radley yeldar.

# Cache Me Outside!

Anthony Dang - Technical Director

@anthonydotnet

# Caching Bug or Feature?



```php
public function getCachedChartBy($get_by, $get_by_value, $date = null, $limit = 100){
  $cachekey = md5("{$get_by}{$get_by_value}{$limit}{$date}");
  return Cache::remember($cachekey, rand(30,300), function() use ($get_by, $get_by_value,
    return $this->getChartBy($get_by, $get_by_value, $date, $limit);
  });
}
```

radley yeldar.

A bit about me

CERTIFIED MASTER

PAL I

PSM I

AMBASSADOR GITKRAKEN

UMBRACO MVP

# Why do we cache?

- Limited bandwidth / availability
- Cost
- Blazing fast UX
- SEO
- Poor Performance?

# Caching Options

DNS/Proxy

Network

Page Output

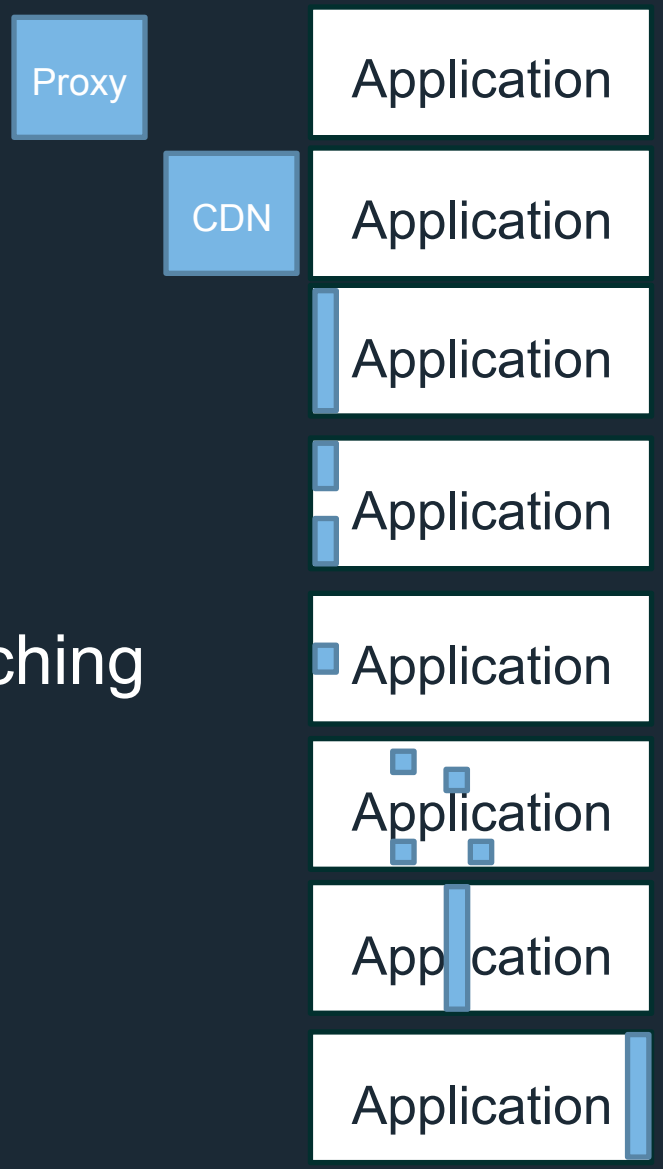
Donut Caching

Donut Hole Caching

Methods

Service layer

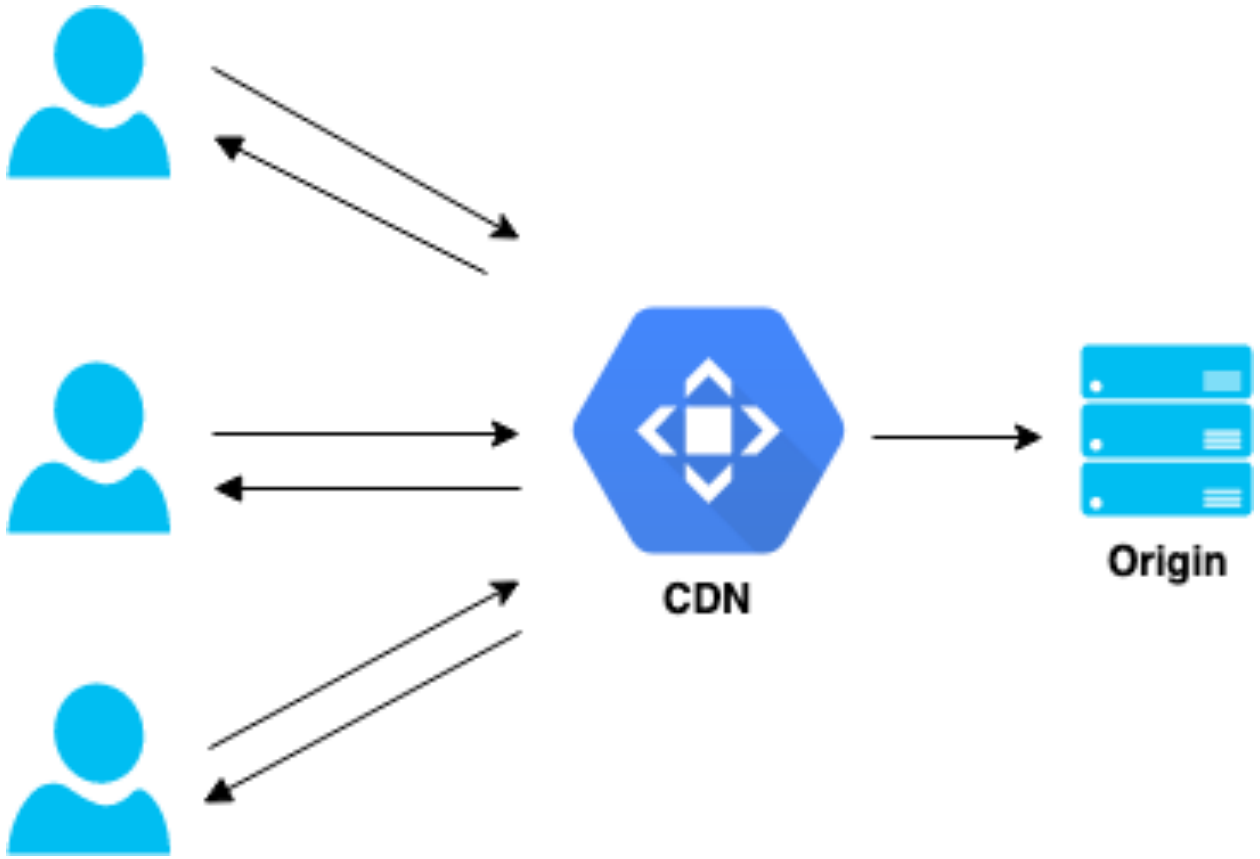
Data layer




## Pre-Indexing

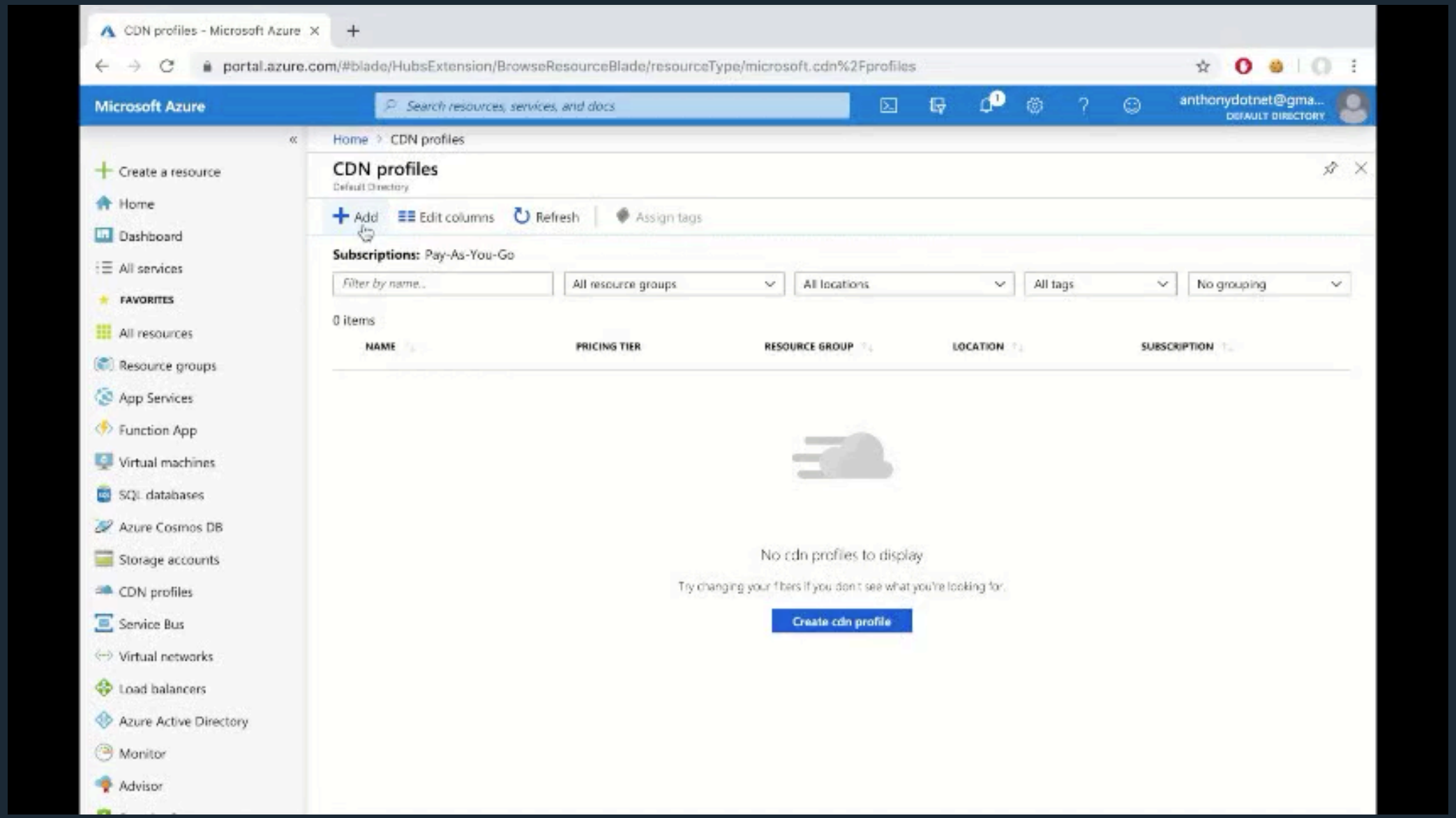

## Multi-layered

# Content Delivery Networks



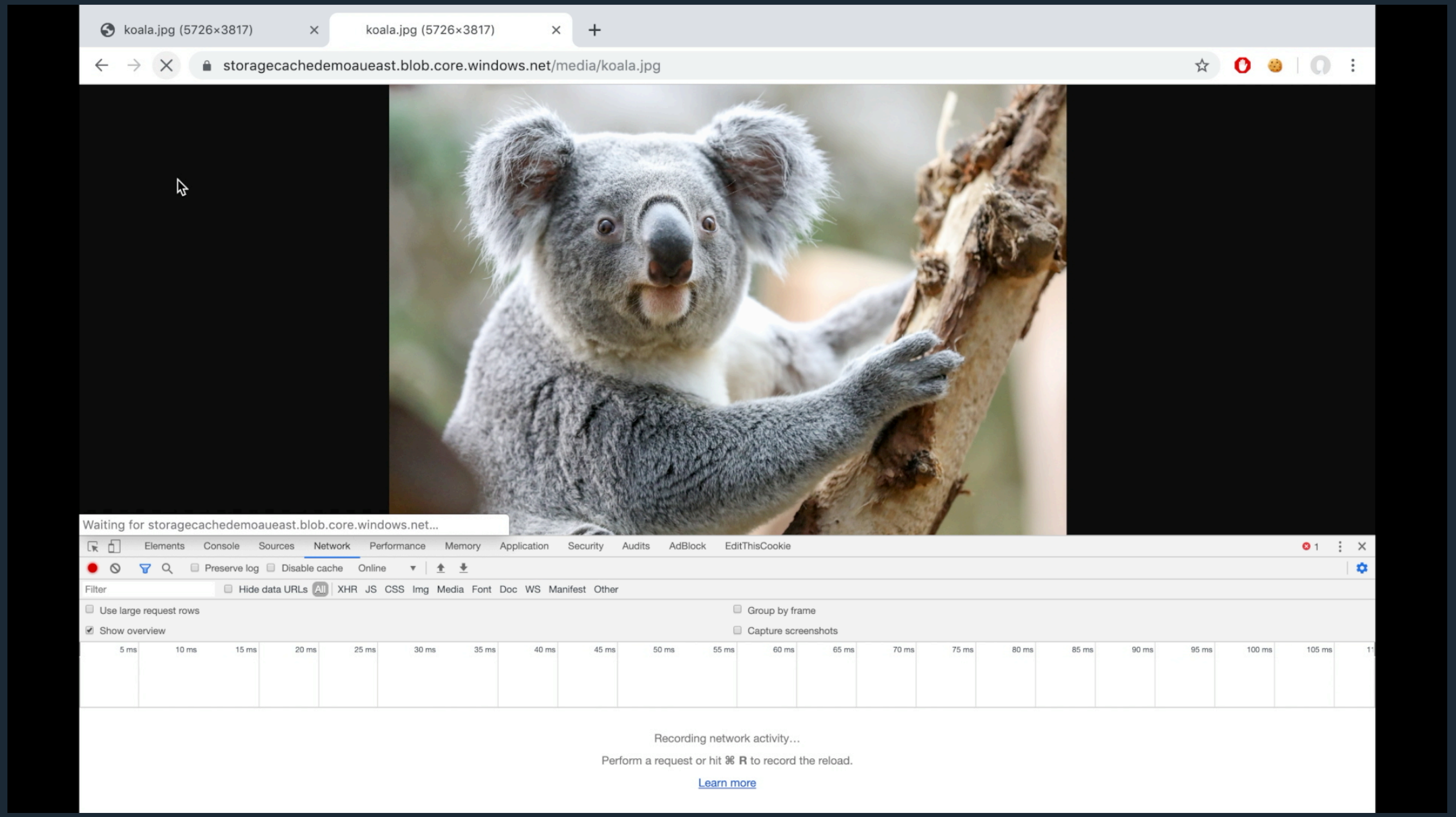
# Content Delivery Networks - Azure Portal





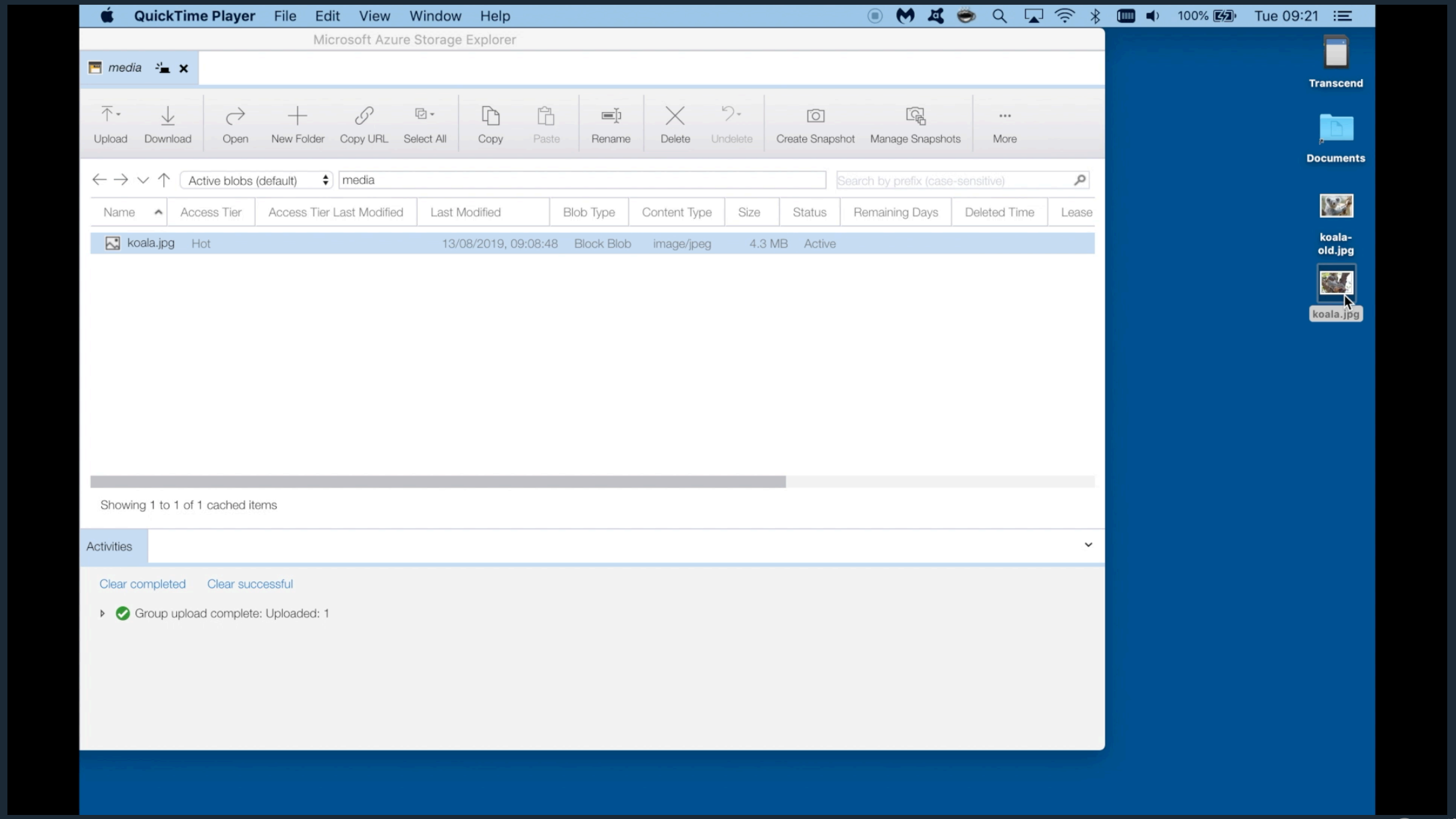
# Blob Storage vs Content Delivery Networks





