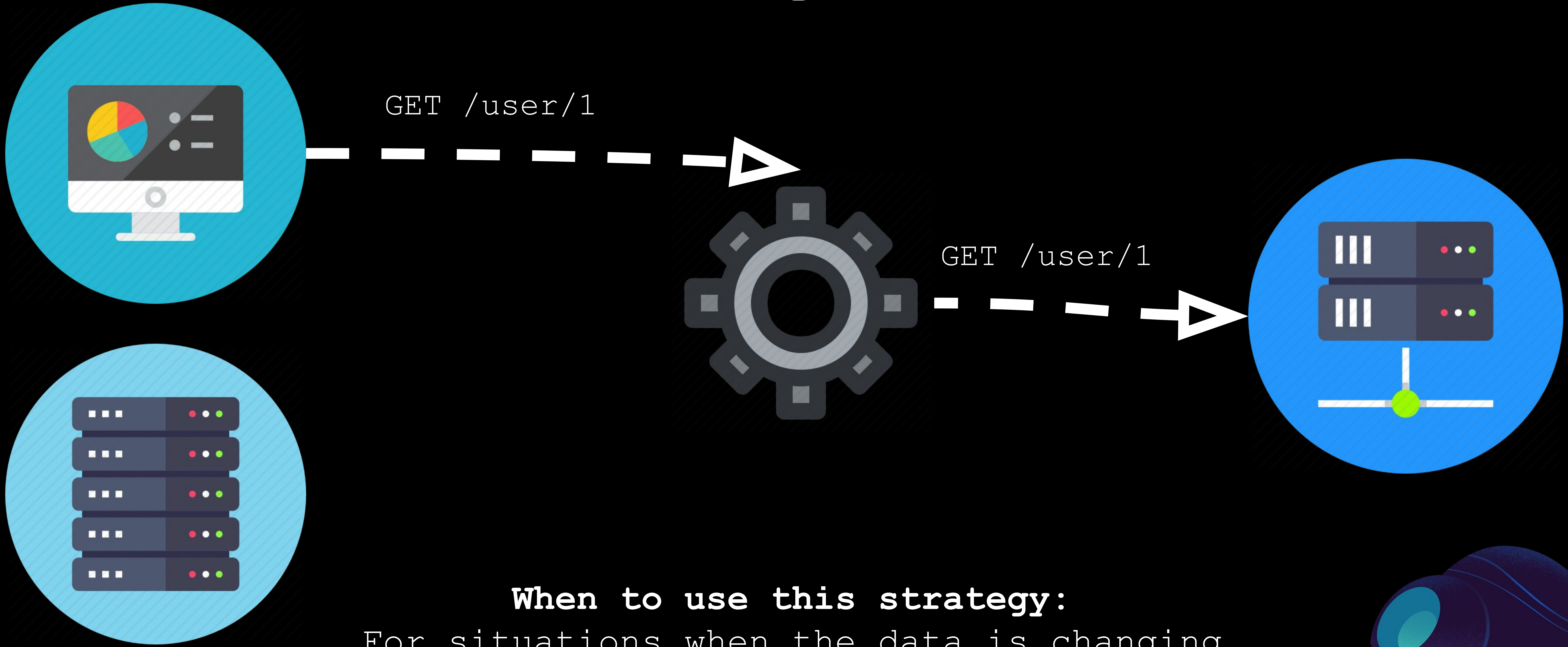
Cache-only

Cache falling back to network

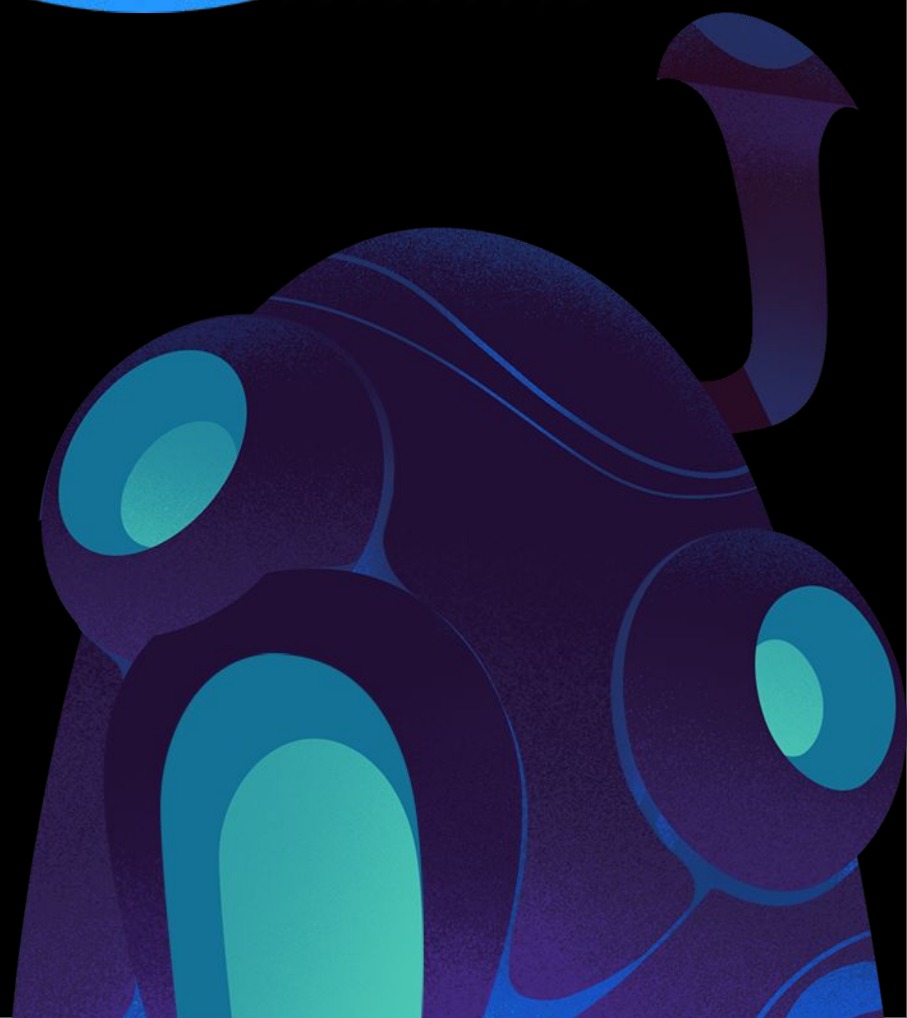
