Memory management is one of the most important aspects of an application that a developer needs to consider. In the context of developing Draw It or Lose It, the game must run on multiple operating platforms and be able to choose and render one of 200 high-definition photos, each about eight megabytes in size. The drawings will be rendered to the screen at a fixed rate over the course of thirty seconds. To keep up performance, the game should make use of caching to “remember” the items that will be used frequently (i.e. logos, icons, banners). Performance is increased if the game does not have to re-render everything to the screen from scratch if part of what it is loading is cached. The game’s appearance will not change between rounds, so we can also make use of Client Side Rendering (or CSR for short). CSR has JavaScript do the rendering in the browser, meaning the logic would be done on the webpage itself (Miller and Osmani). This is perfect for the game as the static things can be cached and the drawing can be rendered in real time for the client with some efficient JavaScript. It would likely have to parse through the image and render each line of pixels at a fixed rate during the thirty seconds, which should not be heavy on the browser whatsoever.

The next question to be answered is how the images will be stored and retrieved. With Draw It or Lose being a web-app across multiple platforms and a mobile app on IOS and Android, the answers differ. The images could be stored in a database like MySQL or MongoDB and be retrieved at random when the round begins, but that requires additional dependencies and knowledge of these tools, not to mention that storing images in a database is slow, especially at higher volumes. Rather, the ideal solution would be to determine the storage size needed for the image library. The logos, icons, colors, and text are all embedded in the UI, and have no reason to be stored elsewhere. Knowing that each picture is approximately eight megabytes in size at 200 images total, we need at least 1600 megabytes of space, or 1.6 gigabytes. For the web-application, it is easy enough to store the images in the file server of the site, as it makes them easy to retrieve. And, because only one image is rendered per game instance the game’s performance should not take a hit. The mobile app does not have a web-based file server, so the images would need to be included in the app. To prevent the app from being 1.6+ gigabytes, the images would need to be compressed in some form before they are packaged into the application for download. Each image could be compressed individually, or they could all be shrunken and placed in one file, similar to a sprite sheet. Cloud storage could also be considered as it would make the process of image retrieval uniform across platforms, but it would also add to The Gaming Room’s expenses. Google Cloud’s standard storage cost is $.026 per gigabyte per month (Perry). The game would need at least two gigabytes per month, possibly more if the image gallery were expanded, and this would cost The Gaming Room about $.624 a year as there is no cost for data retrieval. Sixty cents a year isn’t bank breaking, but it would likely be better to save the money and go with the aforementioned storage choices.

Memory and storage, while equally important to the game, are also equally different from each other. Memory is where the processes are currently being executed, and storage is where the data and components are held. Translated to the game, memory is where the actions of the game are being sequenced. Every pixel rendered or player’s guess goes into the memory. The storage is where the game’s data is held, like the image library or the data from a player’s profile. The cache kind of resides in-between storage and memory but is generally considered fast memory. It holds the pieces that the device believes will be needed next. As stated earlier, the cache will hold the pieces of the game that are used and displayed frequently, like the logos, banners, and icons. As the developing team, we must know how to differentiate between and effectively manage the two to ensure the user will have a positive experience when playing Draw It or Lose It.

Works Cited

Miller, Jason, and Addy Osmani. “Rendering on the Web.” *Web.dev*, 6 Feb. 2019, https://web.dev/rendering-on-the-web/.

Perry, Yifat. “Understanding Google Cloud Storage Costs.” *NetApp BlueXP*, 4 Oct. 2021, https://bluexp.netapp.com/blog/cvo-blg-understanding-google-cloud-storage-costs.