# Content Delivery Networks - Cache Update & Purging





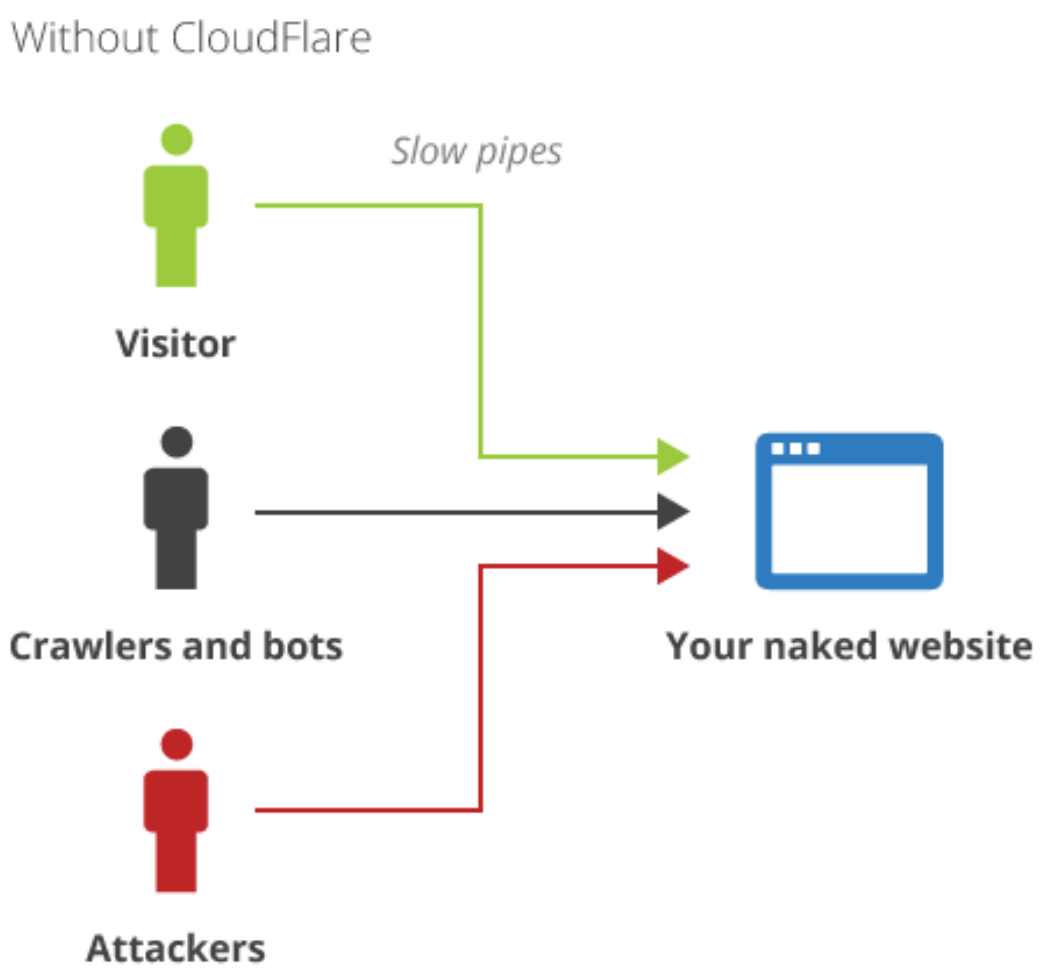
# Content Delivery Networks - Considerations

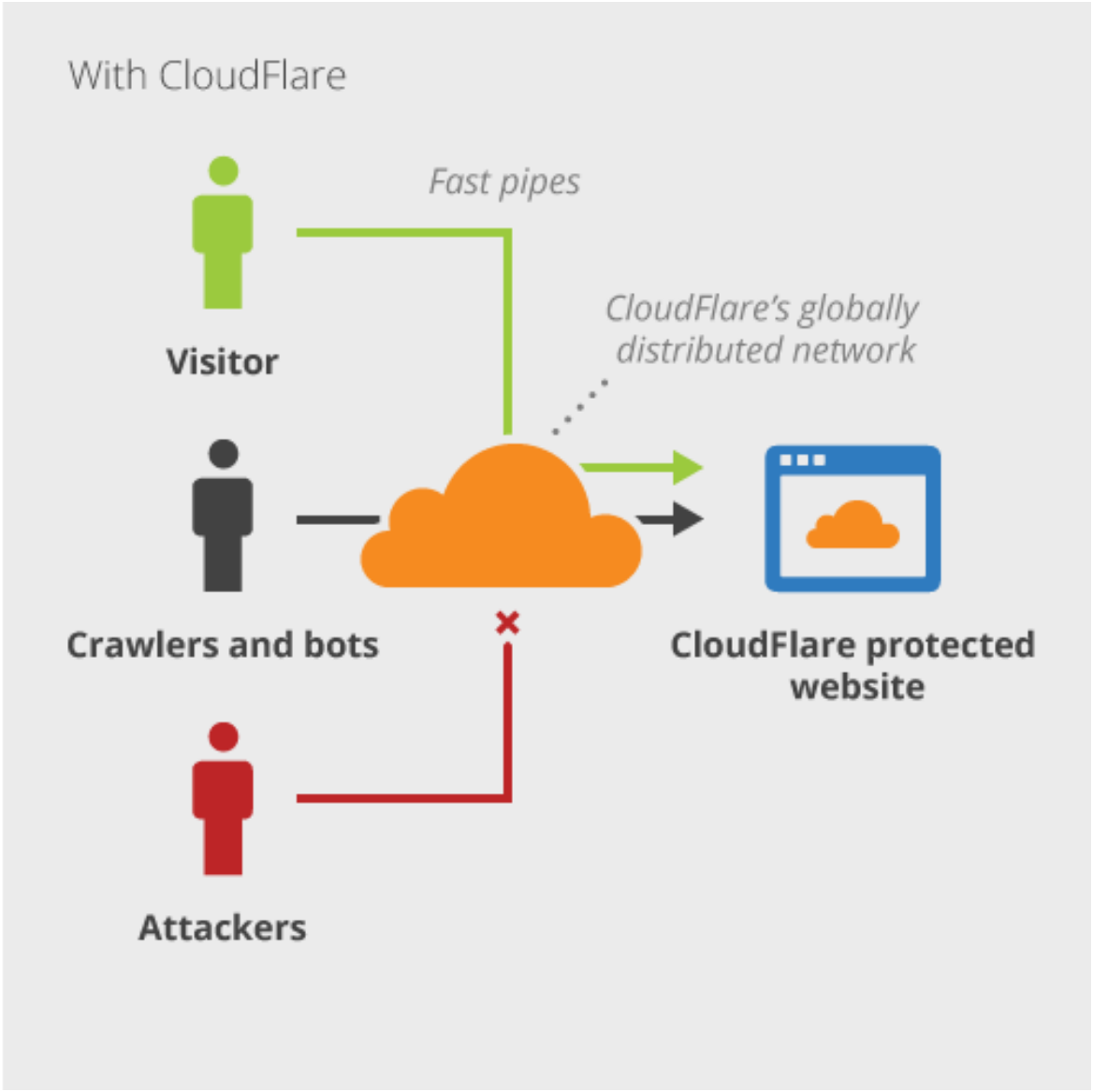
- When will cache timeout?
- Is stale content acceptable?
- How to clear the cache?
- How to get files up there?

