Permission Check Performance Optimization with Caching

*Issue:*

Now we have data level permission check which will call database recursively to do complex evaluation before real business logic every time. The DB access is expensive and unnecessary when the permission is already checked for one record.

*Solution:*

We added caching mechanism to permission check component so that we only do one time check at the first time.

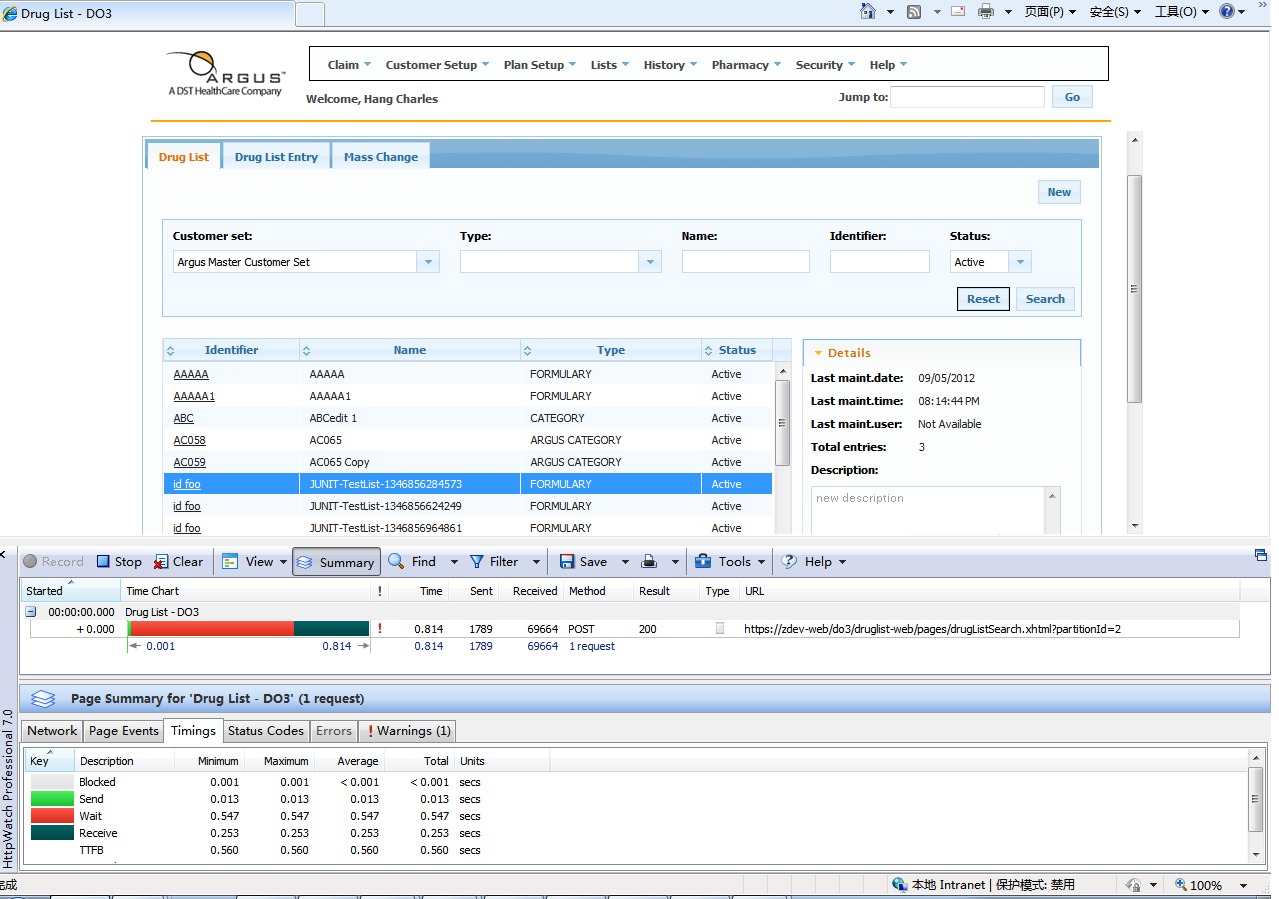
*Test Result:*

Below are some comparison of main scenarios after the optimization. Note that the total time is ignored here since we have a very unstable network today and the network transfer time is useless here.