Network falling back to cache



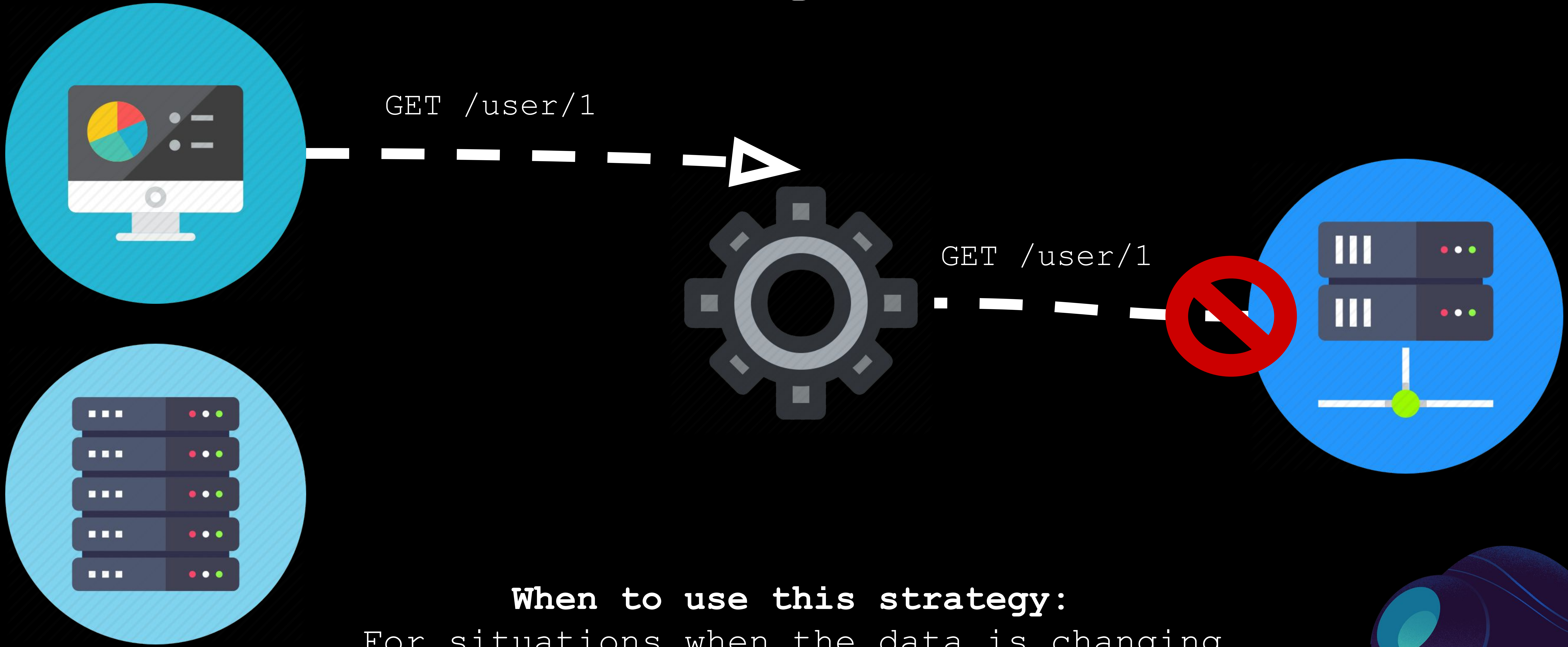
Network falling back to cache



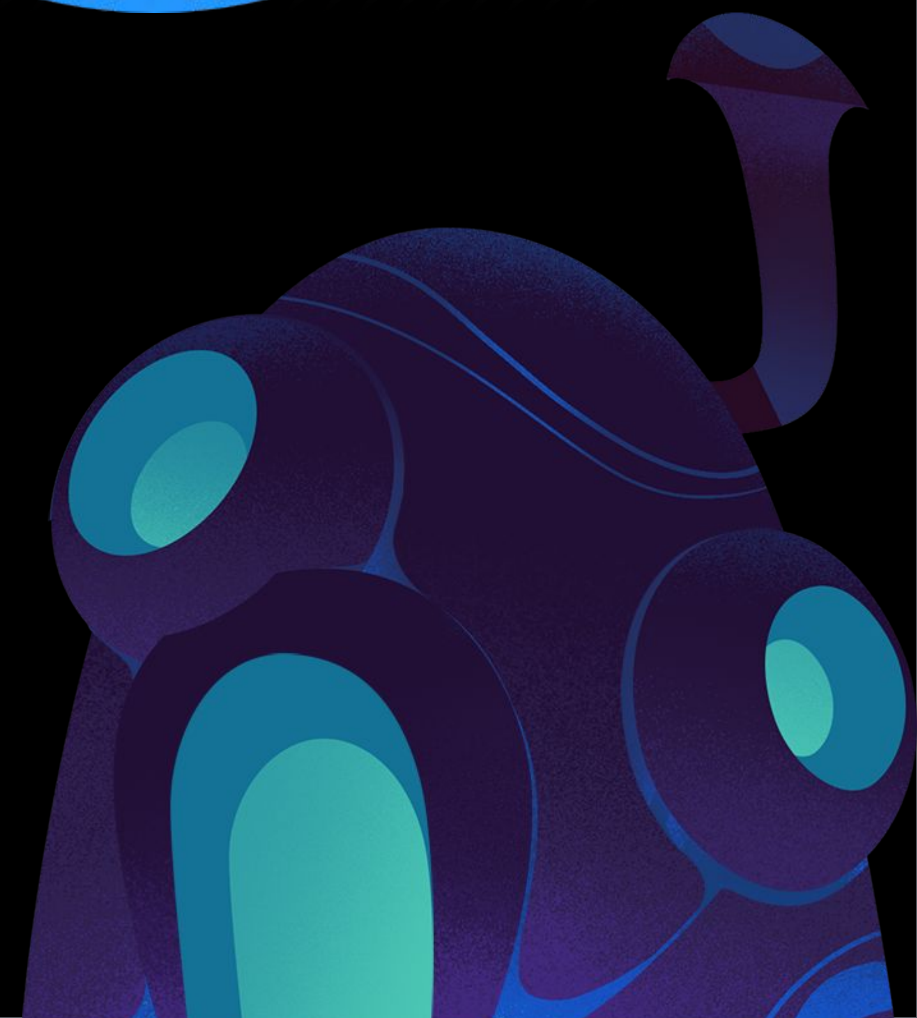
When to use this strategy:
For situations when the data is changing very quickly, like tracking stocks or game leaderboards



