# Performance Optimization Report

**For Combined System Development Services for Licensing Self-Certification Portal of Buildings Department**

**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.

## Distribution

| Copy No. | Holder |
| --- | --- |
| 1 | Buildings Department (BD) |
| 2 | Master Concept (Hong Kong) Limited (MC) |

## 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](#introduction)
   * [1.1 Goal of Performance Optimization](#goal-of-performance-optimization)
     + [1.1.1 Server Loading](#server-loading)
     + [1.1.2 Bandwidth Usage](#bandwidth-usage)
     + [1.1.3 Better User Experience](#better-user-experience)
   * [1.2 Performance Optimisation Actions](#performance-optimisation-actions)
   * [1.3 Storage Allocation](#storage-allocation)
   * [1.4 Required Response Time](#required-response-time)
     + [1.4.1 Online Transaction](#online-transaction)
     + [1.4.2 Online Reports](#online-reports)
2. [Critical Online Transition Timing](#critical-online-transition-timing)
3. [Critical Batch Cycle Timing](#critical-batch-cycle-timing)
4. [Optimization Changes](#optimization-changes)
   * [4.1 Optimization Actions](#optimization-actions)
     + [4.1.1 Create Stored Procedures](#create-stored-procedures)
     + [4.1.2 Create Clustered Indexes](#create-clustered-indexes)

## 1. Introduction

The performance optimization of the system can be classified into optimization of Online Transactions.

## 1.1 Goal of Performance Optimization

The main goal of performance optimization is to improve the response time of the system for 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.
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.
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 other 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 badly 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 server loading and bandwidth resources more efficiently. The following 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 resources that require 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 across dual database systems:

i. System structured data ? Store in the "MS SQL Database" server in the Integrated system.

ii. Document-oriented data and file references ? Store in the "MongoDB Database" server in the Integrated system.

All the required system and textual data will be logically stored in various filegroups of Microsoft SQL Server database and collections within MongoDB. The following tables show the logic data storage in both database systems.

### 1.3.1 SQL Server Database (bd\_scs)

The following table shows the key tables and their estimated sizes for the MS SQL Server database:

| FileGroup | TableName | TableSize (Estimated) | Primary Purpose |
| --- | --- | --- | --- |
| PRIMARY | SchoolApp\_Infos | 0.5 GB | Stores school application information and metadata |
| PRIMARY | SchoolApp\_Submissions | 0.3 GB | Tracks submission records for school applications |
| PRIMARY | ApplicationCases | 0.2 GB | Manages application case records |
| PRIMARY | ApplicationFiles | 1.0 GB | Stores file metadata for application attachments |
| PRIMARY | AdrBlk | 0.7 GB | Stores address block information |
| PRIMARY | AdrBlk\_T | 0.5 GB | Contains temporary address block data |
| PRIMARY | ApRse | 0.1 GB | Registered structural engineer information |
| PRIMARY | Attachment | 0.4 GB | General attachments for all application types |
| INDEXES | Indexes for primary tables | 0.8 GB | Optimized search capabilities |

The filegroup growth size has been set to meet the recommendation for below 256 MB for data files.

### 1.3.2 MongoDB Database (bd)

Based on the provided database\_schema.md, the MongoDB database bd has the following characteristics:

**Database Statistics:**

* Database Size: 88.10 MB
* Collections: 12
* Total Documents: 1278983
* Total Data Size: 371.24 MB

**Collections Overview:**

| Collection Name | Document Count | Size (MB) |
| --- | --- | --- |
| tasks | 5523 | 0.99 |
| eminutes | 133 | 0.03 |
| submissions | 0 | 0.00 |
| applications | 381 | 0.36 |
| notifications | 1837 | 0.24 |
| bsblocks | 98397 | 6.40 |
| cases | 451 | 1.17 |
| oauthtokens | 3019 | 2.29 |
| sysfilerefs | 601808 | 204.70 |
| attachments | 370 | 0.13 |
| users | 116 | 0.04 |
| adrblkfilerefs | 566948 | 154.89 |

**Observations and Potential Optimizations (MongoDB):**

* **sysfilerefs and adrblkfilerefs dominate the database size.** Focusing on optimizing queries and indexing on these collections will likely yield the greatest performance improvements. Consider the common query patterns and add indexes accordingly.
* **submissions collection is empty.** If this collection is not being used, consider removing it to reduce database overhead.
* **Mixed data types in fields:** Several fields have mixed data types (e.g., tasks.user, eminutes.from, eminutes.to). This can hinder indexing and query performance. Standardizing data types would be beneficial.
* **Consider Indexing:** Based on the field analysis in database\_schema.md, identify frequently queried fields in each collection and create appropriate indexes. For example, indexing application and submissionCase in the tasks collection might improve query performance. Similarly, indexing adrBlkId and sysFileRefId in adrblkfilerefs would be beneficial.

## 1.4 Required Response Time

The required response time is defined according to 2 pre-defined categories: Online Transaction and Online Report. All of the programs and reports that require interaction and provide immediate response to users are categorized. The categorized programs and reports should respond to the user within the 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 cannot be optimized to have an immediate response, some parts of the program that take a lot of processing time will be swapped to batch jobs.

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 the 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 transactions:

* User Account Program
* Form and Record Management Program

Online transactions can be classified into the following groups:

| 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:

* (To be defined - 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 |
| --- | --- |
| (To be defined) | (To be defined) |
|  |  |

## 2. Critical Online Transition Timing

The following matrix is the list of programs with the complexity and transaction type being marked.

Offline modules help 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.

| **Module ID** | **Program ID** | **Program Name** | **Complexity (Simple / Medium / Complex)** | **Online Transaction Program** | **Report Program** | **Type of transactions** | **Mobile Apps** |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
| **USER ACCOUNT PROGRAM** |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

## 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. They are different from modules in the last section, whereas they will run a thousand times every day, 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 the 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 **only** focus the performance on programs and reports.

## 4.1 Optimization Actions

### 4.1.1 Create Stored Procedures

All reports are created by stored procedures. Stored procedures are precompiled as opposed to the dynamic prepared statements that are compiled whenever your application code invokes a call. Once you execute a stored procedure, it remains in the cache, saving the execution time. In addition, the stored procedures are mostly using primary keys for searching.

### 4.1.2 Create Clustered Indexes

When creating a primary key constraint, a unique clustered index on the column or columns is automatically created if a clustered index on the table does not already exist and you do not specify a unique non-clustered index. The primary key column cannot allow NULL values. In addition, when creating a unique constraint, a unique non-clustered index is created to enforce a unique constraint by default. When designing a clustered index, we have considered that the data types to be used as clustering keys. For instance, the primary keys are BIGINT data types, which are the best choices as clustered index keys.

| **Table ID** | **Table Name** | **LSCP Entity** | **Key Nature** | **Index Field** |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

<< End of Document >>