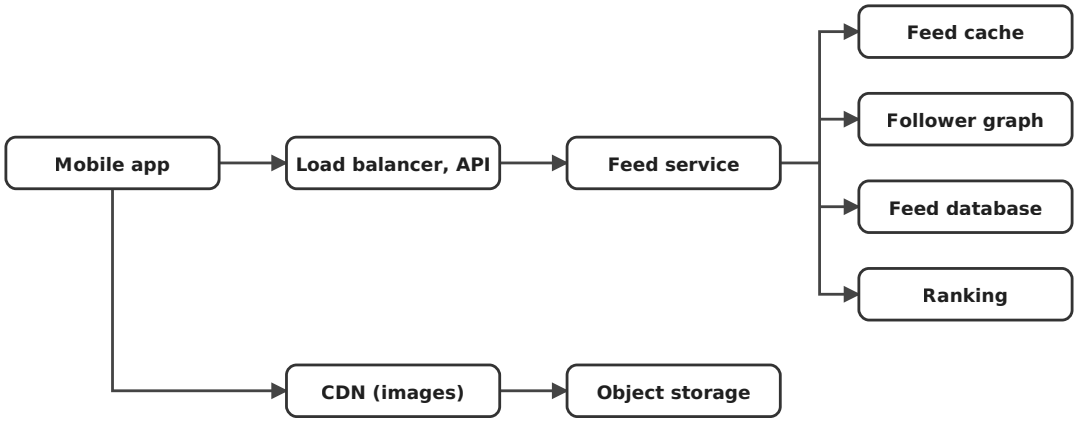
# Photo Feed · Chaos Experiment Card

**Andrew IDs:**

---

Your group runs reliability for a photo-sharing app. During the lecture you will design one chaos experiment for its most-used journey, **loading the feed**: a user opens the app and sees recent photos from the accounts they follow. Fill in one part at the end of each section.

## The system




- The app makes about **100 million** feed loads every 28 days.
- **Last 28 days, at the load balancer:** 0.02% of feed loads failed; 95% finished within 500 ms, 99.5% within 1 s, and 99.95% within 2 s. Ranking takes about 100 ms of each load.
- The feed cache answers about **95%** of feed loads; misses go to the follower graph and feed database.
- Photos load separately: once the feed arrives, the app fetches each image from the CDN.

## How the feed service answers `GET /feed`

```
def load_feed(user):
    post_ids = feed_cache.get(user)
    if post_ids is None:                                  # cache miss
        follows = follower_graph.following(user)
        post_ids = feed_db.recent_posts(follows, limit=500)
        feed_cache.put(user, post_ids, expires_in="10 min")
    ranked = ranking.rank(user, post_ids)                 # most interesting first
    return feed_db.post_details(ranked[:20])              # captions and CDN image URLs
```

Every call uses its client library’s default settings.

## Key terms

- **SLI:** the proportion of valid events that were good, measured at a stated point.
- **SLO:** the target for an SLI over a window, such as 99% over 28 days.
- **Error budget:** the share of events the SLO allows to be bad (100% − SLO).
- **Hypothesis:** a prediction, in SLI terms, that the steady state survives a specific failure.

## 1 · Steady state: how do we know the feed is working?

How would you know the feed is working for users? Propose two SLIs, then choose one.

**SLI A**

**SLI B**

**Choice and SLO.** Which SLI did you choose, and why? Set a 28-day SLO and work out its error budget. Would last month have met it?

## 2 · Hypothesis: what could go wrong?

Brainstorm failures: imagine the feed broke last night, recall outages you know, and walk the request hop by hop through the diagram. Pick three. Note what users would see and whether your SLI notices, rate impact and likelihood **H**, **M**, or **L**, then circle your top hazard.

| Failure: what happens, and where? | What users see | SLI notices? | Impact | Likelihood |
|---|---|---|---|---|
| | | | | |
| | | | | |
| | | | | |

**Hypothesis.** When _____ for the experiment group, the SLI stays within _____ of the control group's.

## 3 · Run: how do we test it safely?

Use your top hazard. If you don’t have one, use: **ranking responds slowly.**

| Where to inject | Example tools |
|---|---|
| Code | Mocks, failpoints |
| Service calls | Filibuster, Netflix FIT |
| Network | Toxiproxy |
| Container or host | Pumba, Chaos Monkey |
| Platform orchestration | Chaos Mesh |

**Injection.** Where and how will you cause the failure, and under what load?

**Groups.** Who is in the experiment and control groups, and how many users?

**Abort condition.** When will you stop the experiment? Name the signals and their thresholds.

## 4 · Learn: your hypothesis was refuted

The SLI fell during the experiment.

**Defense.** Which pattern should contain your hazard? What will users see instead?

**Rerun.** What must the rerun show for you to accept the change?