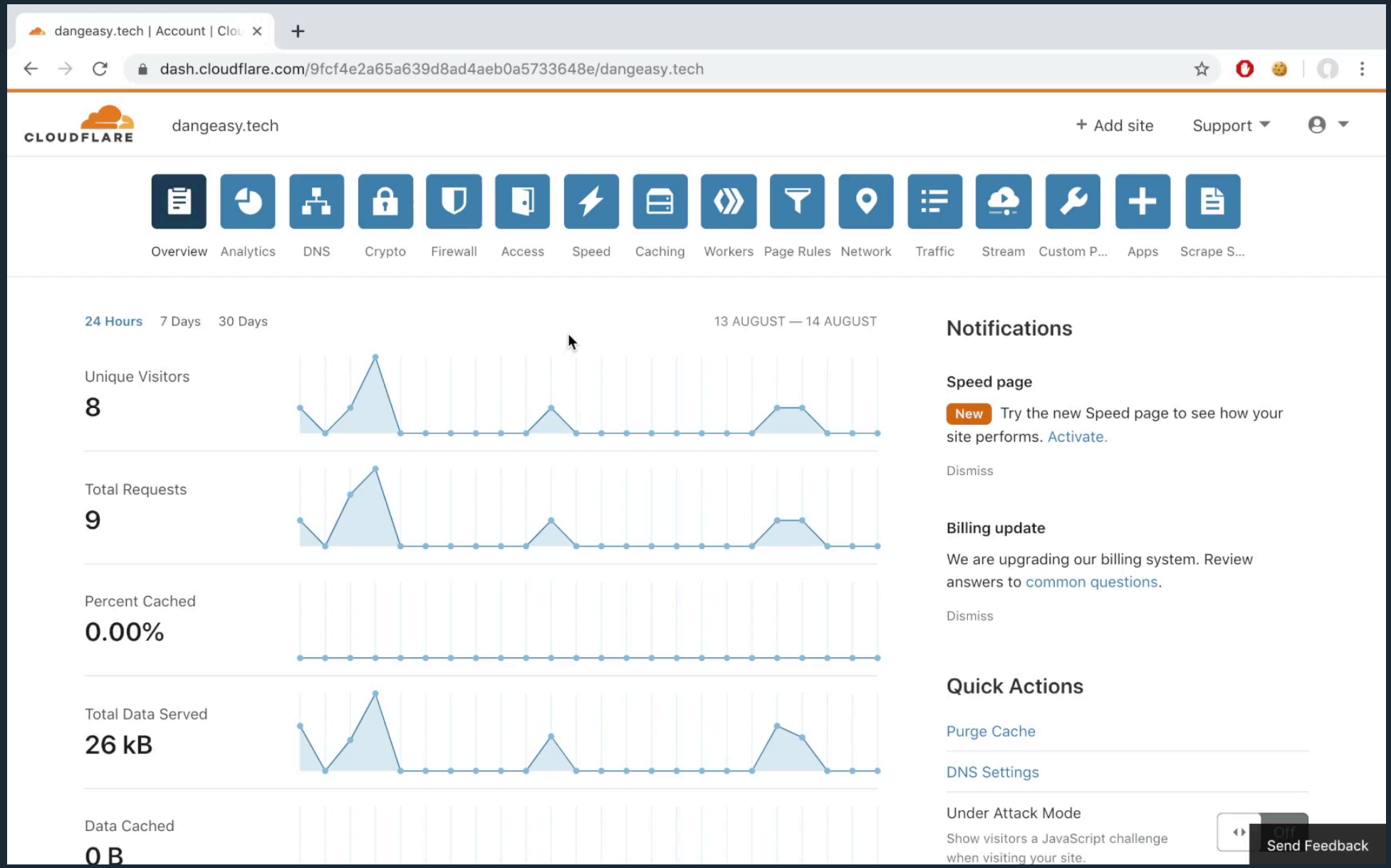
## Browser cache?

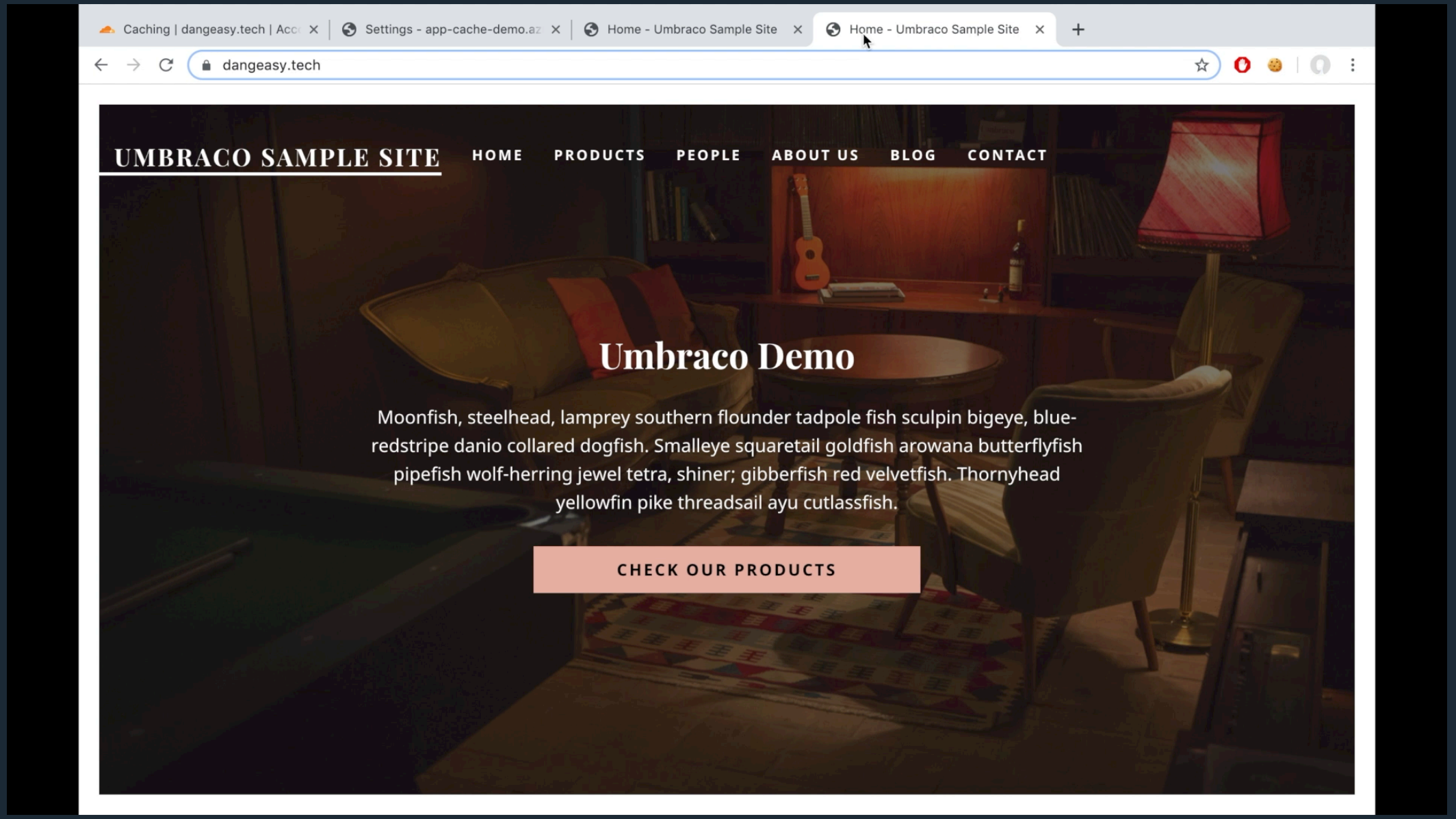
- Is this a problem?

# Cloudflare


Without CloudFlare
Slow pipes
Visitor
Crawlers and bots
Your naked website
Attackers


With CloudFlare
Fast pipes
CloudFlare's globally distributed network
Visitor
Crawlers and bots
CloudFlare protected website
Attackers


dangeasy.tech | Account | Clou
dash.cloudflare.com/9fcf4e2a65a639d8ad4aeb0a5733648e/dangeasy.tech
CLOUDFLARE
dangeasy.tech
+ Add site
Support
Overview
Analytics
DNS
Crypto
Firewall
Access
Speed
Caching
Workers
Page Rules
Network
Traffic
Stream
Custom P...
Apps
Scrape S...
24 Hours
7 Days
30 Days
13 AUGUST — 14 AUGUST
Unique Visitors
8
Total Requests
9
Percent Cached
0.00%
Total Data Served
26 kB
Data Cached
0 B
Notifications
Speed page
New
Try the new Speed page to see how your site performs. Activate.
Dismiss
Billing update
We are upgrading our billing system. Review answers to common questions.
Dismiss
Quick Actions
Purge Cache
DNS Settings
Under Attack Mode
Show visitors a JavaScript challenge when visiting your site.
Off
Send Feedback


Caching | dangeasy.tech | Acc
Settings - app-cache-demo.az
Home - Umbraco Sample Site
Home - Umbraco Sample Site
dangeasy.tech
UMBRACO SAMPLE SITE
HOME
PRODUCTS
PEOPLE
ABOUT US
BLOG
CONTACT
Umbraco Demo
Moonfish, steelhead, lamprey southern flounder tadpole fish sculpin bigeye, blue-redstripe danio collared dogfish. Smalleye squaretail goldfish arowana butterflyfish pipefish wolf-herring jewel tetra, shiner; gibberfish red velvetfish. Thornyhead yellowfin pike threadsail ayu cutlassfish.
CHECK OUR PRODUCTS

