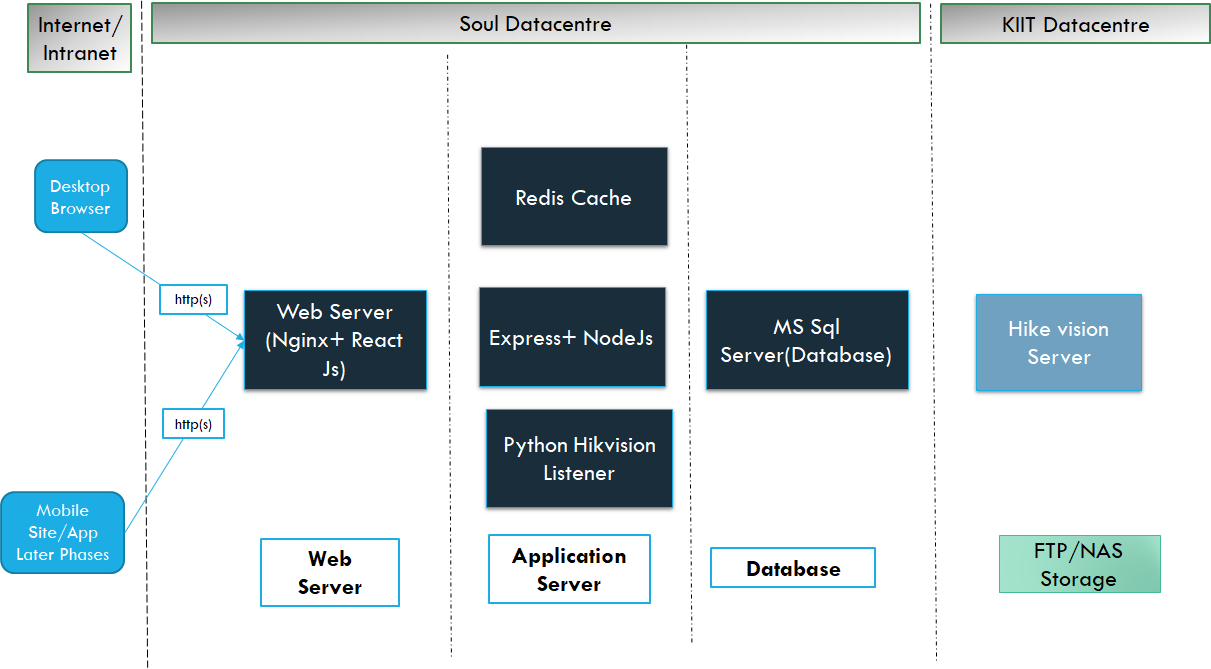
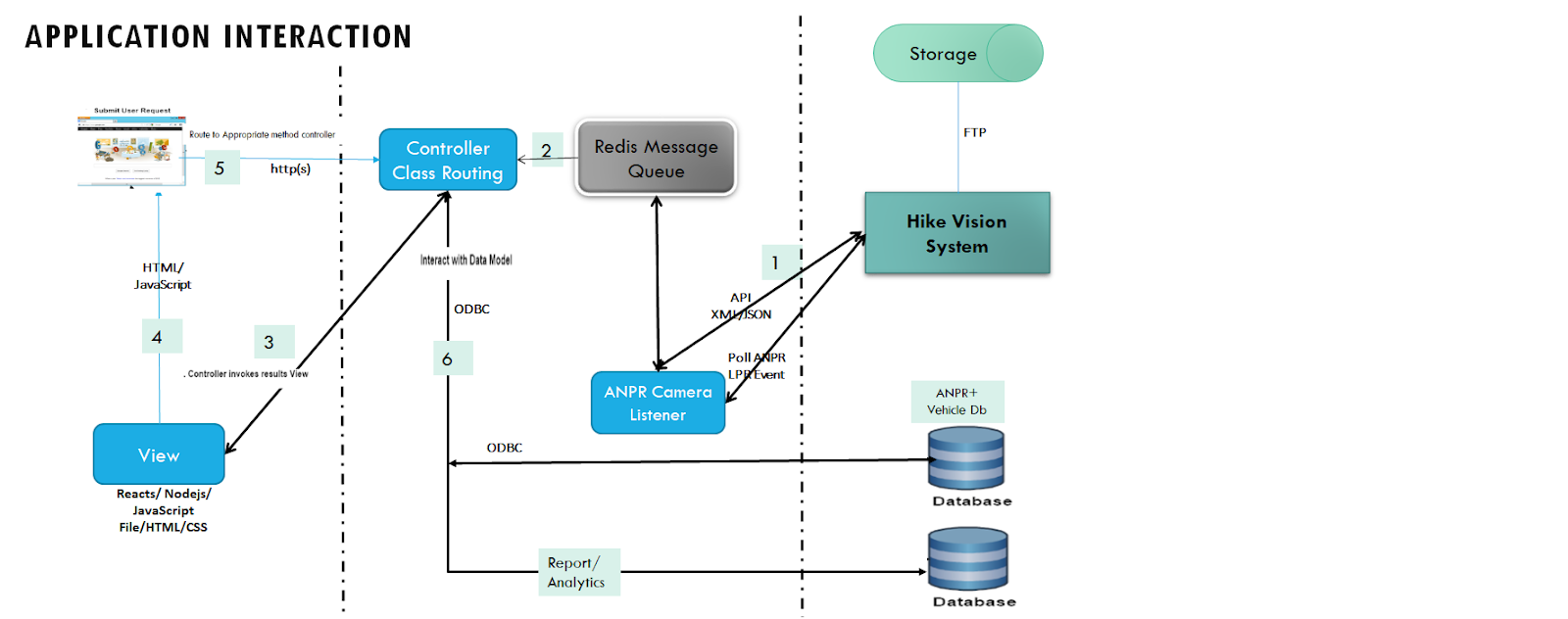
The solution is proposed to be developed as a 3 tier web application and facility for a mobile application in later phases along with support  for ~~lIve view~~ and advanced notification.

