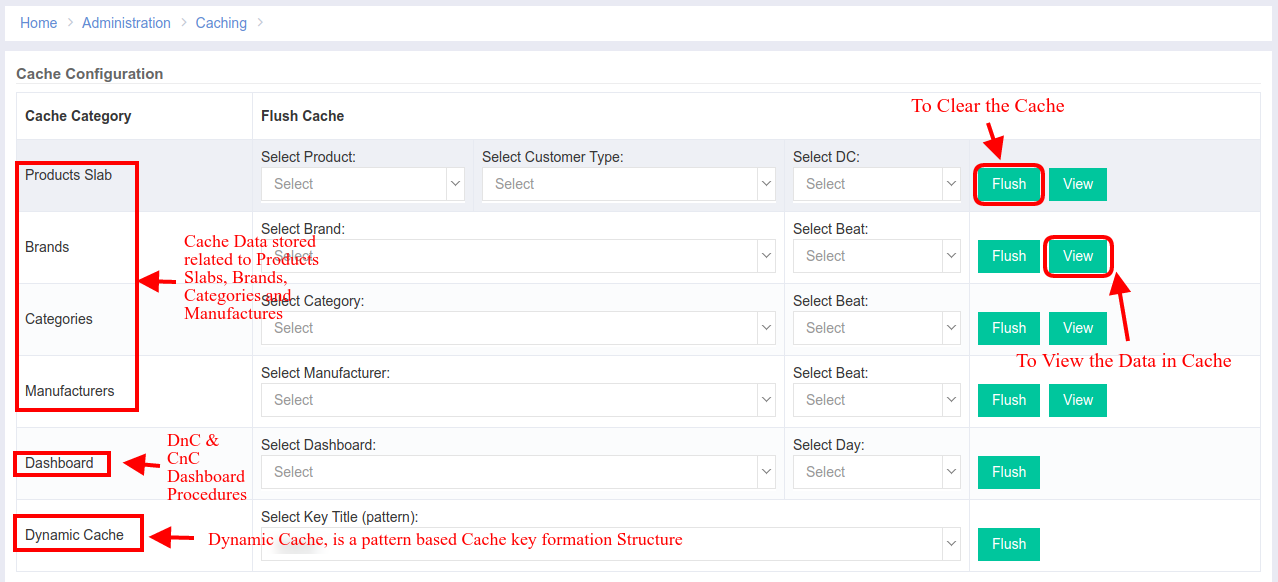
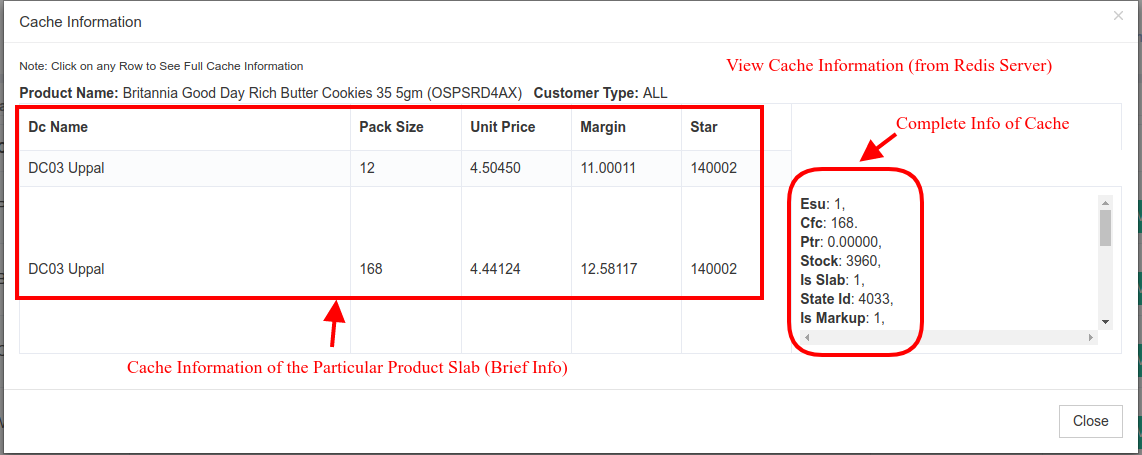
Caching

A **cache** is a temporary storage area. For example, the database products you automatically request by looking at a Mobile are stored on Redis in a cache subdirectory. When you return to a page you've recently looked at, the mobile can get those files from the redis cache rather than the original database, saving you time and saving the network the burden of additional traffic.  
We have the advantage of setting the time in cache, which is the expiry limit of the Cache.  
  
  
  
View Cache Information (from Redis Server):  
  
  
When there is a request from the Mobile for a list of products or brands or categories or manufacturers, then the mobile api would hit Cache based on the key which is created by ids.  
Then, the Cache will check the key in the redis server, if found it will return if not the php code will fetch it from database, and then will put into the Cache Redis and throws it the api as response. The flow looks simple, with the use Cache facade of Laravel.  
  
Code Details:  
Module Name: **Caching**  
URL: <http://portal.ebutor.com/cache>  
Features:  
 Access Cache Flush Permission [**CASHFLUSH**]  
  
Things to do remember:  
Cache mechanism uses 2 pre defined functions

Cache::**get**(‘key\_string’) and  
Cache::**put**(‘key\_string’,‘data’,time)

**‘key\_string’ =>** Key string is the unique key for every cache element.

**‘data’** => Data is an object or array which is stored in the redis database.

**‘time’** => Time is an integer which is default in minutes.  
  
Code Structure:  
**The Products Slab Component**: The cache for the products slab is get and put in Cpmanager products controller and in the caching module, we will just flush and view in the cachin module

**The Brands/Categories/Manufacturers Component**: The cache for all these 3 tabs are similar and I used same code, the key strings of these 3 components end with their own component names.

**The Dashboard Component**: The cache for the dashboard component is generated at the home page of the ebutor (DnC Business and CnC Business). There are around 11 tabs at the bottom of the dashboard grid and all those tabs (including dashboard grid) is retrieved from the cache and the time limit is 15 minutes.  
Specially for the dashboard component I used **cache tags** structure as to differentiate the DnC and CnC Dashboards Cache Data.  
**The Dynamic Cache Component**: This component is something different than all other prescribed above. This has been designed as for the scope of future code. If any developer wants to store the cache code in a particular pattern based key string, then he/she can hard code it in the code value and with an entry of the same in database table **dynamic\_cache\_keys** anyone can flush that cache memory, without the need of any flush logic written for it.  
  
Code Flow:  
When we hit the url then it will be called to **index()** and in that function, we retrieve and populate all the data like products, brands, categories etc.,   
The **flushProductsSlab()** function is used to check the cache for the particular product and it will call **flushSingleProduct()** to flush(clear) the product cache in single.   
The **viewProductsSlab()** is used to generate a user interface to the cache data,