# Content Delivery Networks - Considerations

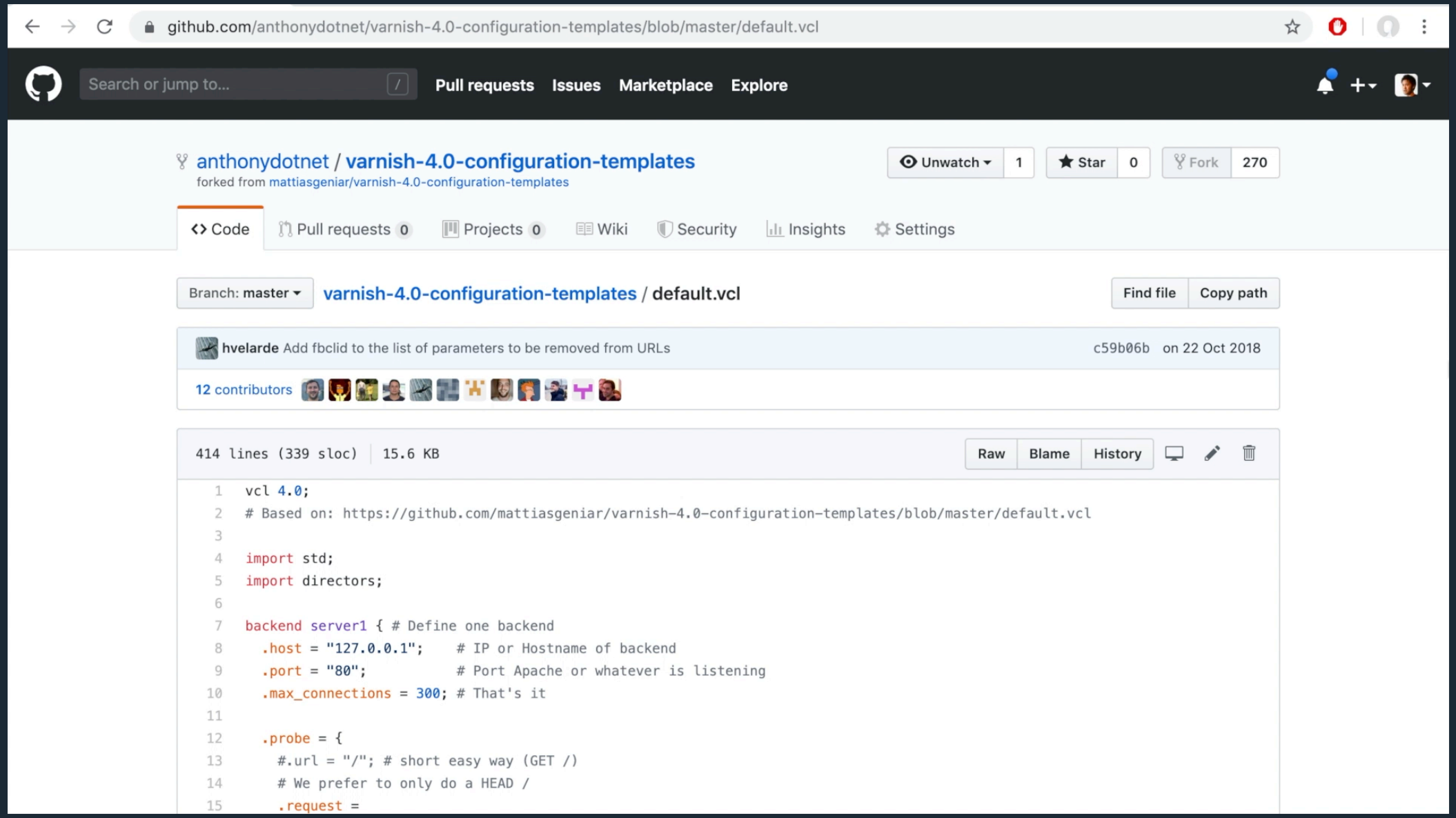
- Hides downtime/crash until timeout
- Errors being cached
- Limited page rules
  - 100, then 1USD per rule
- Security

# Varnish

- Self hosted – Linux
- Cached HTTP Proxy
- Granular unlimited rules
- Configuration

# Varnish Config





# Varnish - Considerations

- Crazy config file
- Maintain linux server
    - Fastly?

# Coding time

# Partial Caching



```
@Html.CachedPartial("MyPartialName", new MyModel(), 3600)
```

# Partial Cache - Considerations

- Hardcoded cache time
- How to clear cache?
- How to turn off cache?

```
@Html.CachedPartial("MyPartialName", new MyModel(), 3600)
```

# Output Caching



```
public class ProfileController : Controller
{
    [OutputCache(CacheProfile="Cache1Hour")]
    public string Index()
    {
        return DateTime.Now.ToString("T");
    }
}
```

```
<caching>
  <outputCache enableOutputCache="true" />
  <outputCacheSettings>
    <outputCacheProfiles>
      <add name="Cache1Hour" duration="3600" varyByParam="none" />
    </outputCacheProfiles>
  </outputCacheSettings>
</caching>
```

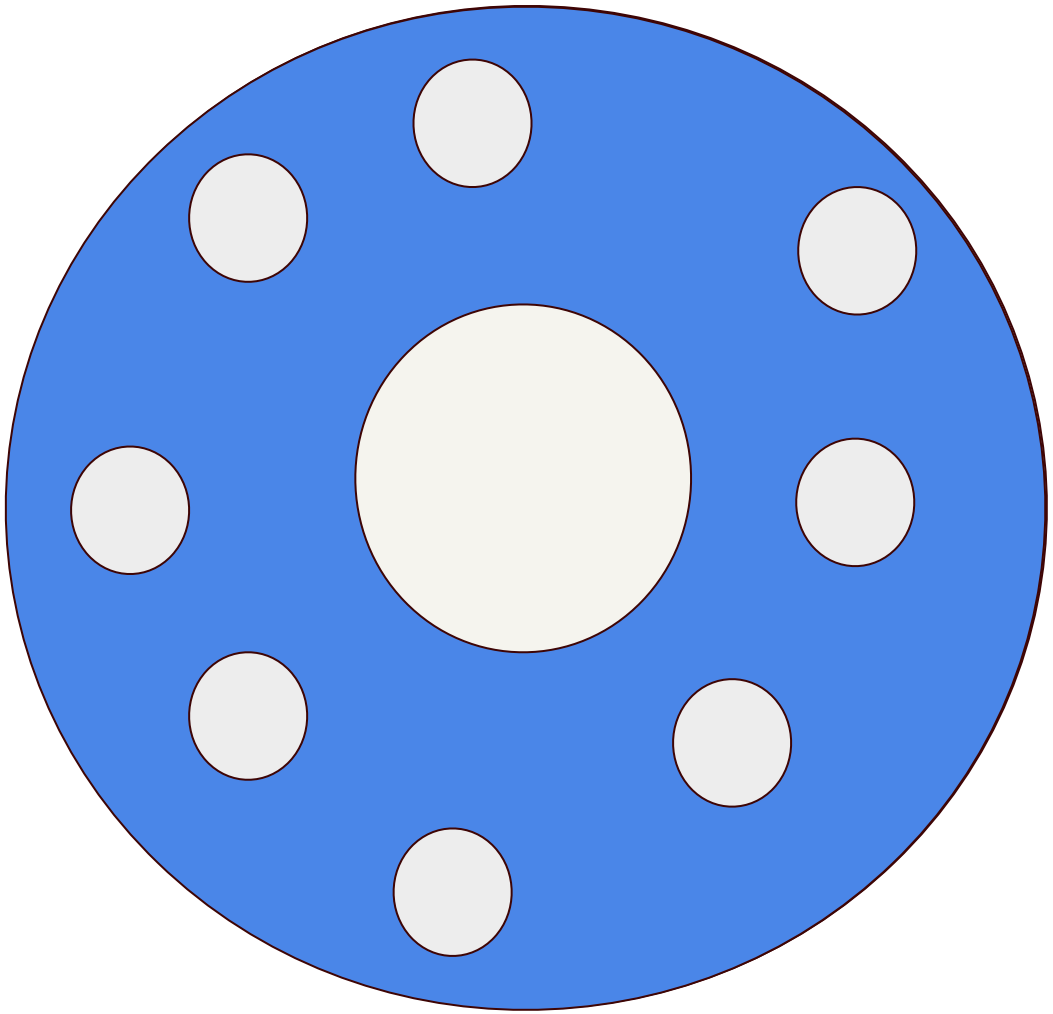
# Donut & Donut Hole Cache - Considerations

- Remember to cache by param etc.
- Caches everything!
- Cache not close to the problem?
- Harder to diagnose performance problems

```
public class ProfileController : Controller
{
    [OutputCache(CacheProfile="Cache1Hour")]
    public string Index()
    {
        return DateTime.Now.ToString("T");
    }
}
```

# Donut Caching





# Donut Caching

```
@Html.Action("Login", "Account", true)
```

excludeFromParentCache

```
[DonutOutputCache(Duration=60)]
public ActionResult Index()

[DonutOutputCache(CacheProfile="TwoMins")]
public ActionResult Index()

[DonutOutputCache(Duration=60, VaryByCustom="whatever")]
public ActionResult Index()

[DonutOutputCache(Duration=60, VaryByParam="something;that")]
public ActionResult Index(string something)
```

