Dwight Look College of ENGINEERING
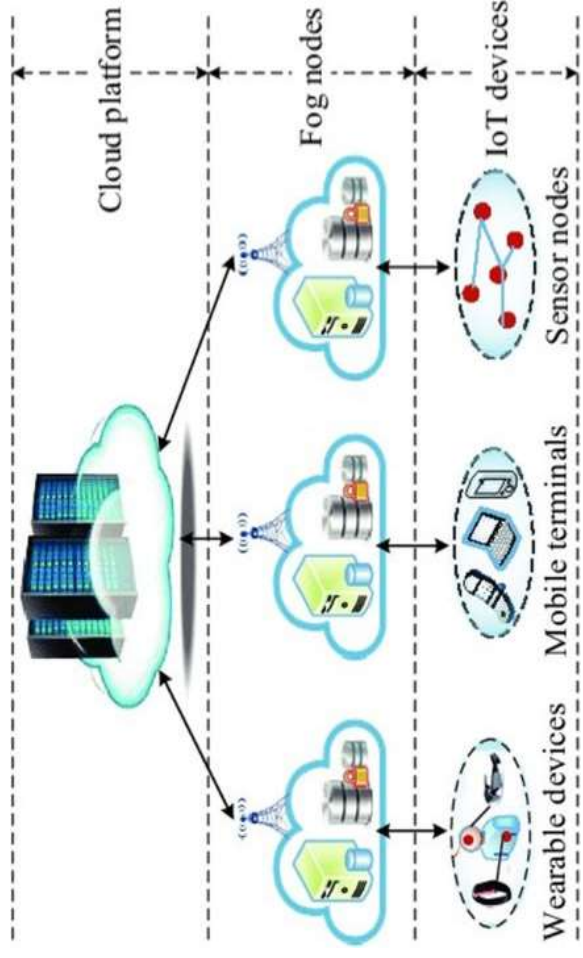TEXAS A&M UNIVERSITY

Team 21: Load Balancing on an IoT Fog
Bi-Weekly Update 1
Sebastian Correa, Christopher Gonzales
Sponsor: Swarnabha Roy

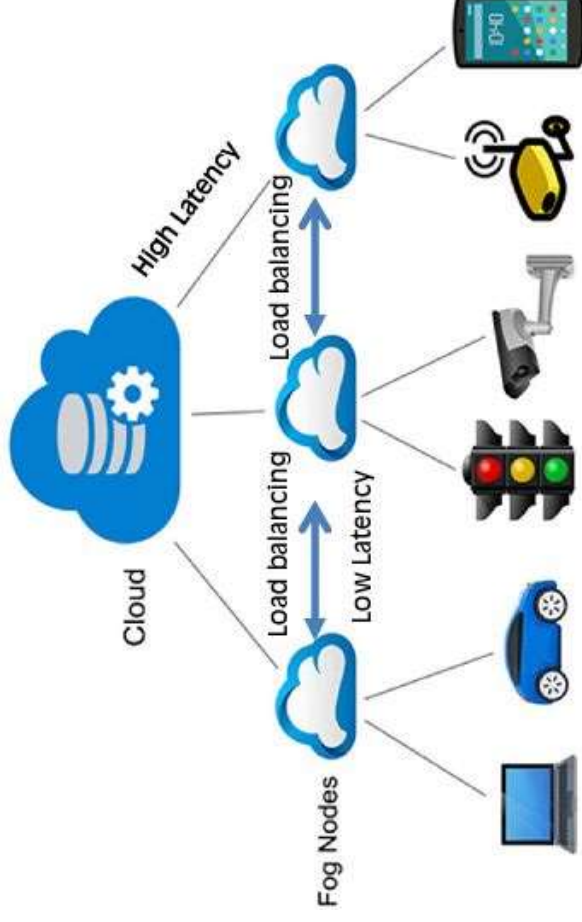# Project Summary

Problem statement:

- IoT devices are becoming more common
- Fog Computing
- System must handle the total workload
-

Project Summary

Load-Balancing Solution:
- Takes input from IoT devices
- Distributes the workload
- Sends data to the cloud

