<img src="media/image1.jpg" style="width:2.02569in;height:1.51944in"  
alt="BDlogo" />\*\*<span class="smallcaps">  
</span>\*\*  
  
\*\*<span class="smallcaps">PERFORMANCE OPTIMIZATION REPORT</span>\*\*  
  
\*\*<span class="smallcaps">FOR</span>\*\*  
  
\*\*<span class="smallcaps">COMBINED SYSTEM DEVELOPMENT SERVICES</span>\*\*  
  
\*\*<span class="smallcaps">FOR</span>\*\*  
  
\*\*<span class="smallcaps">LICENSING SELF-CERTIFICATION PORTAL</span>\*\*  
  
\*\*<span class="smallcaps">OF</span>\*\*  
  
\*\*<span class="smallcaps">BUILDINGS DEPARTMENT</span>\*\*  
  
\*\*Version 0.1\*\*  
  
\*\*Jan 2025\*\*  
  
? The Government of the Hong Kong Special Administrative Region  
  
The contents of this document remain the property of and may not be  
reproduced in whole or in part without the express permission of the  
Government of the HKSAR.  
  
<table>  
<colgroup>  
<col style="width: 28%" />  
<col style="width: 71%" />  
</colgroup>  
<thead>  
<tr class="header">  
<th colspan="2"><strong>Distribution</strong></th>  
</tr>  
<tr class="odd">  
<th>Copy No.</th>  
<th>Holder</th>  
</tr>  
<tr class="header">  
<th>1</th>  
<th><blockquote>  
<p>Buildings Department (BD)</p>  
</blockquote></th>  
</tr>  
<tr class="odd">  
<th>2</th>  
<th><blockquote>  
<p>Master Concept (Hong Kong) Limited (MC)</p>  
</blockquote></th>  
</tr>  
</thead>  
<tbody>  
</tbody>  
</table>  
  
| Amendment History | | | | |  
|---------|------------------------------|-----------|----------|-------------|  
| Change Number | Revision Description | Pages Affected | Revision/ Version Number | Date |  
| 1 | 1st draft | All | 0.1 | 16/01/2025 |  
| | | | | |  
| | | | | |  
| | | | | |  
| | | | | |  
  
\*\*TABLE OF CONTENTS\*\*  
  
[\*\*1. Introduction 5\*\*](#introduction)  
  
> [1.1 Goal of Performance Optimization  
> 6](#goal-of-performance-optimization)  
>  
> [1.1.1 Server Loading 6](#server-loading)  
>  
> [1.1.2 Bandwidth Usage 6](#bandwidth-usage)  
>  
> [1.1.3 Better user experience 6](#better-user-experience)  
>  
> [1.2 Performance Optimisation Actions  
> 7](#performance-optimisation-actions)  
>  
> [1.3 Storage Allocation 8](#storage-allocation)  
>  
> [1.4 Required Response Time 9](#required-response-time)  
>  
> [1.4.1 Online Transaction 9](#online-transaction)  
>  
> [1.4.2 Online Reports 10](#online-reports)  
  
[\*\*2. Critical online transition timing  
11\*\*](#critical-online-transition-timing)  
  
[\*\*3. Critical Batch Cycle Timing 12\*\*](#critical-batch-cycle-timing)  
  
[\*\*4. Optimization changes 13\*\*](#optimization-changes)  
  
> [4.1 Optimization Actions 14](#optimization-actions)  
>  
> [4.1.1 Database Indexing 14](#database-indexing)  
>  
> [4.1.2 Query Optimization 14](#query-optimization)  
>  
> [4.1.3 Data Archiving 15](#data-archiving)  
  
##  
  
## 1. Introduction  
  
This Performance Optimization Report outlines strategies and considerations for enhancing the performance of the Licensing Self-Certification Portal. The system's performance is crucial for user satisfaction and the overall efficiency of the Buildings Department's operations. This report is based on an analysis of the database schema and general performance optimization principles.  
  
