The Design Behind Kerby

Team C1
Kanvi Shah, Mrinmayee Mandal, Neville Chima

Recap: Kerby's Use Case

SPOT HERO

- Current parking apps only work in garages or enclosed parking lots.
- What about street parking?

Recap: Kerby's Requirements

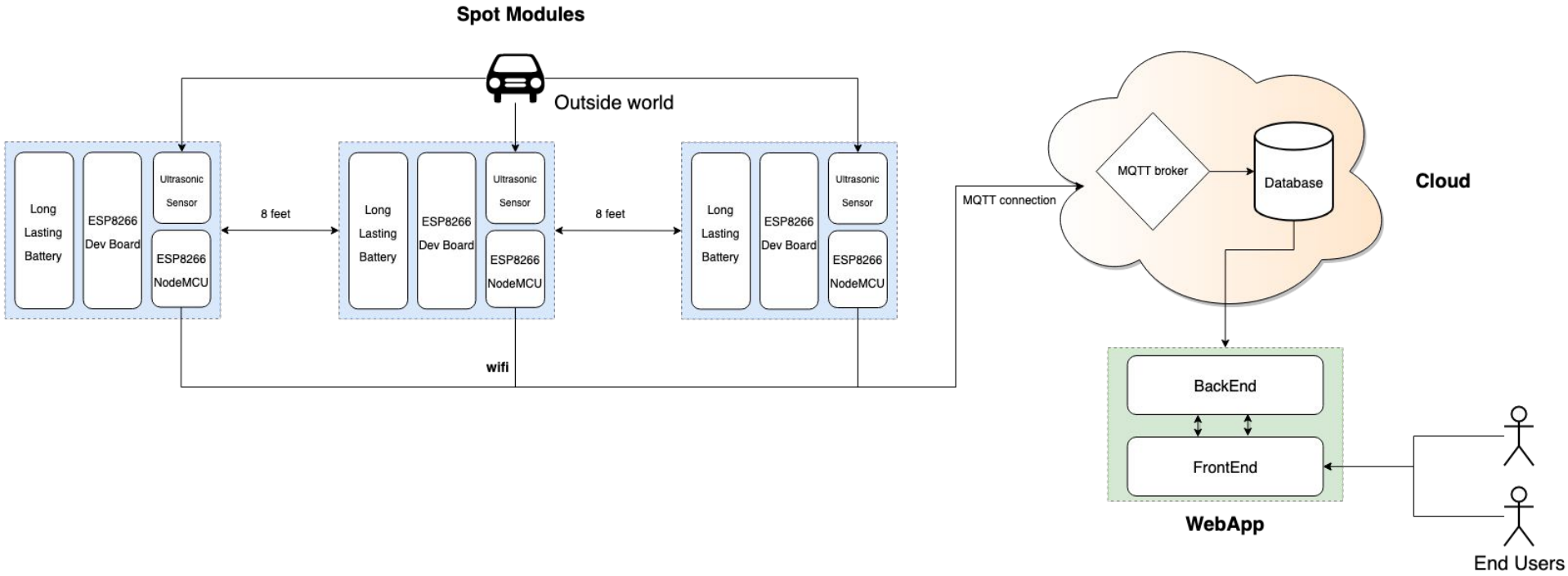
Description	Motivation
Car length < 16 ft	Avg. car length = ~14.5ft
Location accuracy within 30ft	30 ft = ~2 cars
Sensors wake up every 5 min	Takes people at least ~3 min to park and leave
Find street parking within <0.5mi to destination	To keep benefit of street parking over garage parking
Cheap spot modules < $50	To enable future scalability

Solution Approach

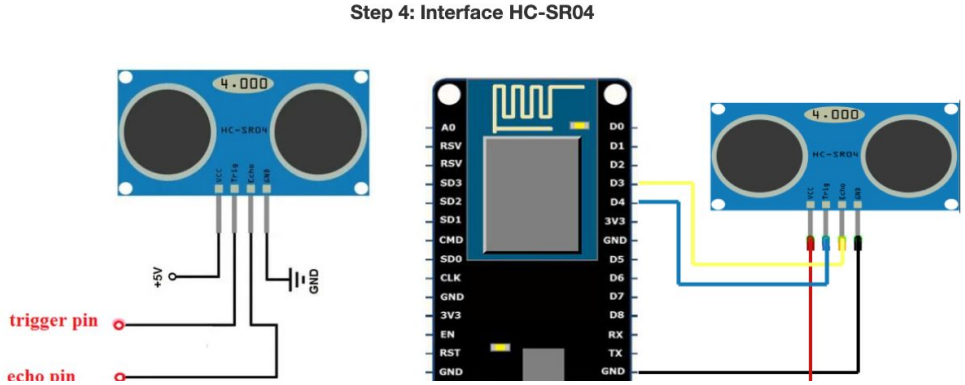
The Main Problem:
wasting time

The Solution:
Easy installation, modular design, control in the hands of the drivers