Subsystems Diagram

Cloud Layer

Sebastian Correa

Fog Layer

Christopher Gonzales

IoT Devices

Users

Load Balancer on an IoT Fog
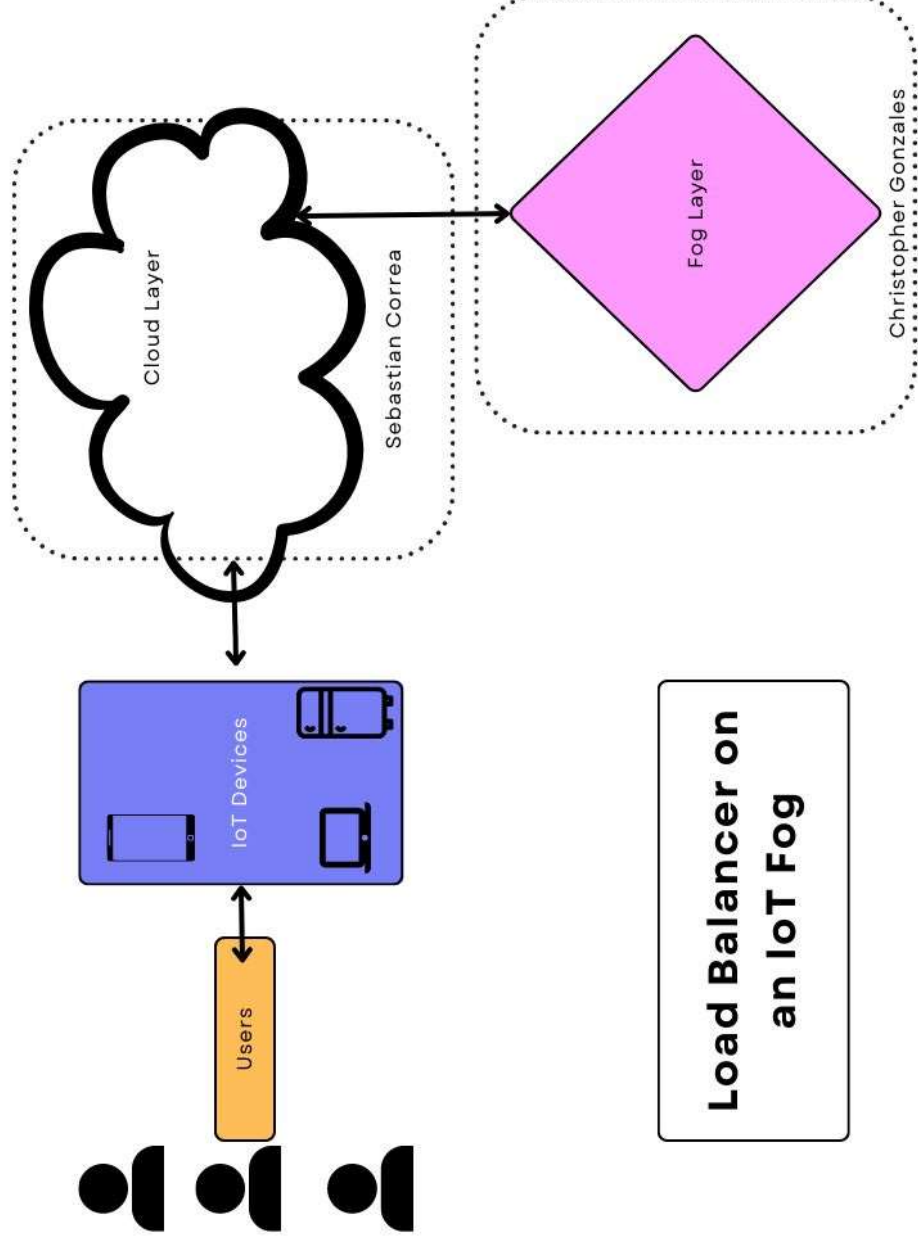
Major Project Changes for 404

- Demo
  - Old demo was inefficient
  - New demo easier to present
- Cloud
  - AWS too expensive
  - Microsoft Azure cheaper and compatible.

