CREATE CUSTOM CACHE

Create a file under custom module:

etc/cache.xml

```
<?xml version="1.0"?>
<config
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="urn:magento:framework:Cache/etc/cache.xsd">
    <type name="shop_by_designer"
instance="VENDOR_NAME\MODULE_NAME\Model\Cache\Type">
        <label>Custom Cache</label>
        <description>Custom Cache
description</description>
    </type>
</config>
```

Model/Cache/Type.php
This would register custom cache under your system/cache management in admin panel.

```
<?php
namespace
VENDOR_NAME\MODULE_NAME\Model\Cache;


class Type extends
\Magento\Framework\Cache\Frontend\Decorator\TagSc
ope
{
    /**
     * Cache type code unique among all cache types
     */
    const TYPE_IDENTIFIER = ‘custom_cache’;


    const CACHE_TAG = ‘CUSTOM_CACHE’;


    const CACHE_KEY = ‘custom_cache’;
    /**
     * @param \Magento\Framework\App\Cache\Type\FrontendPool $cacheFrontendPool
     */
    public function __construct(\Magento\Framework\App\Cache\Type\FrontendPool $cacheFrontendPool)
    {
        parent::__construct($cacheFrontendPool->get(self::TYPE_IDENTIFIER), self::CACHE_TAG);
    }
}
```

Next step is to save cache data:

Usually write cache storage in HELPER. EASY TO ACCESS FROM ANYWHERE

Helper/Data.php

Define local variable

```
const LIFETIME = 86400; //it can be anything, based on your data

/**
    * @var \Magento\Framework\Serialize\Serializer\Serialize
    */
private $serialize;


/**
    * @var \Magento\Framework\App\CacheInterface
    */
private $cache;
```

define constructor dependency

```
\Magento\Framework\App\CacheInterface $cache
\Magento\Framework\Serialize\Serializer\Serialize $serialize,
$this->cache = $cache;
$this->serialize = $serialize;
```

Get cache KEY and TAG

```
$cacheKey =
\VENDOR_NAME\MODULE_NAME\Model\Cache\Type:
:CACHE_KEY;
        $cacheTag =
\VENDOR_NAME\MODULE_NAME\Model\Cache\Type:
:CACHE_TAG;
```

AND SAVE DATA LIKE

```
$this->cache->save(
                $this->serialize->serialize($data),
                $cacheKey,
                array($cacheTag),
                self::LIFETIME
            );
```

RETRIEVE DATA:

```
$data = $this->serialize->unserialize($this->cache->load($cacheKey));
```