Photo sharing app on Android

1. This allows us to share all the photos here

Functional Requirements:

* Core features include image uploading, downloading, viewing, and basic editing.
* User account management and image categorization.
* Search functionality for images.
* Image sharing and social interactions, if required.

Non-Functional Requirements:

* High performance and scalability to handle a large user base and image data.
* Compatibility across various devices and OS versions.
* Considerations for data usage and storage optimization.

What are the key components?

A diagram of a computer network

Description automatically generated

* **Photo Downloader/Uploader** — the central component that provides the client API and brings all components together.
* **Download/Upload Request** — encapsulates a single photo downloading request; accepted by file downloader as input.
* **Download/Upload Task** — represents an asynchronous download/upload operation; produced by the photo downloader as output.
* **Download/Upload Dispatcher** — schedules and dispatches download/upload operations.
* **Network Client** — handles receiving bytes over HTTP.
* **Photo Store** — writes Photo contents to the disk. (if needed for showing offline view, please clarify with the interviewer once)

How is lazy loading image done in Android?

1. This can be achieved with lazy columns as said before, direct component of lazy coluns in Compose

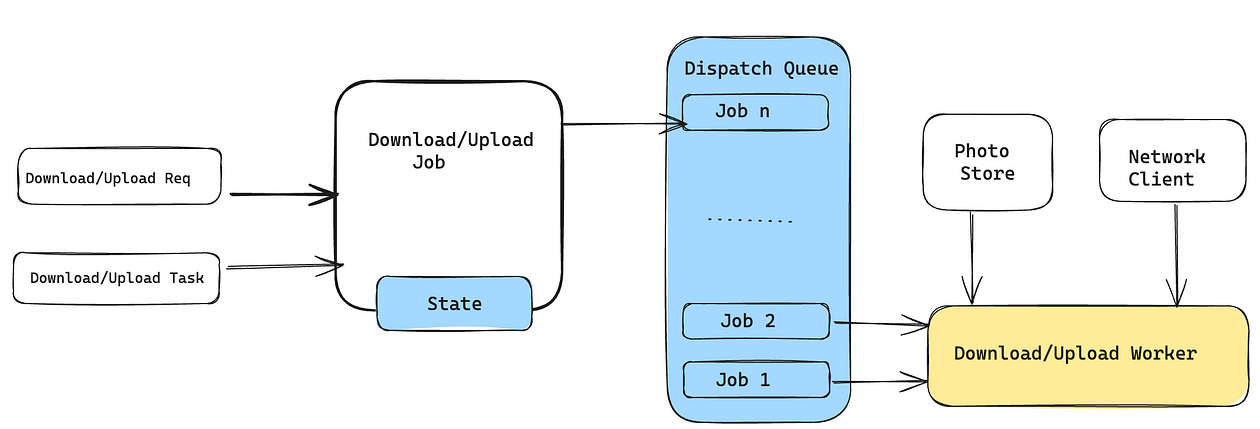
How can image loading be efficently done?

1. This can be done with Glide which caches the image

Offline work

1. Done with WorkManager, when network comes back online, we will reach out here

How does the download/dispatcher look?



* Download/Upload Dispatcher maintains a queue of jobs. Each job consists of a download request, a download task, and a state (PENDING, ACTIVE, PAUSED, COMPLETED, FAILED). The active jobs are dispatched by download workers.
* Job responsibility is to keep track of pending, active, and completed jobs. we also need to maintain the pause state of the scheduled jobs in case of a network failure scenario. Job encapsulates a single downloading request from the user.
* Download Worker is responsible for actual data transmission from the network and handles blocking I/O.