Project Timeline

Date	Progression	Status
8/24/23	Test Pi's condition and choose new cloud service	Completed
8/31/23	Re-establish connection with pi	Incomplete
9/7/23	Test website on Pi	Incomplete
9/14/23	Work on Fault Tolerance	Incomplete
9/21/23	Configure Kubernetes Dashboard for demo	Incomplete

Edge Node Subsystem

Christopher Gonzales

Accomplishments since 403 hrs of effort	Ongoing progress/problems and plans until the next presentation
- Necessary Software Installed - RPi 4 Performance Check	- Writing code for demo website hosted on the Pi - Deploy website hosted in container - Register Pi as Worker node in k3s cluster

sebascor@raspberrypi:~

top - 11:35:34 up 20 min, 3 users, load average: 1.99, 3.13, 2.29
Tasks: 208 total,   1 running, 207 sleeping,   0 stopped,   0 zombie
%Cpu(s): 1.7 us, 1.2 sy, 0.0 ni, 97.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 1849.2 total,   100.0 free,   670.6 used,   1078.7 buff/cache
MiB Swap: 100.0 total,   24.2 free,   75.8 used.   1084.9 avail Mem

  PID USER   PR NI   VIRT   RES   SHR S %CPU %MEM   TIME+ COMMAND
 1708 root   20   0 1194600 430848 58160 S   7.8 22.8   3:33.61 k3s-server
 3439 sebascor 20   0 751668 23120 10112 S   1.6   1.2   0:25.59 metrics-server
11310 sebascor 20   0   11344   2944   2508 R   1.0   0.2   0:00.51 top
 1765 root   20   0 771684 628840 40072 S   0.7   3.3   0:21.87 containerd
 2862 root   20   0 762336 30004 17920 S   0.7   1.6   0:06.94 coredns
    1 root   20   0   35084   9208   6872 S   0.3   0.5   0:20.75 systemd
   37 root   20   0   0   0   0 I   0.3   0.0   0:04.08 kworker/0:2-events_freezable
 2487 root   20   0 720696   9864   7176 S   0.3   0.5   0:01.65 containerd-shim
 2489 root   20   0 720440   9108   6984 S   0.3   0.5   0:01.60 containerd-shim
 2879 root   20   0 733236 13376   6828 S   0.3   0.7   0:01.55 local-path-prov

Cloud Subsystem

Sebastian Correa

| Accomplishments since 403 65 hrs of effort	Ongoing progress/problems and plans until the next presentation
• Created the kubernetes cluster • Established connection with pi	• Moving from AWS to Microsoft Azure • Adjust load balancing algorithm. • Explore Kubernetes Dashboard • Looking into KubEdge

Edge Node Subsystem

Christopher Gonzales

- 5V 3A Type C Power Supply
- Ethernet Connection
- Load Balancer Hosted in the Cloud

Changes:
- Different Demo Method
  - Website
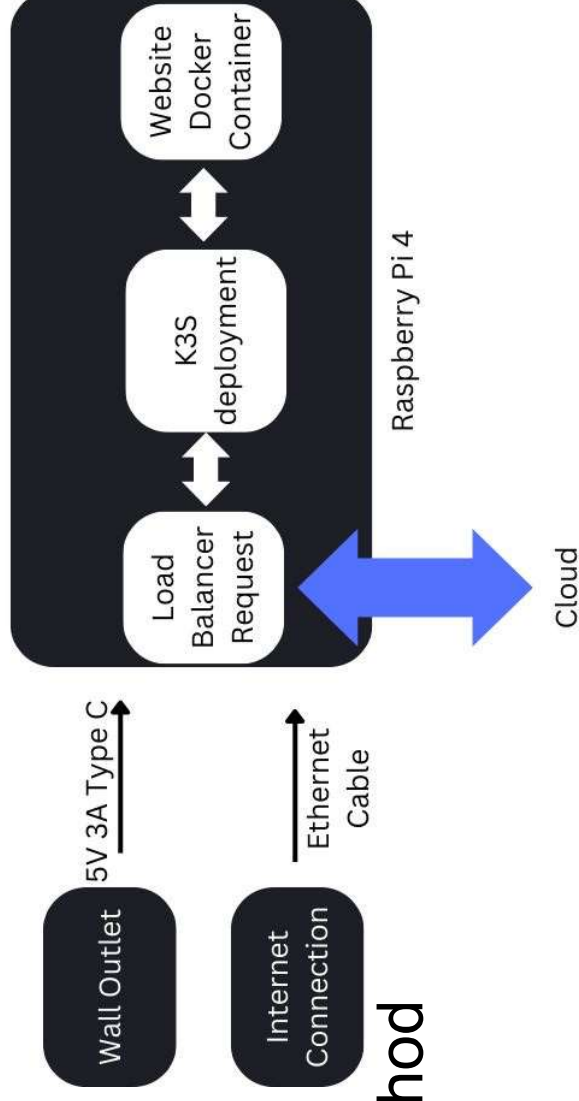
Edge Node Subsystem