\*\*Database Overview (as of 2025/3/4 ??10:10:39):\*\*  
  
- \*\*Database Name:\*\* bd  
- \*\*Database Size:\*\* 88.10 MB  
- \*\*Collections:\*\* 12  
- \*\*Total Documents:\*\* 1278983  
- \*\*Total Data Size:\*\* 371.24 MB  
  
The database comprises 12 collections, with 'sysfilerefs' and 'adrblkfilerefs' being the largest in terms of document count and data size, indicating they are potentially critical for performance considerations.  
  
The performance optimization of the system could be classified into  
optimization of Online Transaction.  
  
## 1.1 Goal of Performance Optimization  
  
The main goal of performance optimization is to improve the response  
time of the system to users. In order to achieve better response time,  
the program implementation should take the following areas into  
consideration.  
  
### 1.1.1 Server Loading  
  
The capacity of Server Loading is a fixed variable of a system. An  
increase in server loading would increase the response time of the  
system. Server loading is affected by  
  
1. the number of system users; and  
  
2. the efficiency of the programming code.  
  
When the number of system users increases, the server loading increases  
and hence there is less resource for running programming code for each  
system user. The response of the program would be slower as there is  
less server resource for the program to utilize.  
  
On the other hand, if the programming code could use fewer server  
resources, there is less server loading and hence the server could  
respond to users quicker and serve more users simultaneously.  
  
Therefore, response time could be reduced if the programming code could  
be optimized to use fewer server resources.  
  
### 1.1.2 Bandwidth Usage  
  
Capacity of Bandwidth Usage is a fixed variable of a system. An increase  
in bandwidth usage would decrease the response time of the system.  
Bandwidth loading is affected by  
  
1. the number of system users; and  
  
2. the size of transmitting resource.  
  
When the number of system users increases, the bandwidth usage increases  
and hence there is less bandwidth resource for transmitting data to each  
system user. The response time would be slower as there is less  
bandwidth for each user to utilize.  
  
On the other hand, if the size of transmitting data is smaller, there is  
less bandwidth usage and hence the fixed network bandwidth could serve  
more users simultaneously.  
  
Therefore, response time could be reduced if the size of the  
transmitting resource could be optimized to use less server resources.  
  
### 1.1.3 Better User Experience  
  
This application will be used by the public, in order words, it  
represents the department. If it performs well, the department can take  
credit from it, the image of the department may be affected if this  
application performs bad or causes other problems, for example very slow  
or no response.  
  
## 1.2 Performance Optimisation Actions  
  
To serve more users simultaneously with better response time, the system  
and the programs should be optimized to use the server loading and  
bandwidth resource more efficiently. The followings are the possible  
measures that could be taken generally:  
  
- Optimize the programming and query logic to reduce server loading  
 burden.  
- Optimize the page size and image size to reduce bandwidth burden.  
- Improve resource retrieval speed by indexing and hashing.  
- Cache frequently used resources.  
- Pre-generate resource that required heavy instant server loading.  
- Reduce waiting time of third-party services.  
- Archive expired records to keep the size of active datastore minimal.  
  
## 1.3 Storage Allocation  
  
The storage of the database data will be stored as following:  
  
i. System data ? Store in the ?Database? server in the Integrated  
system.  
  
ii. Textual data ? Store in the ?Database? server in the Integrated  
system.  
  
The system utilizes a MongoDB database. Based on the database analysis, the following collections are the largest in size and document count and should be considered for storage optimization and performance tuning:  
  
| Collection Name | Document Count | Size (MB) | Average Document Size (KB) |  
|-------------------|----------------|-----------|----------------------------|  
| sysfilerefs | 601808 | 204.70 | 0.35 |  
| adrblkfilerefs | 566948 | 154.89 | 0.28 |  
| bsblocks | 98397 | 6.40 | 0.07 |  
| tasks | 5523 | 0.99 | 0.18 |  
| cases | 451 | 1.17 | 2.65 |  
| oauthtokens | 3019 | 2.29 | 0.78 |  
| notifications | 1837 | 0.24 | 0.13 |  
| applications | 381 | 0.36 | 0.96 |  
| attachments | 370 | 0.13 | 0.37 |  
| users | 116 | 0.04 | 0.39 |  
| eminutes | 133 | 0.03 | 0.24 |  
| submissions | 0 | 0.00 | 0.00 |  
  
  
Optimizing storage for the 'sysfilerefs' and 'adrblkfilerefs' collections will have the most significant impact on overall database performance due to their size. Strategies such as data archiving and efficient indexing should be considered for these collections.  
  