# Donut & Donut Hole Cache - Considerations

- Remember to punch holes
- Cache not close to the problem?
- Harder to diagnose performance problems

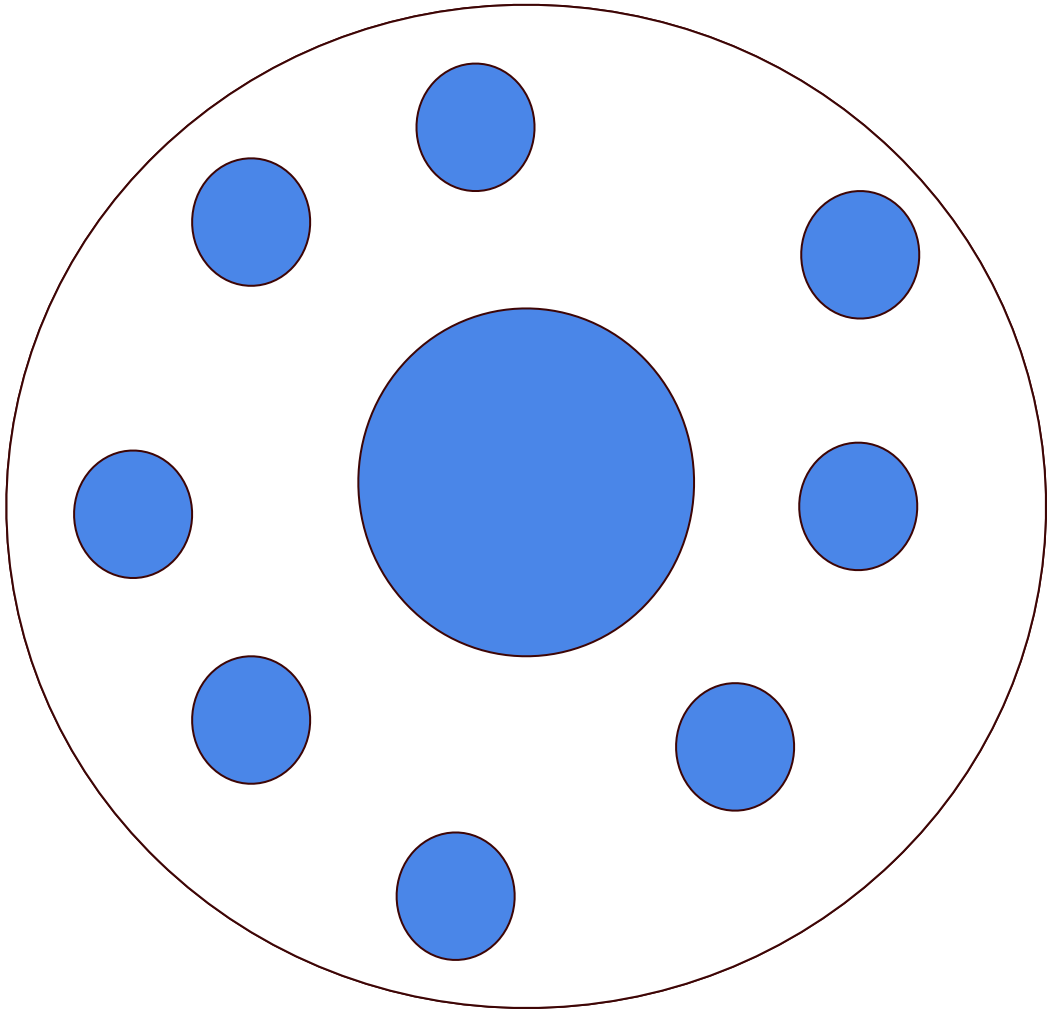
```
[DonutOutputCache(Duration=60)]
public ActionResult Index()

[DonutOutputCache(CacheProfile="TwoMins")]
public ActionResult Index()

[DonutOutputCache(Duration=60, VaryByCustom="whatever")]
public ActionResult Index()
```

# Donut Hole Caching





# Donut Hole Caching



```
<div>@Html.Action("CategoriesList")</div>
```

```
[ChildActionOnly]
[OutputCache(Duration=60)]
public ActionResult CategoriesList()
{
 // Get categories list from the database and
 // pass it to the child view
 ViewBag.Categories = Model.GetCategories();
 return View();
}
```

# Output, Donut & Donut Hole - Considerations

- Cache not close to the problem?
- Harder to diagnose performance problems
- No granular control

```
[ChildActionOnly]
[OutputCache(Duration=60)]
public ActionResult CategoriesList()
{
 // Get categories list from the database and
 // pass it to the child view
 ViewBag.Categories = Model.GetCategories();
```

# Custom Method Caching



```
public object GetValue(string id)
{
    var value = GetSomethingExpensive(id);

    return value;
}
```

# Custom Method Caching

```
public object GetValue(string id)
{
    var cacheKey = id;

    object value = _cache.Get<object>(cacheKey);

    if (value == null)
    {
        value = GetSomethingExpensive(id);
        _cache.Add(cacheKey, value, new TimeSpan(ONE_HOUR));
    }

    return value;
}
```

# Custom Method Caching

- Not single responsibility
- Repeated code
- Where is cache used?
- How to clear / turn off?
- Diagnose performance?


```
public object GetValue(string id)
{
    var cacheKey = id;

    object value = _cache.Get<object>(cacheKey)

    if (value == null)
    {
        value = GetSomethingExpensive(id);
        _cache.Add(cacheKey, value, new TimeSpa
    }

    return value;
}
```

# Service Layer - IoC Cached Proxy Pattern - Interface

radley yeldar.

- Dependency Injection & Interfaces
- Service Layer
- Cached proxy
- Replace concreate implementation at startup

# Service Layer - IoC Cached Proxy Pattern - Interface



```
public interface IExampleService
{
    object GetValue(string id);
}
```

```
public class ExampleService : IExampleService
public class ExampleServiceCachedProxy : IExampleService
```

# Service Layer - IoC Cached Proxy Pattern - Registration

```
private static void RegisterCachedServices(Composition composition)
{
    if (ConfigurationHelper.IsServiceCacheEnabled())
    {
        composition.Register(typeof(ExampleService), typeof(ExampleService));

        composition.Register(typeof(ICache), typeof(Cache));
        composition.Register(typeof(IExampleService), typeof(ExampleServiceCachedProxy));
    }
    else
    {
        composition.Register(typeof(IExampleService), typeof(ExampleService));
    }
}
```

```
<add key="ServiceCache:Enabled" value="true" />
```

# Service Layer - IoC Cached Proxy Pattern - Proxy Method

```
public object GetValue(string id)
{
    var value = GetSomethingExpensive(id);

    return value;
}
```

```
public object GetValue(string id)
{
    var cacheKey = $"{typeof(ExampleServiceCachedProxy)}_{id}";

    return _cache.Get(cacheKey, () => _exampleService.GetValue(id));
}
```

# Service Layer - IoC Cached Proxy Pattern - Cache Purging

```
public void Initialize()
{
    ContentService.Published += ContentService_Published;
}


private void ContentService_Published(IContentService sender,
    ContentPublishedEventArgs e)
{
    _cache.RemoveByPrefix(typeof(ExampleServiceCachedProxy).ToString());
}
```

# IoC Cached Proxy Pattern - Considerations

- Learning curve can be hard
- IoC init get be bloated
- Some methods only pass through
- No way to manually clear

# Data Repository Layer

```
internal class DefaultRepositoryCachePolicy<TEntity, TId> : RepositoryCachePolicyBase<TEntity, TId>
    where TEntity : class, IEntity

public override TEntity Get(TId id, Func<TId, TEntity> performGet, Func<TId[], IEnumerable<TEntity>> perfo
{
    var cacheKey = GetEntityCacheKey(id);
    var fromCache = Cache.GetCacheItem<TEntity>(cacheKey);

    // if found in cache then return else fetch and cache
    if (fromCache != null)
        return fromCache;
    var entity = performGet(id);

    if (entity != null && entity.HasIdentity)
        InsertEntity(cacheKey, entity);

    return entity;
}
```