Christopher Gonzales

- Necessary software downloaded
- Able to create, manage, and host simple containers
- Previous demo method froze pi
- New Demo Method

```
sebascor@raspberrypi:~/403/load_balancer $ docker ps -a
CONTAINER ID   IMAGE          COMMAND             CREATED         STATUS              PORTS     NAMES
553a834151a0   load_balancer  "echo 'This is the l…"  4 minutes ago  Exited (0) 4 minutes ago            load_balancer
6610bdc9a27d   server2        "echo 'This is serve…"  6 minutes ago  Exited (0) 6 minutes ago            server_container2
05c99989b121   server1        "echo 'This is serve…"  8 minutes ago  Exited (0) 8 minutes ago            server_container1
sebascor@raspberrypi:~/403/load_balancer $
```

```
sebascor@raspberrypi:~/403 $ docker build -t minecraft-server -f minecraft-server/minecraft_dockerfile .
[+] Building 0.9s (10/10) FINISHED
 => [internal] load build definition from minecraft_dockerfile                                            0.1s
 => => transferring dockerfile: 965B                                                                      0.0s
 => [internal] load .dockerignore                                                                         0.1s
 => => transferring context: 2B                                                                           0.0s
 => [internal] load metadata for docker.io/arm64v8/openjdk:17-jdk-slim                                    0.7s
 => [1/5] FROM docker.io/arm64v8/openjdk:17-jdk-slim@sha256:d732b25fa8a69u4d3124766805d086aeaaa6c9e2fbc3aef0482b8190d5e0e32e10  0.0s
 => [internal] load build context                                                                         0.0s
 => => transferring context: 295B                                                                         0.0s
 => CACHED [2/5] WORKDIR /minecraft                                                                       0.0s
 => CACHED [3/5] COPY /minecraft-server .                                                                 0.0s
 => CACHED [4/5] COPY minecraft-server/server.properties .                                                0.0s
 => CACHED [5/5] COPY minecraft-server/server.jar /minecraft/minecraft_server.jar                         0.0s
 => exporting to image                                                                                    0.0s
 => => exporting layers                                                                                   0.0s
 => => writing image sha256:7990b8389eac7ad054b140631b3c00681e4cdded137d5e9e1fa451e031cc0bd23             0.0s
 => => naming to docker.io/library/minecraft-server                                                       0.0s
sebascor@raspberrypi:~/403 $ docker run -d -p 25565:25565 -e "EULA=true" minecraft-server
```

Cloud Subsystem

- Takes in unfulfilled requests
- Stores data and runs virtual machines
- Hosts the K3s cluster