## 1.4 Required Response Time  
  
The required response time is defined according to 2 pre-defined  
categories. They are Online Transaction and Online Report. All of the  
programs and reports that require to interact and provide immediate  
response to users are categorized. The categorized programs and reports  
should respond to the user within required response time. The required response time should be defined according to the complexity of the programs. The required response time and the complexity will be  
discussed in the coming section.  
  
For those procedures or reports that could not able to optimize to have  
immediate response, some part of the program that take a lot of  
processing time will be swapped to batch job.  
  
95% of all online functions should meet the committed response time.  
  
The committed response time is affected by the network status of the  
site. The environment of testing site should meet an agreed network  
health level. The following criteria define the agreed network health  
level.  
  
- Maximum number of Concurrent users is 100.  
- Minimal bandwidth is 2Mb/s per testing machine.  
- Maximum network round-trip latency (ping) to the Integrated system  
 is 200ms.  
- Remote testing site will have \*\*50% mark-up\*\* time to the committed  
 response time.  
  
### 1.4.1 Online Transaction  
  
The programs in the following category are classified as online  
transaction:  
  
- User Account Program  
- Form and Record Management Program  
  
Online transactions can be classified into the following groups:  
  
<span class="mark">\[RY Note: Needs user input/ refer to Load Test data  
given by user]</span>  
  
| Transaction Complexity | Number of Concurrent Users | | | |  
|---------------------|-----------------|-----------------|-----------------|--|  
| | 40 | 80 | 100 | |  
| Simple | \< 1 sec | \< 2 sec | \< 3 sec | |  
| Medium | \< 2 sec | \< 3 sec | \< 4 sec | |  
| Complex | \< 2.5 sec | \< 3.5 sec | \< 5 sec | |  
  
Online transactions can also be classified as follows:  
  
| Online transactions | Description |  
|---------------------------------|---------------------------------------|  
| Online Update Transactions | Used to update records in LSCP, for example, create supervision plan |  
| Online Enquiry Transactions | Used to retrieve records from LSCP, for example, filter site monitoring records |  
| Full-text Search | Used to search for records with given key words, for example, search assigned TCP |  
  
###  
  
### 1.4.2 Online Reports  
  
The programs in the following category are classified as online reports:  
  
- XXXXXXXXXXX  
  
As the report is a standalone internal system for BD, the measurement  
would only estimate when the concurrent users are not more than 20.  
  
| On-line Reports | Committed Response Time |  
|-----------------|-------------------------|  
| | |  
| | |  
  
#  
  
# 2. Critical Online Transition Timing  
  
The following matrix is the list of programs with the complexity and  
transaction type being marked.  
<br/>  
Offline module helps users save information in local devices when  
Internet connectivity is not available. Once Internet connectivity is  
back, it will submit the data and store it in the database. The major  
difference between offline modules with other modules is that the  
offline module is asynchronous. It will not submit immediately but will  
wait until the Internet is available. For the asynchronous handling  
logic, we included it in the functional test. On the other hand, the  
data sending logic is no different with other API, thus we would not  
have a separate section for offline module in critical online transition  
timing.  
  