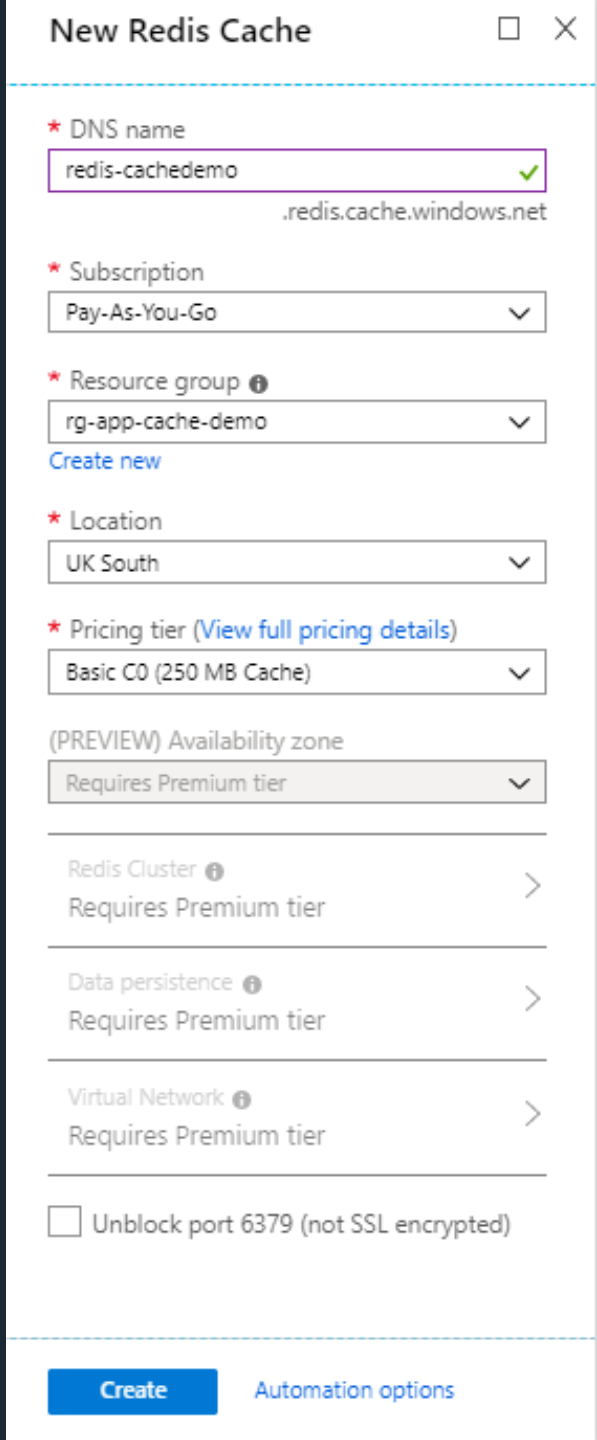
# Data Repository Layer - Considerations

- All previous problems
- Stale entities – Concurrent write & reads?
- Difficult to distribute (e.g. multi web-heads)

Consider your architecture first!

# Redis

- External Key-Value storage service
- Great alternative to local memory
- Great for large amounts of data

# Redis



| CACHE NAME | CACHE SIZE | NETWORK PERFORMANCE | NUMBER OF CLIENT CONNECTIONS | PRICE |
|---|---|---|---|---|
| C0 | 250 MB | Low | 256 | ~£14.963/month |
| C1 | 1 GB | Low | 1,000 | ~£37.542/month |
| C2 | 2.5 GB | Moderate | 2,000 | ~£60.937/month |
| C3 | 6 GB | Moderate | 5,000 | ~£122.418/month |
| C4 | 13 GB | Moderate | 10,000 | ~£142.548/month |
| C5 | 26 GB | High | 15,000 | ~£285.640/month |
| P1 | 6 GB | Moderate | 7,500 | ~£150.709/month |
| P2 | 13 GB | High | 15,000 | ~£301.962/month |
| P3 | 26 GB | High | 30,000 | ~£603.380/month |
| P4 | 53 GB | Highest | 40,000 | ~£1,207.848/month |

# Data Repository Layer - Considerations

- Cost
- More infrastructure
- Uptime

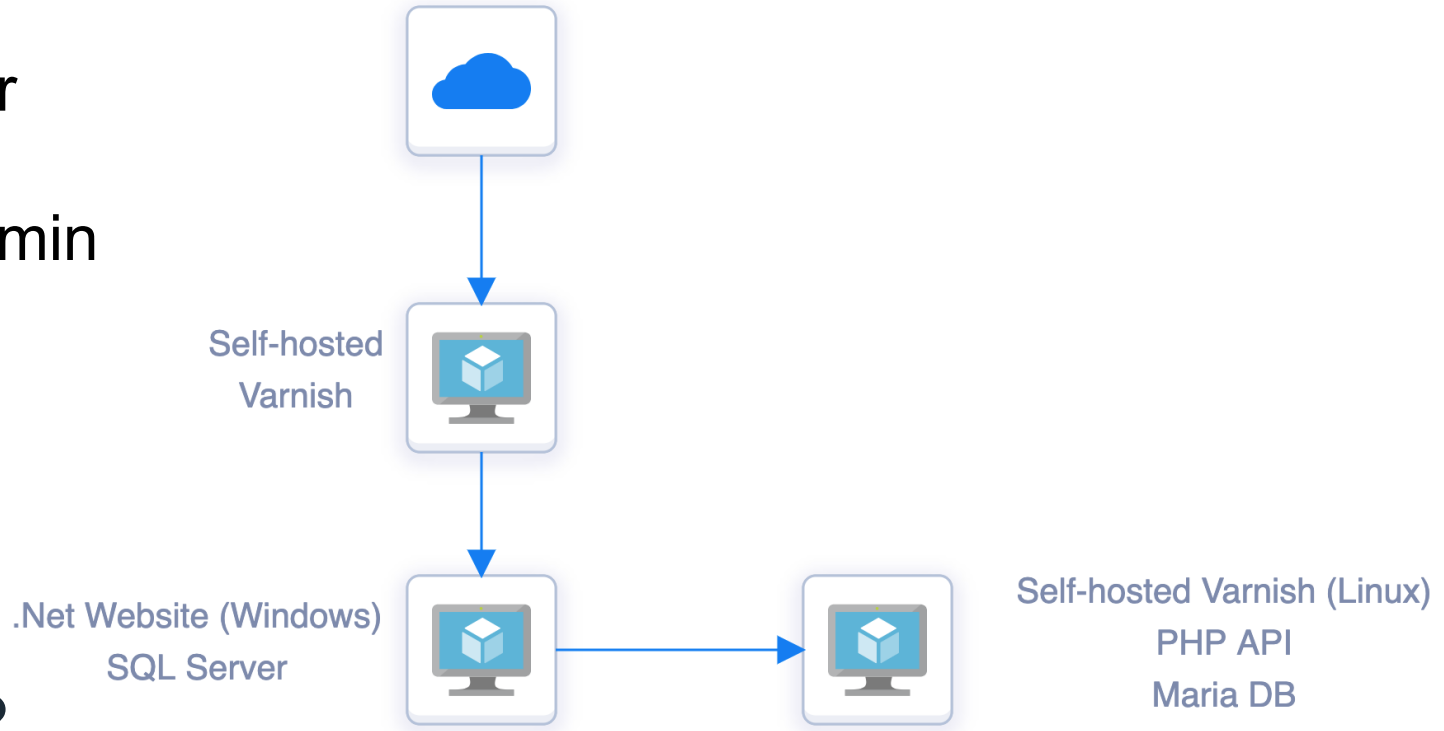
redis

# Multi-layered Caching - Real world example



# Multi-layered Caching

| | |
|---|---|
| Varnish (VM 1) | 24hr |
| Website: Memory Cache (VM 2) | 1hr |
| Varnish (VM 3) | 1hr |
| API: Random Memory Cache (VM 3) | 10min |