Changes:
- Microsoft Azure
- Kubernetes Dashboard

Cloud

Servers

K3s Cluster

Load Balancer

Fog Layer
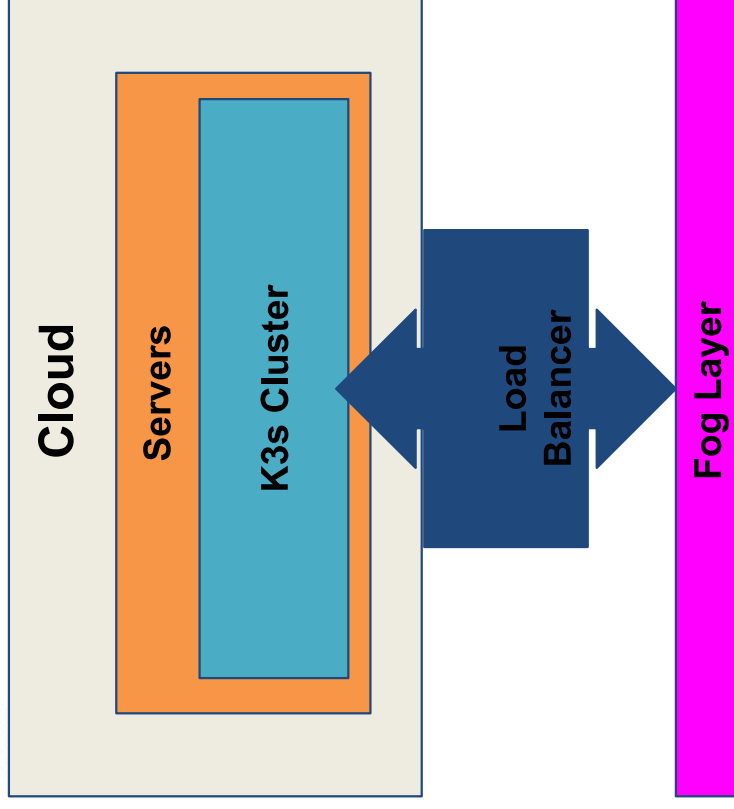
Parts Ordering Status

- Cloud Setup
  - Adjust cloud usage to go with the budget
- Overall, all parts necessary have been received

Execution plan

Work	End Date	Owner	Status	Completion Date
Check pi condition	8/24	Christopher	Complete	8/24
Choose new cloud service	8/24	Sebastian	Complete	8/24
Status Update Presentation 1	8/28	All	Incomplete	N/A
Re-establish connection with pi	8/31	All	Incomplete	N/A
Convert all virtual machines to Microsoft Azure	8/31	Sebastian	Incomplete	N/A
Build website	9/7	Christopher	Incomplete	N/A
Configure kubernetes dashboard	9/7	Sebastian	Incomplete	N/A
Status Update Presentation 2	9/11	All	Incomplete	N/A
Test website	9/14	All	Incomplete	N/A
Configure fault tolerance	9/14	All	Incomplete	N/A
Adjust demo visuals	9/21	Sebastian	Incomplete	N/A
Add more features to website	9/21	Christopher	Incomplete	N/A
Status Update Presentation 3	9/25	All	Incomplete	N/A

Execution Plan

Work	End Date	Owner	Status	Completion Date
Bug fix on website	9/28	Christopher	Incomplete	N/A
Work on cloud efficiency	10/5	Sebastian	Incomplete	N/A
Status Update Presentation 4	10/9	All	Incomplete	N/A
Test the system	10/12	All	Incomplete	N/A
Work on any issues	10/19	All	Incomplete	N/A
Monitor the System	10/19	Sebastian	Incomplete	N/A
Status Update Presentation 5	10/23	All	Incomplete	N/A
Final Presentation	11/15	All	Incomplete	N/A

Validation plan

Task	Specification	Summary	Result	Owner
Cloud Response Time	<500ms	Amount of time it takes the Cloud to respond to Load Balancer		Sebastian
Edge Device Runtime	<500ms	Amount of time it takes the Edge Device to respond to Load Balancer		Christopher
Edge and Cloud Transmission Time	<500ms	Amount of time it takes for the edge device and cloud to respond to one another		All
Reading Traffic	<500ms	Amount of time it takes the K3s cluster to read the incoming traffic		Sebastian
Minimum Number of Test Cases	50	50 test cases, or traffic data, being sent to our system for testing		All
Minimum Number of applications to run	1	Our application we're going to use for testing		All
Load Balancing Test	40%-60%	Ensure that the load is distributed equally between rpi website container and VM container		All
Failover Test	<500 ms	Shut down one edge node and ensure traffic is redirected to working node within seconds		All

Questions?