Note:

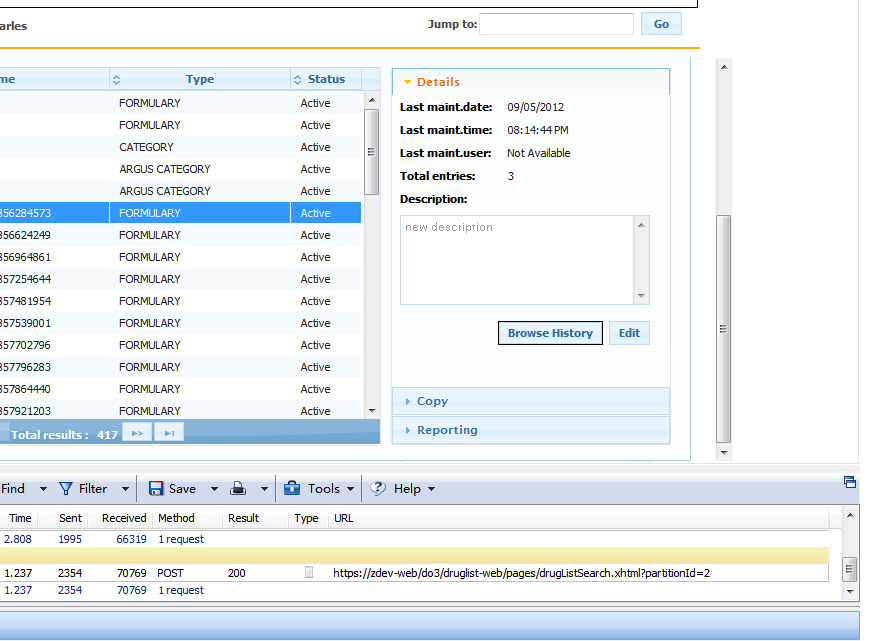
* The yellow background color section is the time spent with permission check caching.
* The white background color section is the time spent without permission check caching.

1. Click a druglist record in the druglist result table.



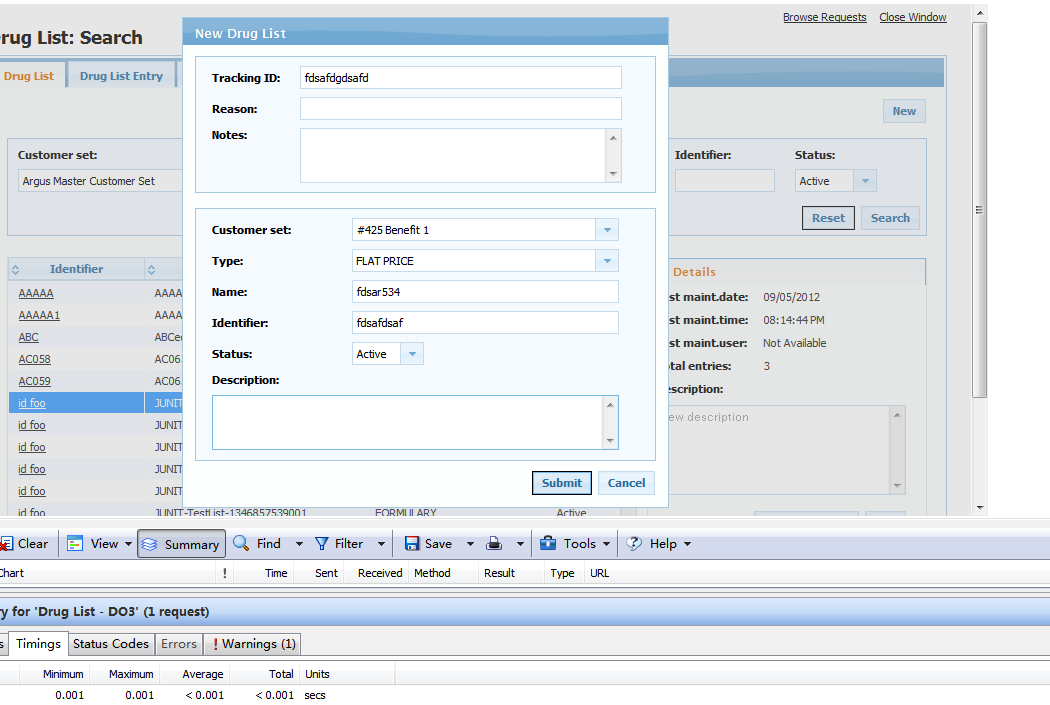
|  |  |  |
| --- | --- | --- |
|  | Backend(wait time in Http Watch) | |
|  | Without Permission Cache | With Permission Cache |
| 1 | 0.748 | 0.644 |
| 2 | 1.116 | 0.535 |
| 3 | 0.976 | 0.547 |
| 4 | 0.879 | 0.584 |
| 5 | 1.008 | 0.867 |
| 6 | 0.601 | 0.574 |
| 7 | 0.647 | 0.592 |
| 8 | 0.714 | 0.522 |
| 9 | 0.564 | 0.918 |
| 10 | 1.172 | 0.794 |
| AVG | 0.8425 | 0.6577, reduced by 22% |

