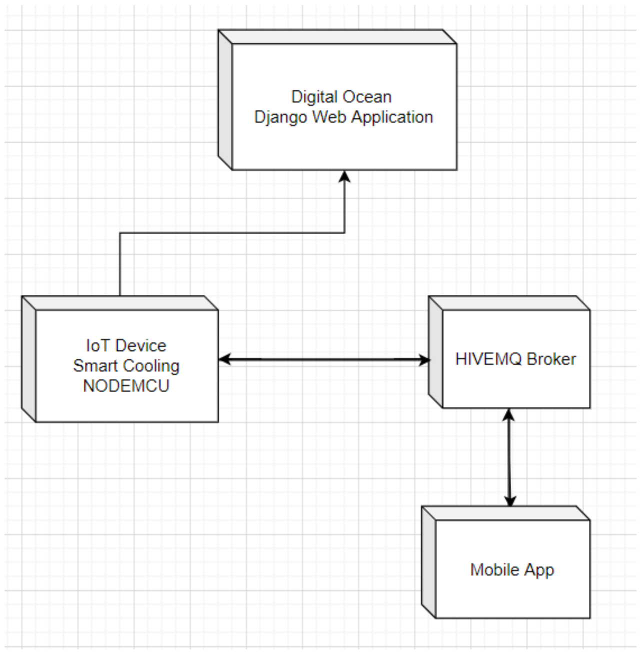
Deployment Design Decision

Current Single Smart Fan

Currently, we are only using one smart fan and our Deployment Architecture looks like the following:



This design choice is feasible when there is only one Smart Fan shown in the left. Deploying the system in this configuration ensures that performance is gained. The data is sent directly from the Smart Fan to the Django Web Application on Digital Ocean. However, this deployment architecture would not be feasible for when there are multiple Smart Fans available.

For Multiple Smart Fans

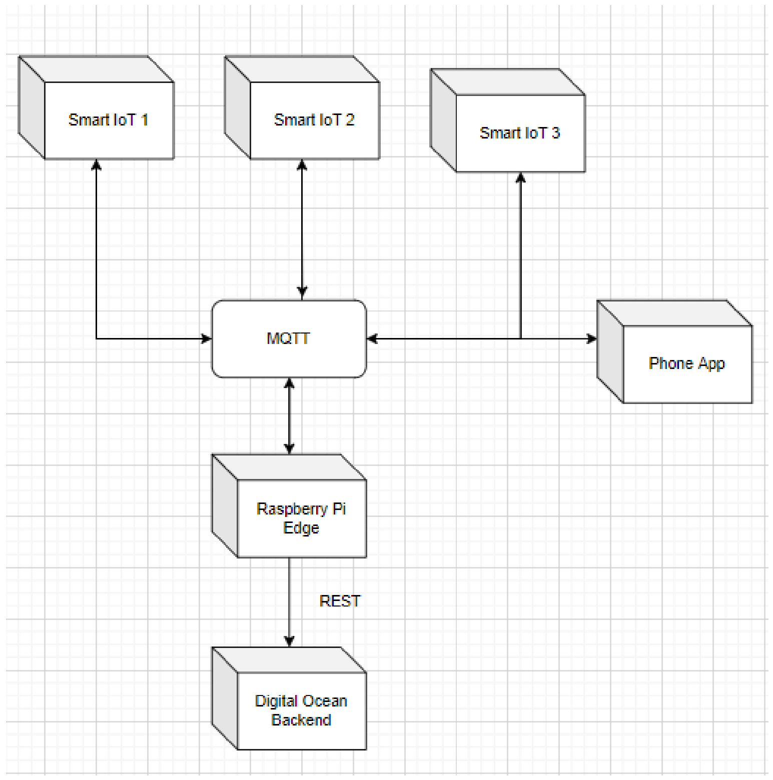
Edge Processing Decision

Edge detection would only be needed if we assume that the user needs many smart fans. In such a case, we would need edge processing. However, In our selection, since we were only working with one Smart IoT fan, we did not need edge processing or data filtration using an additional edge device such as a raspberry pi. However, if our requirements were different to allow users to have multiple Smart Fans available, the method and Architecture is shown below.

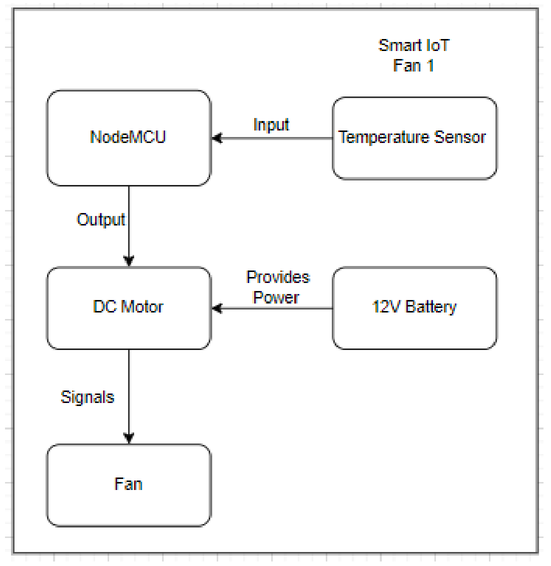
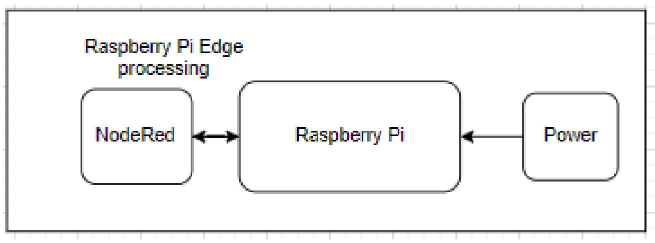
Method:

The idea of edge processing for our design would only be necessary if we have more than 1 Iot Smart Fans in the household. With multiple IoT fan units, it would be better to use a raspberry pi 3 for edge computation and data filtration. The idea is to change the architecture in a way such that the user is able to incorporate multiple IoT smart fan units in the house. The architectural design is shown below.

Deployment Architecture



Component Design	Design

Smart IoT Fan	 Smart IoT Fan 1 NodeMCU Input Temperature Sensor Output DC Motor Provides Power 12V Battery Signals Fan
Raspberry Pi for Edge processing	 Raspberry Pi Edge processing NodeRed Raspberry Pi Power

Data processing

At the edge, the raspberry pi would receive temperature data from all the smart fans which would allow the users to set up fans in different locations. The raspberry pi would maintain a queue or buffer for each historic data coming from the smart fans. Finally when the buffer reaches a certain limit, all the data in the queue would be sent to the backend for data storage. There the users can login in and view their data on the digital ocean. The node red would enable this workflow from collecting data from the each IoT device and using rest to communicate with the backend on digital ocean

Pros

- Using edge processing ensures high availability and reliability on the digital ocean.
- Post request would only be called to the digital ocean endpoint when the buffer is full
- Makes the system fault tolerant from losing connectional
- Allows for data recovery due to the queue
- Data security and integrity is maintained

Cons

- Adds cost to the overall system.
- Performance would be less since the data would have to propagate through the edge device.