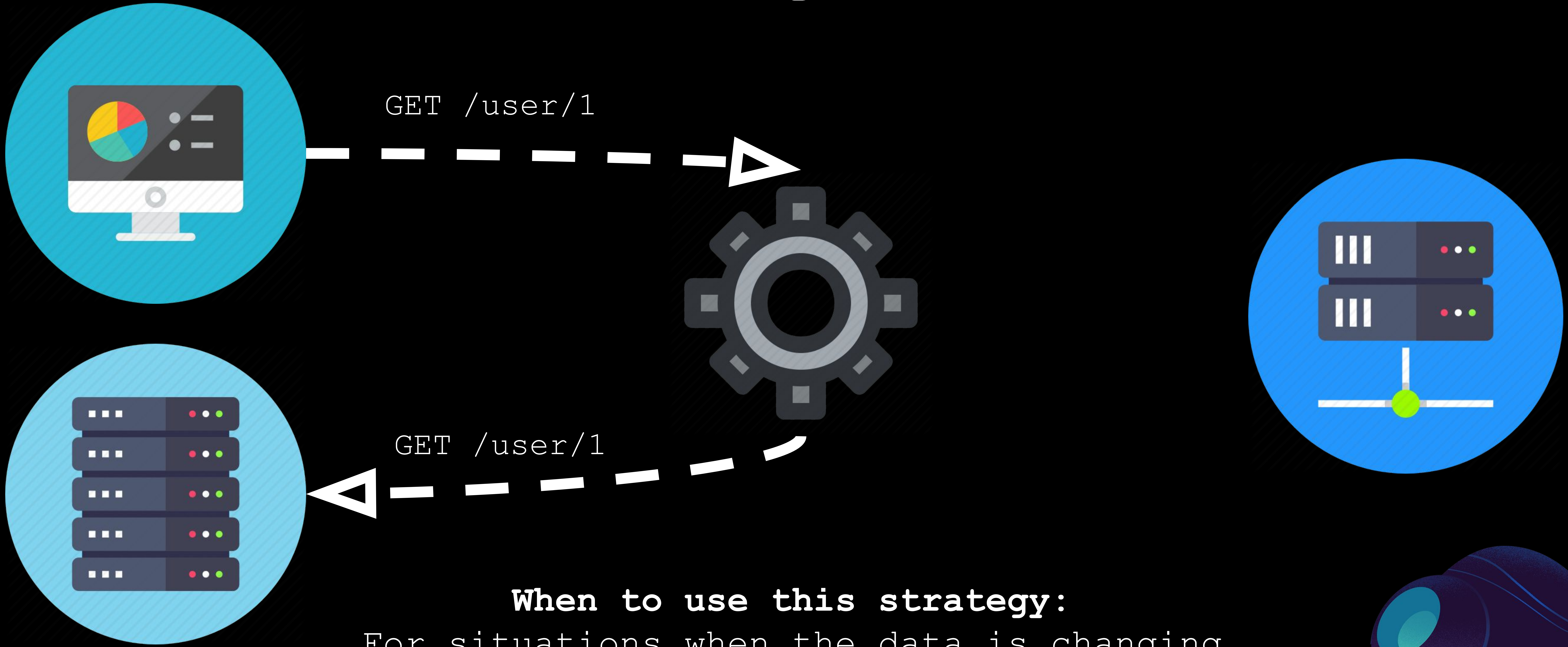
Network falling back to cache



When to use this strategy:
For situations when the data is changing very quickly, like tracking stocks or game leaderboards

