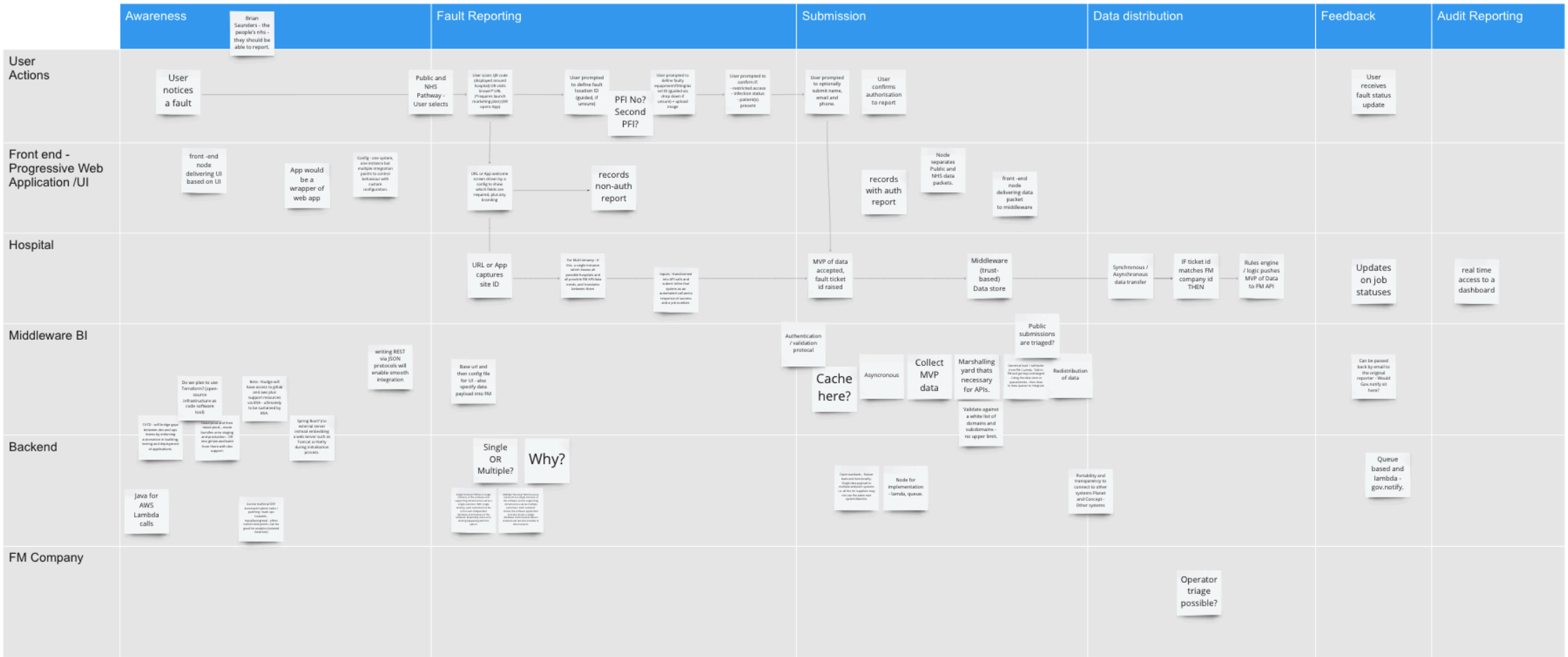
Awareness
Fault Reporting
Submission
Data distribution
Feedback
Audit Reporting
User Actions
Front end - Progressive Web Application /UI
Hospital
Middleware BI
Backend
FM Company
Brian Saunders - the people's nhs - they should be able to report.
User notices a fault
Public and NHS Pathway - User selects
User prompted to define fault location ID (guided, if unsure)
PFI No? Second PFI?
User prompted to confirm it: - restricted access - infection status - patient(s) present
User prompted to optionally submit name, email and phone.
User confirms authorisation to report
User receives fault status update
front -end node delivering UI based on UI
App would be a wrapper of web app
records non-auth report
records with auth report
Node separates Public and NHS data packets.
front -end node delivering data packet to middleware
URL or App captures site ID
MVP of data accepted, fault ticket id raised
Middleware (trust-based) Data store
Synchronous / Asynchronous data transfer
IF ticket id matches FM company id THEN
Rules engine / logic pushes MVP of Data to FM API
Updates on job statuses
real time access to a dashboard
writing REST via JSON protocols will enable smooth integration
Base url and then config file for UI - also specify data payload into FM
Authentication / validation protocal
Cache here?
Asyncronous
Collect MVP data
Marshalling yard thats necessary for APIs.
Validate against a white list of domains and subdomains - no upper limit.
Redistribution of data
Public submissions are triaged?
Can be passed back by email to the original reporter - Would Gov.notify.io here?
Single OR Multiple?
Why?
Node for implementation - lambda, queues.
Portability and transparency to connect to other systems Planet and Concept - Other systems
Queue based and lambda - gov.notify.
Java for AWS Lambda calls
Operator triage possible?