Following is the high level view of the proposed infrastructure setup across 3 tiers and spanning Soul/Kiit data center to host the application and relevant servers.



The design of the application will be composed of a modular code base based on 3 tier MVC architecture with responsive design to support Tab form factor for the security at the gate.



The data requirements for the application can be divided into 2 parts Master and transactional and analytics or denormalized data for reporting later. The below suggested schema will be used by the data processing layer to interact with ANPR database.

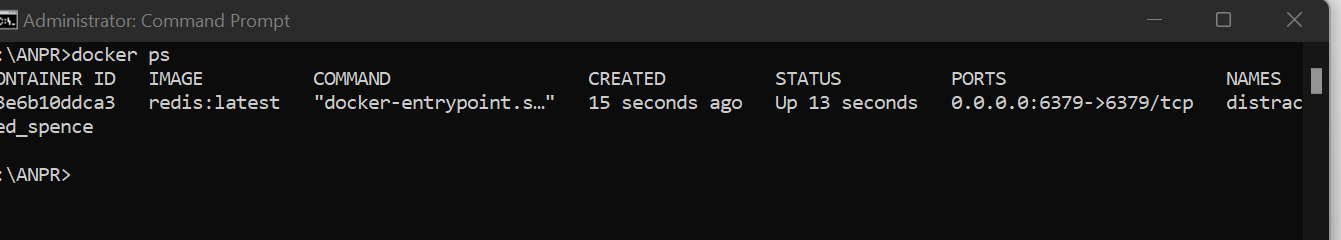
**Business Component (data flow and integration)**