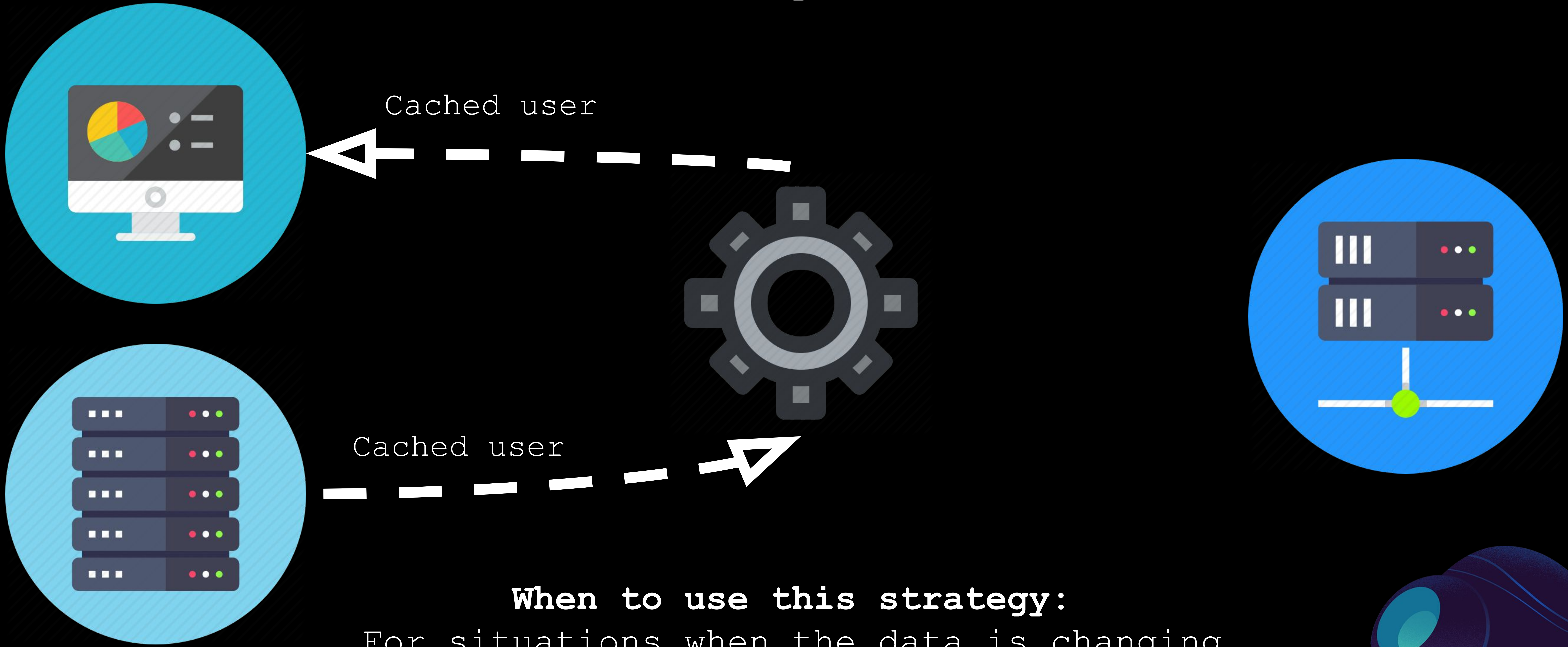


Network falling back to cache



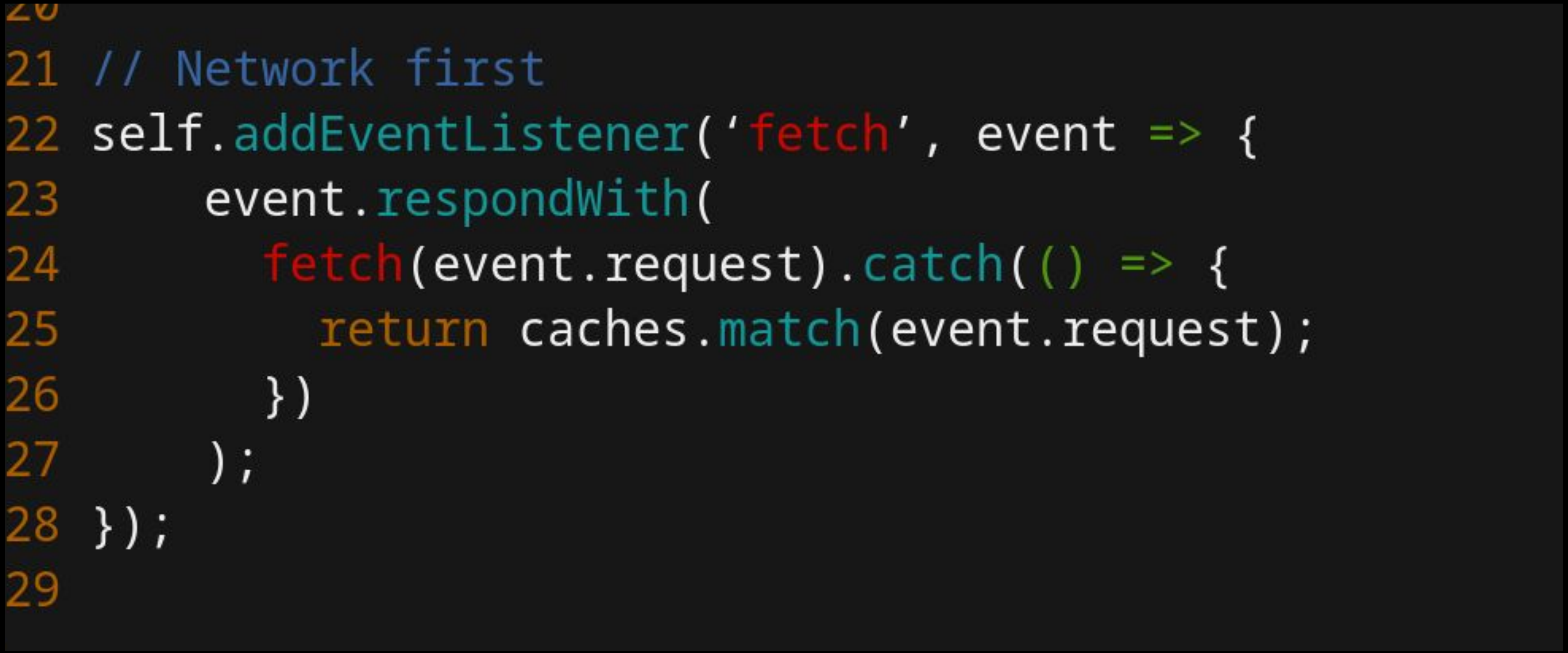
When to use this strategy:
For situations when the data is changing very quickly, like tracking stocks or game leaderboards