1. Click the Detail accordion in the right side of the druglist result table.



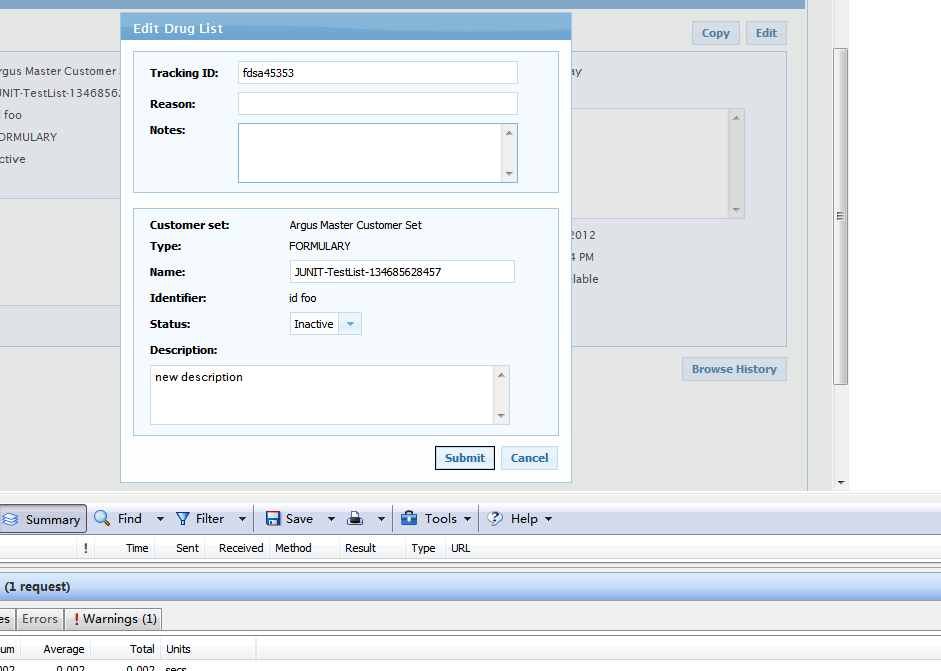
|  |  |  |
| --- | --- | --- |
|  | Backend(wait time in Http Watch) | |
|  | Without Permission Cache | With Permission Cache |
| 1 | 0.945 | 0.516 |
| 2 | 0.697 | 0.521 |
| 3 | 0.854 | 0.516 |
| 4 | 0.732 | 0.509 |
| 5 | 0.956 | 1.093 |
| 6 | 0.683 | 0.808 |
| 7 | 0.678 | 0.554 |
| 8 | 1.442 | 0.733 |
| 9 | 0.782 | 0.623 |
| 10 | 0.735 | 0.579 |
| AVG | 0.8504 | 0.6452, reduced by 24% |

1. Click Submit button to create a new druglist on the New Drug List pop up.



|  |  |  |
| --- | --- | --- |
|  | Backend(wait time in Http Watch) | |
|  | Without Permission Cache | With Permission Cache |
| 1 | 3.651 | 2.434 |
| 2 | 1.658 | 2.882 |
| 3 | 2.701 | 1.891 |
| 4 | 7.238 | 1.6 |
| 5 | 1.701 | 1.706 |
| 6 | 3.501 | 1.732 |
| 7 | 3.373 | 1.595 |
| 8 | 1.703 | 1.713 |
| 9 | 1.584 | 1.697 |
| 10 | 1.882 | 1.614 |
| AVG | 2.8992 | 1.8864, reduced by 35% |

1. Click Submit button to update a druglist on the Edit Drug List pop up.



|  |  |  |
| --- | --- | --- |
|  | Backend(wait time in Http Watch) | |
|  | Without Permission Cache | With Permission Cache |
| 1 | 1.067 | 0.821 |
| 2 | 1.253 | 0.77 |
| 3 | 0.965 | 0.833 |
| 4 | 0.636 | 0.639 |
| 5 | 0.651 | 0.667 |
| 6 | 1.084 | 0.602 |
| 7 | 0.727 | 0.619 |
| 8 | 2 | 0.676 |
| 9 | 1.423 | 0.629 |
| 10 | 1.041 | 0.619 |
| AVG | 1.0847 | 0.6875, reduced by 37% |

*Summary:*

This solution has overall performance improvement on not only Druglist, but all the other existing re-engineered applications. It may not solve a specific issue since most of issues are caused by front-end in Druglist, but it is believed to have architectural benefit on all applications.