REQUEST FOR COMMENTS

Please add this document to Photobox Weekly Tech Forum RFC shared drive.

<Title Goes Here>

This template is intended to encourage an engineer to think through and document a proposed design upfront. As well as encouraging rigour in design, it enables other engineers to give useful opinions and advice on design decisions (normally through the WTF), reducing the cost of later changes in direction. This is not intended to be a living document; only a snapshot of the design choices made. Final decisions should be documented as an ADR within the GITHUB repo related to the change.

Not all of these sections will relevant to all features - mark as N/A as appropriate

Problem & Context

What (technical and/or business) problem does this feature solve? Why are we doing this? Link to more information (JIRA tickets, specs) as needed

Considered, but out of Scope

It's often worth listing the features that have been considered but which are out of scope for this iteration of the design. This helps preempt questions during the discussion and show that the features have not accidentally been missed.

Solution

How are you intending to solve this problem? Start with an overview (diagrams are great here).

Architecture

- *What does this service or feature look like? Please draw diagrams where possible (Google drawings, Lucidchart, photos of whiteboards/pad paper…)*
- *Why does it look like that?*
- *Consider the service/feature boundaries; what business capability is this service responsible for?*
- *What data does it own?*

Dependencies & Integration

How does it interact with existing services/functionality/components, both upstream and downstream? Consider both existing and planned software and services.

Interfaces

What interfaces and operations might this expose? Don't need to be final, but good to have an idea.

Infrastructure

What pieces of infrastructure are you using/reusing? E.g. new AWS service, new open source libraries, new databases, new programming languages? Are you introducing new dependencies?

Scale & Performance

Have does feature scale?

- *What are the expected scale characteristics (at least back of an envelope) for this service? Consider CPU, memory and storage. Can it horizontally scale?*
- *Give a rough estimate of AWS (or other) costs as your feature/service grows*
- *How might your design decisions increase/reduce cost?*

Reliability

What level of reliability are you aiming for? If client-side software, what range of devices and OS versions are you aiming to test on and support? If a service, are you comfortable achieving 99.95% (or higher) availability? Is the software experimental or aiming to support thousands-millions of users straightaway?

Redundancy

How do you manage backups and restores, if needed? Can you handle the loss of connectivity? Are there alternative services and fallbacks? What are your hard RAM/disk limits, if any?

Monitoring & Instrumentation

How will you understand the behaviour of this feature/service in production - especially how it meets product goals / provides customer value? What data/events will we be sending to the data platform to support these metrics? Will there be any alerting or metrics that you will measure? How will you monitor the technical behaviour of your service/feature?

Failure Scenarios

When will this design/feature/approach fail? How will you mitigate the impact of these failures?

Security

What threats does your software face? If it uses cryptography, describe it. How secure is your data? If someone intercepts your comms, replays your messages, dos's your service, how do you cope with it?

If your service stores personal data, how is it protected? How is internal access to that data controlled?

Does your feature comply with the engineering security principles?

Privacy

What personal data does your software use? Does your software use biometric data, identity documents, or other high-risk data? Where is personal data stored and how is it protected?

Under what circumstances will personal data be deleted or moved to long term storage? How are individuals able to exercise control (access and deletion) over their personal data?

Operational Implications

Who will run this feature/service on a day-to-day basis? Does it impact DevOps/SRE? Does it require company operations support e.g. from finance, service delivery or data supplier operations?

Future Directions

What are you building that might be of use to future products and services? Have you decoupled potentially generic functionality from product-specific functionality? Are there future requirements you should take into account so that you can more easily accommodate them without a major redesign? If you made some trade-offs to develop the current version faster, knowing that some refactoring lies ahead, then highlight them here.

Rollout

How will you roll this service/feature out? Consider feature flags / canary / dark launch / communication / etc

Risks & Open Questions

What are the major risks that might prevent your software from working or being successful? What don't you know yet that might change this design or how you approach implementation?

Alternative Approaches

If you considered and rejected some alternative approaches, describe them. Someone reading this design might think one of these options was intuitively more obvious and it would help to explain why we are not following it.