**Memory Management**:

In the gaming applications of memory, we want to look at the allocation. Specifically, blocking a certain amount of memory for certain usage. When there are allocated memory blocks, if certain data is no longer needed, they will be recycled for salvage. Other applications memory management usually must deal with few things such as CPU Overhead- extra time the memory management takes when the program is running. Pause Times- ability to have a rapid respond while using it such as computer connection (internet), Memory Overhead- determining how much space is wasted for administration, external fragmentation, and internal fragmentation. We should look at avoiding fragmentation and ensure that all allocation is moved in a proper manner and while minimizing gaps. Increase cache management while promoting balancing speed and space. Client should be monitoring the usage and profile on a regular basis to catch early issues.   
  
**Store Management**:  
Clients goals is to expand their application across all OPs (or as far as expanding to enough to where it will reach more customers). Clients should cover the following blueprints to ensure that store management is manageable for their expectations. We need to identify what type of data will be stored based on size and growth rate, for example media files, logs, and/or user data. Early predictions of usage can not only help with accountable goals with financial structure but also ensures that when there is a high demand for the applications and customers area using it, you will not run out storage. Client should have a plan for dismissal and backup strategies to ensure data integrity and availability. Implementation of compression to save needed space and positive implementation for large volumes of text or media files. Again, the client should be tracking usage and setting up special alerts for certain thresholds to prevent unexpected shortages.

**Comparison**:   
Between storage and memory management, there are obvious difference not all differences serve negative aspects for the client. Memory (RAM) is the game’s live performance space. This consists of fast access, memory allowing for quick access and data execution, and following instructions. Data in memory is volatile when the game is closed, or power is off. The execution of the game has elements like character movements, environment contribution, and real-time calculations. Storage (HDD/SSD) is the game backstage support for memory, where the data is permanent, keeping saved games, assets, and other user settings when the customer decides to end the game for the day (for example). Client should understand that loading data from storage can be slower compared to penetrating it from memory. With the game application in full progression for other OPs, the education of resource management is very important for keeping data files, textures, audio, and/or video files are stored in storage; loading all data into memory when needed.   
In conclusion, our clients should be able to look at the importance of why storage and memory are different. What purpose does each serve with the continuance of the game application, if the object of the goal is to expand to other operating platforms? Each functionality of storage and memory serves purposefully in the expansion.

Citations and Attributions

* [1. Overview — Memory Management Reference 4.0 documentation](https://www.memorymanagement.org/mmref/begin.html#mmref-overview-app)
* [CHAPTER 12 - Mass -Storage Structure | Operating System Concepts, 8th Edition (oreilly.com)](https://learning.oreilly.com/library/view/operating-system-concepts/9780470128725/silb_9780470128725_oeb_c12_r1.html)