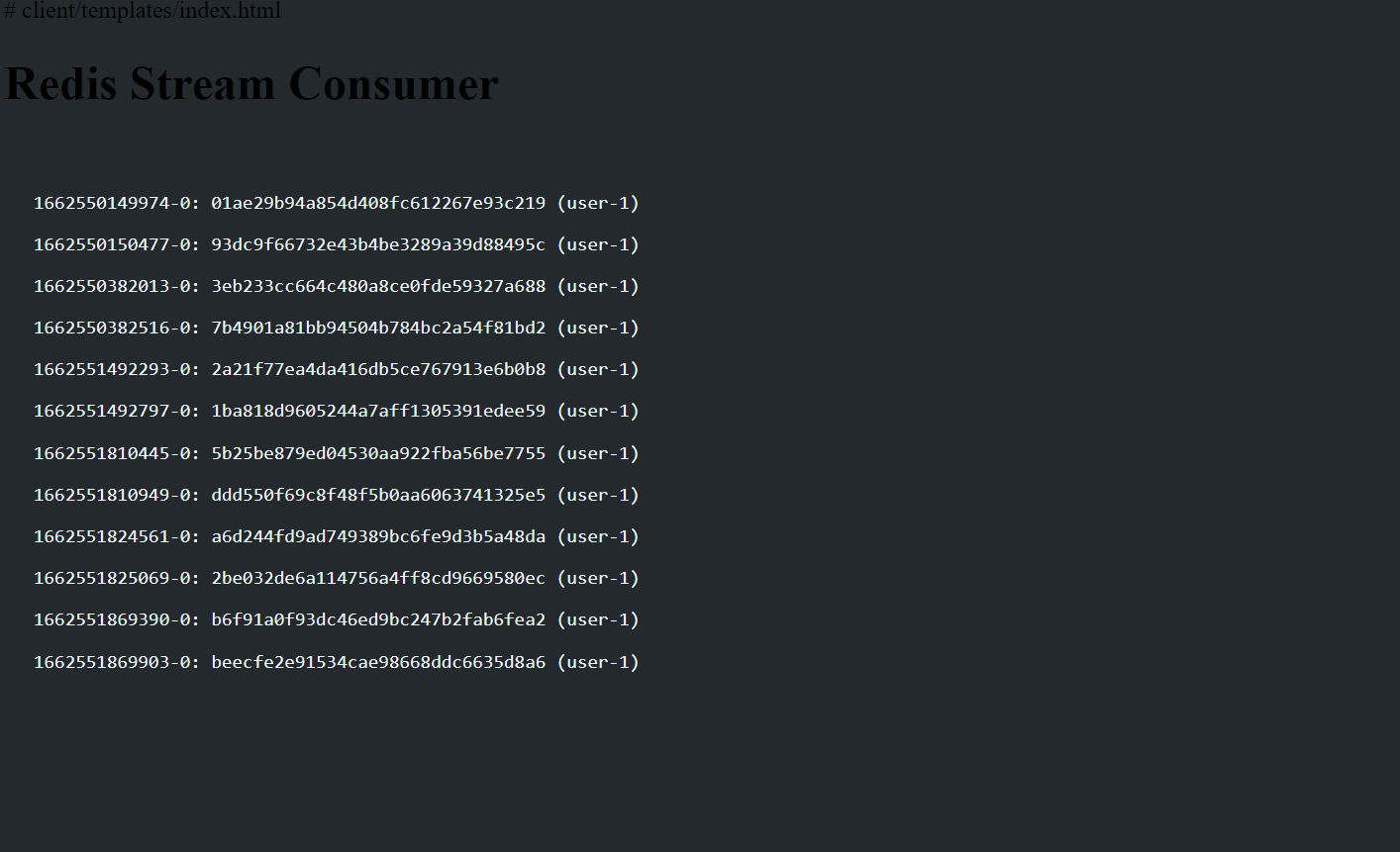
HikVision Python Listener (WIP)

Polling frequency : <to be set post discussion>

Python API to web server and web view integration

Redis on local (change for server) port 6379 mapped to host 6379





Message output on web server on a html page (can be also react js/angular or vuejs or similar JS framework)

**Publisher/subscriber setup**



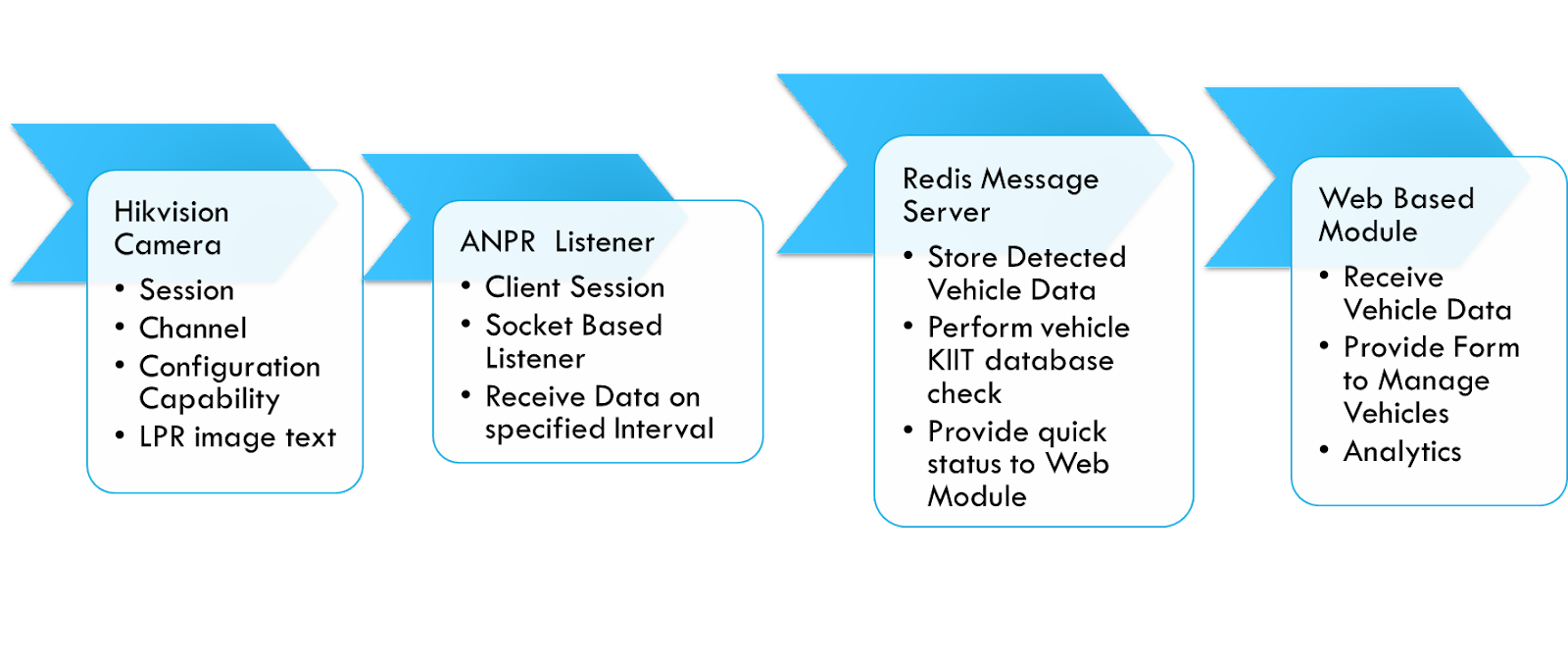
Tools: redis/nodejs socket io/web socket technology