<table>  
<colgroup>  
<col style="width: 8%" />  
<col style="width: 8%" />  
<col style="width: 16%" />  
<col style="width: 13%" />  
<col style="width: 12%" />  
<col style="width: 9%" />  
<col style="width: 12%" />  
<col style="width: 18%" />  
</colgroup>  
<thead>  
<tr class="header">  
<th><strong>Module ID</strong></th>  
<th><strong>Program ID</strong></th>  
<th><strong>Program Name</strong></th>  
<th><blockquote>  
<p><strong>Complexity<br />  
(Simple / Medium / Complex)</strong></p>  
</blockquote></th>  
<th><blockquote>  
<p><strong>Online Transaction Program</strong></p>  
</blockquote></th>  
<th><blockquote>  
<p><strong>Report Program</strong></p>  
</blockquote></th>  
<th><strong>Type of transactions</strong></th>  
<th><blockquote>  
<p><strong>Mobile Apps</strong></p>  
</blockquote></th>  
</tr>  
<tr class="odd">  
<th colspan="6"><strong>USER ACCOUNT PROGRAM</strong></th>  
<th></th>  
<th></th>  
</tr>  
<tr class="header">  
<th></th>  
<th></th>  
<th></th>  
<th></th>  
<th></th>  
<th></th>  
<th></th>  
<th></th>  
</tr>  
<tr class="odd">  
<th></th>  
<th></th>  
<th></th>  
<th></th>  
<th></th>  
<th></th>  
<th></th>  
<th></th>  
</tr>  
<tr class="header">  
<th></th>  
<th></th>  
<th></th>  
<th></th>  
<th></th>  
<th></th>  
<th></th>  
<th></th>  
</tr>  
<tr class="odd">  
<th></th>  
<th></th>  
<th></th>  
<th></th>  
<th></th>  
<th></th>  
<th></th>  
<th></th>  
<th></th>  
</tr>  
<tr class="header">  
<th></th>  
<th></th>  
<th></th>  
<th></th>  
<th></th>  
<th></th>  
<th></th>  
<th></th>  
<th></th>  
</tr>  
<tr class="odd">  
<th></th>  
<th></th>  
<th></th>  
<th></th>  
<th></th>  
<th></th>  
<th></th>  
<th></th>  
<th></th>  
</tr>  
</tbody>  
</table>  
  
#  
  
#  
  
# 3. Critical Batch Cycle Timing  
  
The below are the list of batch programs. The cycle timing of the batch  
is identified in the tables below.  
<br/>  
They are different from modules in the last section whereas they will  
run a thousand times everyday but the modules in this section probably  
run once or twice daily. Moreover, most of the modules described in this  
section are scheduled jobs running in backend, normally, the fluctuation  
of performance in these items would not affect the end user. Except, the  
generate report module, has a higher impact, thus it is the only item  
required to optimize.  
  
| \*\*Program ID\*\* | \*\*Program Name\*\* | \*\*Online Web\*\* | \*\*Mobile\*\* | \*\*Update Batch\*\* | \*\*Require Optimization?\*\* | \*\*Cycle Timing\*\* |  
|----------------|----------------|---------|---------|---------|---------|--------|  
| \*\*BATCH PROGRAM\*\* | | | | | | |  
| | | | | | | |  
| | | | | | | |  
| | | | | | | |  
| | | | | | | |  
| | | | | | | |  
| | | | | | | |  
| | | | | | | |  
  
# 4. Optimization changes  
  
The optimization will \*\*<u>only</u>\*\* focus the performance on programs  
and reports. Based on the database schema analysis, key areas for optimization include indexing, query efficiency, and data management for large collections.  
  
