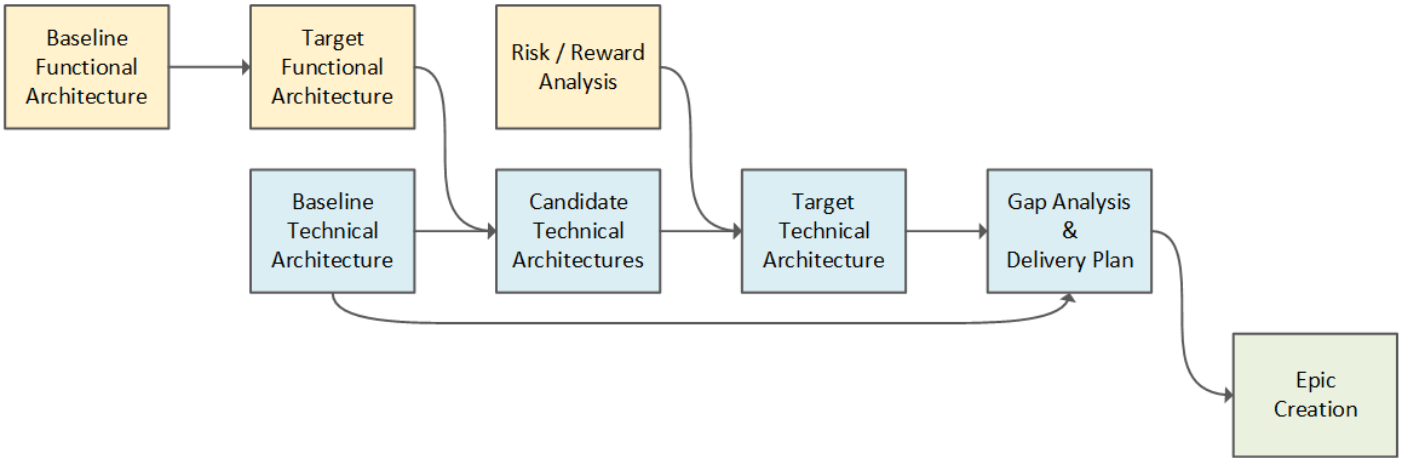
Questions for architecture

> A cheat-sheet to help drive effective outcomes and facilitate inter-departmental communication.



Business Questions

Please describe the high-level vision of the business proposal, the problems it solves and goals it supports.

- Do they understand the current system thoroughly, including its strengths, weaknesses, and dependencies?
- Do they understand the business's needs and goals?
- Can they provide **evidence** of stakeholder engagement and a clear understanding of their requirements?

What business benefits do you anticipate from your proposed architecture?

- How does the proposed architecture link to these benefits?

How do you propose to measure those benefits?

- What are the cost implications of this architecture?
  Infrastructure requirements, licensing costs, etc ...
- How will you measure end-user satisfaction?
  User feedback, adoption rates and other engagement metrics.

What are the risks and risk mitigation strategies to this architecture?

What changes should we make, at an organisational level, to support this project?

What are the priorities for delivery? Can this list form the basis of the Epics in an Agile environment?

How is this architecture future-proofed? How easy it is to adapt the architecture to changing business needs?

Application Questions

Can you describe the data and application architectures in your proposal?

Variants

- Is there more than one candidate solution architecture?
- What are the pros and cons?
- How does the cost and time analysis factor into these choices?
- What candidate architectures were rules out any why?

Look for:

- **Modular Design**
  Can each module can be updated, replaced and scaled individually without affecting the rest of the system?
  (*resilient and cost effective*)
  Do we provide wrappers over off-the-shelf products to make them replaceable if ever needed?
  (abstraction over underlying functionality)
  Is the application broken into small, loosely coupled services, which can be scaled individually.
  micro-services
- ***Auto-Scaling***
  Automatically adjust the consumed resources (and cost) based on current demand without manual intervention.
  (*elastic scaling*)
- **Reliability**
  How many 9s are we targeting?
  (this is a measure of uptime, 99.9% = three 9s, 99.999% = five 9s)
  Is disaster recovery considered?
  Can the system distribute traffic efficiently across multiple servers or instances?
  (load balancing)
- **Monitoring and Logging**
  This is essential for debugging in the cloud.

- **Identity and Access Management**
  How do you ensure that only authorized users can access relevant resource?
  Do we include multi-factor authentication?
  Do we include single sign-on?
  (IAM, MFA, SSO)
- **Data**
  Data should be encrypted both in transit and at rest.
  How will we ensure the architecture handles data consistency across distributed systems?
  (TLS, ACID Vs Eventually Consistent)
- ***Defence***
  Is there a cloud-based firewall to monitor and control traffic to the system?
  (WAF web application firewalls)
- ***API Access (and security)***
  Is there an application gateway? Should there be? These provide authentication, caching, rate limiting, versioning, analytics, and monitoring of API access to prevent abuse.
  (APIM)
- ***Other Systems***
  What other technologies / products are you proposing to use and why? What is the strategy for integration?

Delivery Practices

What is your plan for migrating from the current architecture to the proposed one?

- Is the proposition minimally disruptive? How so?
- Is the migration plan incremental in delivery?
  (strangulation pattern, piecemeal replacement, phrase-based on location or user sub-sets)

How engaged is the implementation team?

- Is the architecture using known technologies?
  If possible, choose technologies, languages, and frameworks the team is already familiar with to reduce the learning curve and to help them implement the architecture effectively.
  Will we provide training if new technologies are necessary?
- Is the architecture sympathetic to delivery capacity?
  If the delivery team is small, or lacks resources, the migration plan should focus on small discreet functionality that can be delivered according to their availability and should be designed to be easy to manage and scale without a lot of manual intervention.
- Is this collaborative?
  Is there a feedback loop to allow the implementation team to voice concerns, suggest improvements, and be part of the decision-making process?

How do you plan to govern the implementation of the architecture?

- Who will be involved?
- What are the decision-making processes?

What is the plan for user acceptance testing?

- How will changes to the architecture be managed once it is in place?
- Do we need to provide end-user training?

Revision #2
Created 21 September 2023 10:58:21 by James Hall
Updated 21 September 2023 21:36:41 by James Hall