* Create a redis setup on port 6379

* Create a redis client for python/nodejs

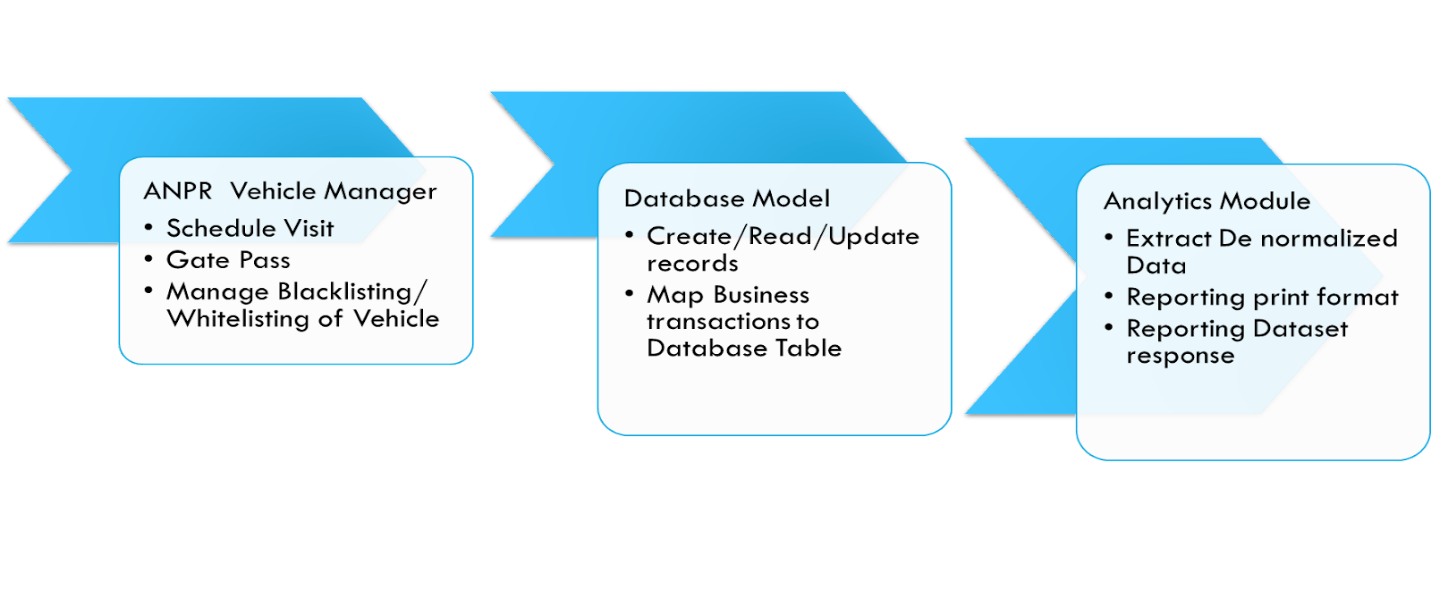
* Create a publisher model with client side browsers as subscribers.
* When any hikvision camera capture event alarm is detected  http post request is sent by a background redis cache job to node js/python module .

* Once all the data is received the react js/javascript/client side tab browser is refreshed with the vehicle details. The page is auto refreshed every 3 seconds.



To add: background/asynchronous worker process)

WIP (analytics/reporting module)



To expose the business or middleware we can expose the functionality as API so that it can be consumed by UI layer or react js.