## 4.1 Optimization Actions  
  
### 4.1.1 Database Indexing  
  
Appropriate indexing is crucial for improving query performance in MongoDB. Based on the field analysis from the database schema, consider the following indexing strategies:  
  
- \*\*Identify frequently queried fields:\*\* Analyze application usage patterns to determine which fields are most frequently used in queries, especially in the 'sysfilerefs', 'adrblkfilerefs', 'tasks', and 'cases' collections, given their size and potential impact.  
- \*\*Index fields used in filters and sorts:\*\* For collections like 'tasks' and 'cases', fields such as 'status', 'application', 'submissionCase', 'taskType', 'createdAt', 'assignedBS', 'assignedGR', 'category', 'ReceivedDate', 'TargetReplyDate' appear to be good candidates for indexing as they are likely used in filtering and sorting operations.  
- \*\*Compound Indexes:\*\* For queries that filter on multiple fields, create compound indexes to improve efficiency. For example, if queries frequently filter on 'sysFileRefId' and 'createdDt' in 'sysfilerefs', a compound index on these two fields would be beneficial.  
- \*\*Index Cardinality:\*\* Ensure indexes are created on fields with high cardinality for better index selectivity.  
  
\*\*Example Indexing Considerations:\*\*  
  
- \*\*`tasks` collection:\*\* Index fields like `status`, `application`, `submissionCase`, `taskType`, `user`, `createdAt`. Consider compound indexes for common query combinations.  
- \*\*`cases` collection:\*\* Index fields like `Category`, `ReceivedDate`, `TargetReplyDate`, `assignedBS`, `assignedGR`, `application`, `team`, `createdAt`. Compound indexes for common case search criteria.  
- \*\*`sysfilerefs` and `adrblkfilerefs` collections:\*\* Given their large size, ensure indexes are in place for fields used in file retrieval and filtering, such as `sysFileRefId`, `adrBlkFileRefId`, `createdDt`, and potentially `createdName`, `createdSection` if these are used in search or filtering.  
  
### 4.1.2 Query Optimization  
  
Inefficient queries can significantly impact performance. Review and optimize queries, especially those targeting large collections:  
  
- \*\*Analyze slow queries:\*\* Use database profiling tools to identify slow-running queries.  
- \*\*Optimize query structure:\*\* Ensure queries are using indexes effectively. Avoid operations that prevent index usage, such as `$where` clauses or functions in query predicates when possible.  
- \*\*Projection:\*\* Retrieve only the necessary fields in queries using projection to reduce data transfer and memory usage.  
- \*\*Limit and Pagination:\*\* Implement limit and pagination for queries that return large datasets to improve response time and reduce server load, especially for online reports and data grids.  
- \*\*Aggregation Pipeline Optimization:\*\* If aggregation pipelines are used, optimize stages to filter and reduce data early in the pipeline. Use indexes to support aggregation operations where possible.  
  
### 4.1.3 Data Archiving  
  
For collections like 'sysfilerefs' and 'adrblkfilerefs', which are very large, consider implementing a data archiving strategy:  
  
- \*\*Identify archive criteria:\*\* Define criteria for archiving data based on age, status, or other relevant factors. For example, archive file references older than a certain period or associated with closed cases.  
- \*\*Implement archiving process:\*\* Regularly move historical data to archive collections or a separate archive database. This will reduce the size of the active collections, improving query performance and reducing storage requirements.  
- \*\*Ensure data accessibility:\*\* Design the archive process to ensure that archived data remains accessible for compliance and historical purposes, while minimizing the performance impact on the active system.  
  
| \*\*Collection Name\*\* | \*\*Optimization Suggestion\*\* | \*\*Details\*\* |  
|---|---|---|  
| sysfilerefs | Indexing, Data Archiving | Focus on indexing fields used in file retrieval and filtering. Implement data archiving for older file references to reduce active dataset size. |  
| adrblkfilerefs | Indexing, Data Archiving | Similar to sysfilerefs, prioritize indexing and data archiving strategies. |  
| tasks | Indexing, Query Optimization | Ensure indexes on key fields like status, application, user. Optimize queries for task retrieval and management. |  
| cases | Indexing, Query Optimization | Index fields relevant to case search and filtering. Optimize queries for case retrieval and reporting. |  
| notifications | Indexing, Query Optimization | Index fields used for notification retrieval and filtering, such as user and notificationType. |  
| applications | Indexing, Query Optimization | Index fields used for application search and filtering. Optimize queries for application management. |  
  
<span class="mark">\<\< End of Document\>\></span>