Michael Ross

CS-230 – Assignment 6-1

2/13/22

Memory Management

# Memory Management

In examining the memory requirements for the application we must know the total memory available in the mobile platforms we’re designing this application for. On a mobile device, we would typically have 4GB or more available if we’re supporting the most current mobile phones. The total size of the images for the game takes up (200 images x 8MB per image) 1600 MB of space. On a mobile phone this amount of images would take up 1.6 GB or more of memory. This could be too much for the operating system to handle for a single application. A few ways we can mitigate this memory requirement are:

* Load only the images required for a single game (10-20) when a game is started. The server would send a specific set of randomized images to each of the clients involved in the game. This would reduce the memory required from 1600MB to 320MB at max.
  + Having the client pull the images from the server ever time would also allow swapping out the images available for games without having to update the mobile app. This will result in more frequent image swaps and more player engagement.
  + In between each game the images from the previous game can be purged and the new images loaded for the next game from the server.
* Store the files with the application itself and load the images (10-20) into memory when a game is started. This would still reduce memory needed for a game to the same amount (320MB at max).
  + Storing the files with the game would result in many app updates to update the image files. This would be inconvenient to the players.

On the server side, storing a single set of the available images in the GameManager singleton and allocating those images out to the clients from there ensures that we don’t have to store a 1.6GB copy of all images per game. Using this method will reduce cost on the server side by reducing the memory needed for the servers. The database will also likely need to utilize in-memory storage due to the performance requirement of managing potentially hundreds of games, logins, and game activities and storing and retrieving that data from the database. Leveraging a PaaS service for these databases can ensure that the amount of memory needed for performant operation will always be available.

# Storage Management

In examining the storage requirements for Draw It or Lose It, we can see that the following components will require storage:

* Mobile:
  + Application files
  + Temporary storage for images.
* Server:
  + Application files
  + Database for User information, game information, and metrics.
  + Storage for Image files.

The application files storage needs will be well known. The compiled size of the application along with any supplementary application components. The temporary storage for images can be sized at the total needed per game. For example if 10 files are required per game we’ll need 80MB for temporary image storage. The database is the biggest variable component for storage. The more players that are onboarded, the more storage will be needed. Leveraging a PaaS service like AWS Dynamo DB or Azure Cosmos DB ensures these databases can grow as large as possible. Lastly, the storage needed for the game images can be easily calculated by the number of images times the size of each image.

To compare the two simply, memory will be used for performance intensive tasks while disk storage will be used for raw image storage. We will store the images actively being used in a game in memory to ensure the speed necessary for that task. The images not being used will be stored on disk waiting to be identified in a game. This is also important for cost management for the game. Memory is much more expensive than disk storage. Ensuring only the minimum amount of data is stored in memory based on storage requirements keeps app cost lower.