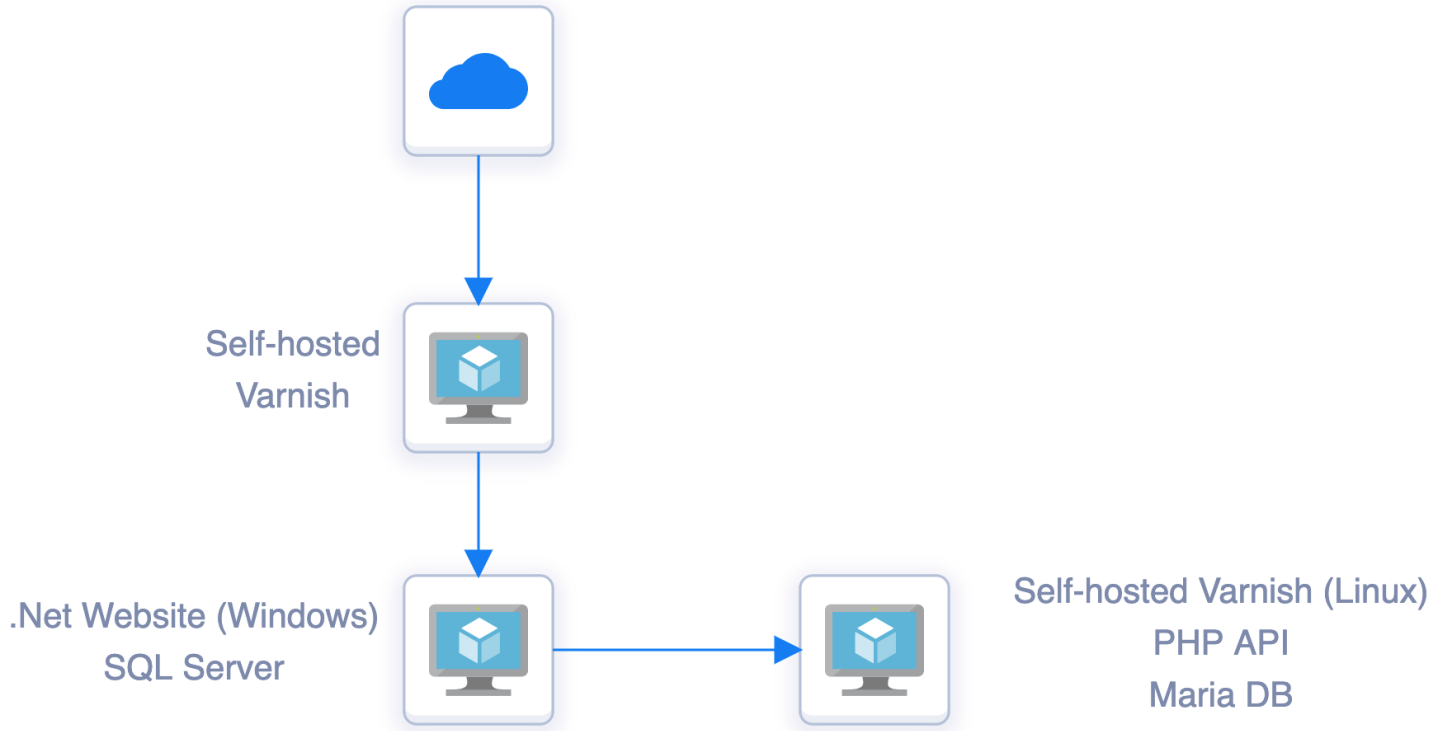



## Considerations

- What is the actual cache time?
- How to clear all the cache?

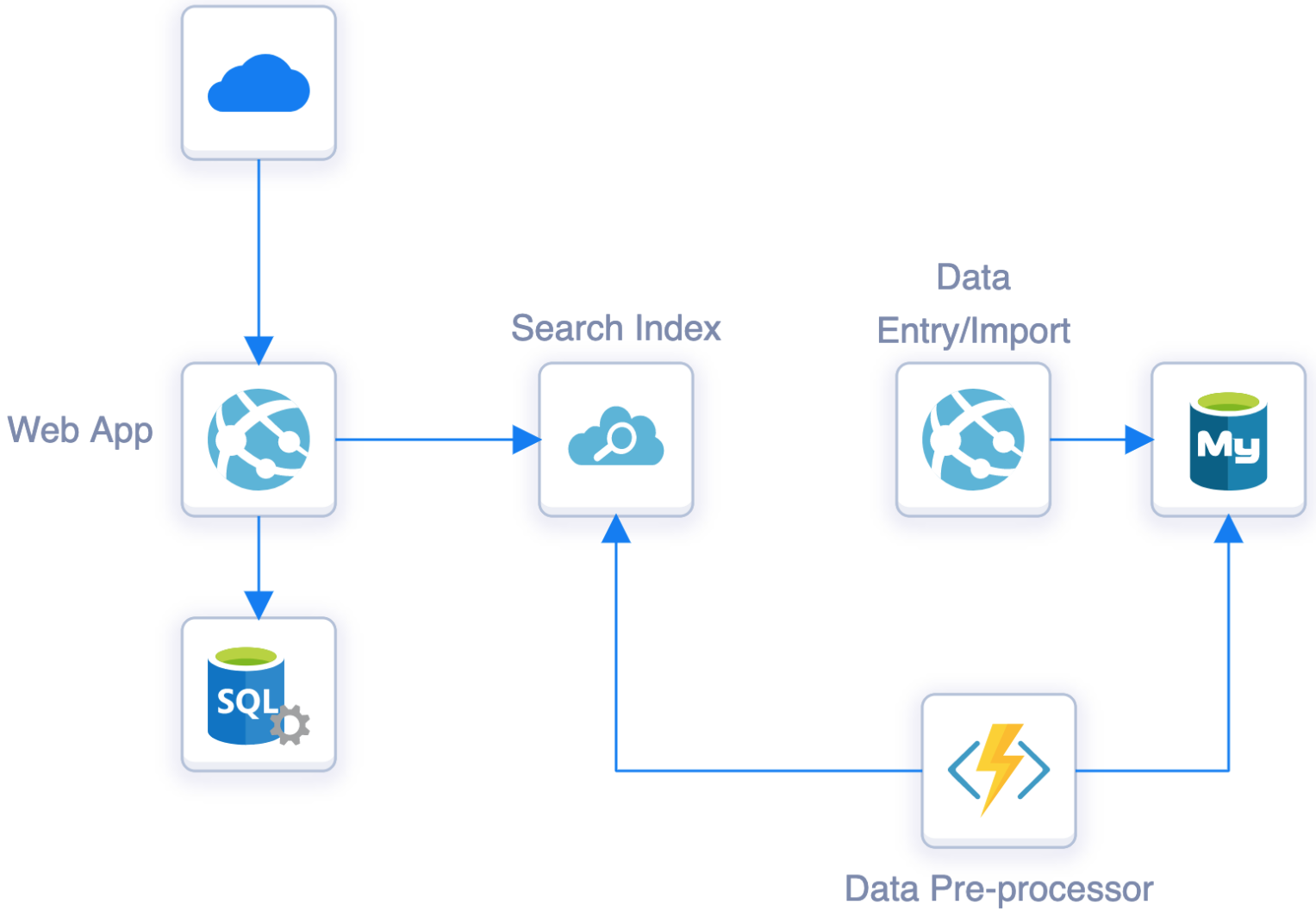
# Multi-layered Caching - A better solution?

- What was the problem?
- Complex / inefficient SQL Queries



# Multi-layered Caching - A better solution?

- Pre-process
- Index



# More infrastructure => More downtime

## 1 Service (99.95%)

21m 54.9s per month

## 2 Services (99.95% each)

99.95 x 99.95 = 99.90%: 43m 49.7s

## 3 Services (99.95% each)

99.95 x 99.95 x 99.95 = 99.85%: 1h 5m 44.6s

# Multi-layered Caching - Considerations

- Can add instability/downtime
- Clearing multiple layers simultaneously
- Pre-index up to date?

Cache is not the answer to everything!!

# Caching Options - Recap

DNS/Proxy

Network

Page Output

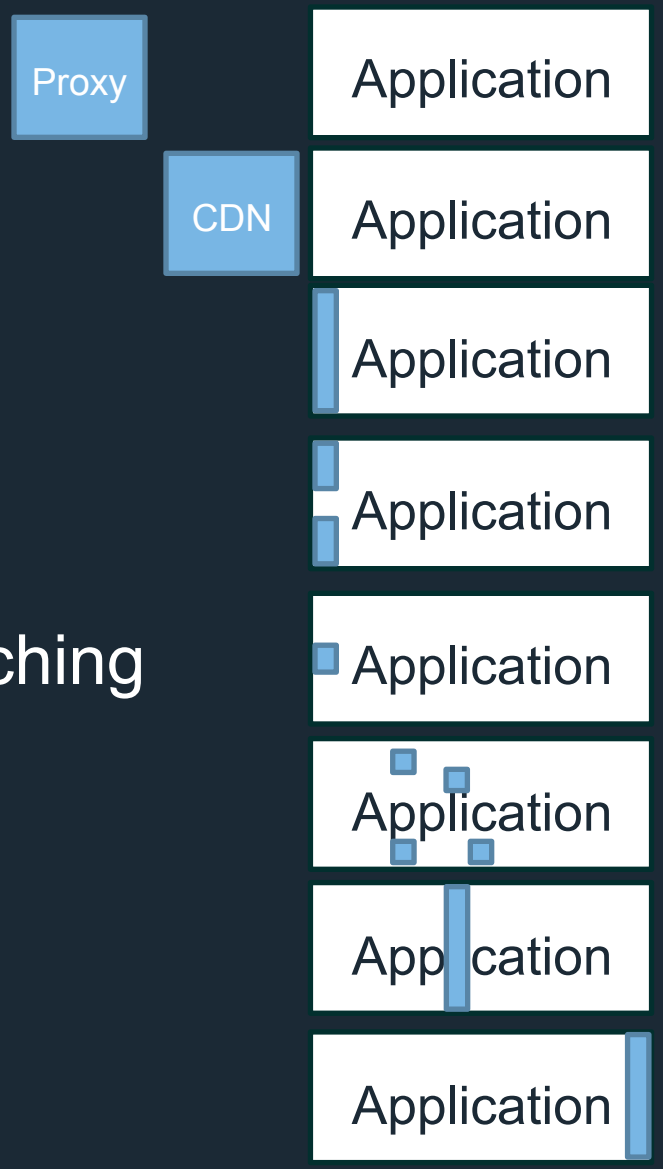
Donut Caching

Donut Hole Caching

Methods

Service layer

Data layer




## Pre-Indexing


## Multi-layered

# Recommendations

- Cache as a last resort
- Favour pre-processing & indexing
- Automatic & manual clearing
- Avoid caching everything
- Cache close to the problem
- Keep it simple and obvious

Multiple layers => multiple headaches

No perfect solution, only trade-offs

# Cache me on Twitter @anthonydotnet

# Git Kraken: anthonyd19