Network falling back to cache



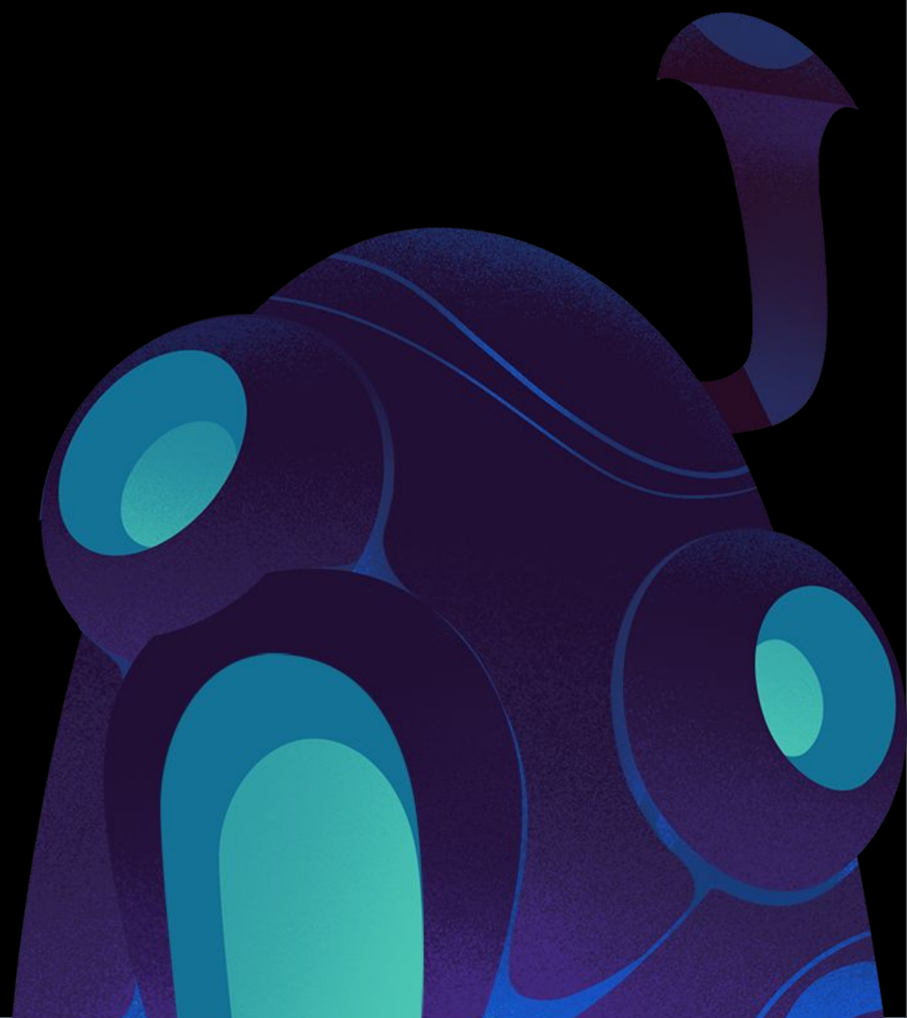
When to use this strategy:
For situations when the data is changing very quickly, like tracking stocks or game leaderboards

Network falling back to cache



```
// Network first
self.addEventListener(‘fetch’, event => {
    event.respondWith(
      fetch(event.request).catch(() => {
        return caches.match(event.request);
      })
    );
});
```

When to use this strategy:
For situations when the data is changing very quickly, like tracking stocks or game leaderboards



Resources

- https://serviceworkies.com
- https://web.dev/reliable
- https://codelabs.developers.google.com/codelabs/workbox-lab/#0
- https://developers.google.com/web/tools/workbox/guides/get-started

That's all folks!

Asset Credit: https://serviceworkies.com/