Kerby's System Specifications



Implementation Plan: Hardware



Spot components: HC-SRO4 sensor + ESP8266 Wi-fi module

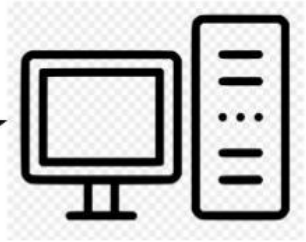
Programmability:

- MicroPython firmware
- Micro-USB cable
- Ampy to load lightweight programs from PC to 2MB flash memory
- Machine Pin library to take sensor readings

Implementation Plan: Software

a) Deploy Atlas cluster **b)** Provision CloudMQTT configurations - topic, ports, clients etc **c)** Initialize MongoDB collection(s)

SUBSCRIBE over Wi-Fi



PUBLISH over Wi-Fi

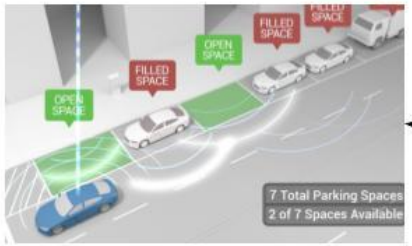
a) Install paho-mqtt client in MCU programming interface **b)** Bind wifi (SS)ID to sensor data **c)** Publish sensor data to MQTT broker

a) Install pymongo library **b)** Subscribe to MQTT broker c) Execute callback on message_received

a) Install pymongo library **b)** Obtain Atlas credentials **c)** Massage data for persistence in MongoDB **b)** Insert JSON blobs into MongoDB collections

mongo DB

a)Start webapp on host server **b)** Execute periodic MongoDB batch queries **c)** Run block detection algorithm **d)** Provide payload response to client request



Implementation Plan: E2E + Subtleties

- X spots : 1 publisher to subscriber ratio for single topic
- Local publishing, Remote subscription
- MQTT Acknowledgement Semantics (Quality of Service)
- Distinct Backend (Subscribe + DB Insert) + Web (Client request) entrypoint scripts

Testing - Verification & Metrics

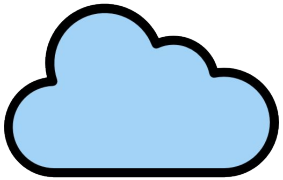
Requirement	Measurement	Goal
Find closest street parking to destination	Using google maps api to find distance between given spot and destination	< 0.5 mi
Accurate parking location	Distance between provided location and actual parking spot in real world	< 30ft
Accurate representation of real world	Confusion matrix from testing with large number or users and requests	< 20% False Positive and Negative
Relatively cheap for scalability	Compare to cost of regular parking meters	< $50 per spot module
Easy-to-use web app	User Testing and recording ratings from 1(bad) to 5(great)	> 3.5/5 stars on average
Easy to install	User Testing and recording time	< 5 min on average

Testing - MVP

1 User

2 Spot Modules

Web software



- Using 2 ends of Margaret Morrison parallel parking
- Using different requests for different destinations on campus

Project Management

kerby: your curbside parking buddy

TEAM C1 **NAMES** Mrinmayee Mandal, Kanvi Shah, Neville Chima

WBS NUMBER	TASK TITLE	TASK OWNER	START DATE	DUE DATE	DURATION	PCT OF TASK COMPLETE
1	**Initial Development**					
1.1	Sensor Research	Minu	2/7/22	2/11/22	5	0%
1.2	IoT Device Research	Kanvi	2/7/22	2/11/22	5	0%
1.3	Database App Research	Neville	2/7/22	2/13/22	7	0%
1.4	Module Set-up (w/o portable power)	Kanvi	2/11/22	2/16/22	6	0%
1.5	Power Supply Research	Minu	2/11/22	2/16/22	6	0%
1.6	Sensor-IoT communication setup	Neville	2/13/22	2/19/22	7	0%
1.7	Design Documents and Presentation	All	2/16/22	2/19/22	4	0%
2	**MVP**					
2.1	Module Set-up (w/ portable power)	Kanvi	2/20/22	2/23/22	4	0%
2.2	Web App Back End: simplest algorithm	Minu	2/20/22	2/23/22	4	0%
2.3	Multi-sensors: Duplicate Module	Minu	2/23/22	2/25/22	3	0%
2.4	IoT Network: connecting >1 modules	Neville	2/25/22	2/28/22	4	0%
2.5	Web App Front End: simplest version	Kanvi	2/25/22	2/28/22	4	0%
2.6	MVP Integration and Testing	All	2/28/22	3/3/22	4	0%
3	**Improvement**					
3.1	MVP Improvement Research	All	3/13/22	3/16/22	4	0%
3.2	Improve Hardware Module	Kanvi	3/17/22	3/22/22	6	0%
3.3	Improve Software Algorithm	Minu	3/17/22	3/22/22	6	0%
3.4	Improve Communication Protocols	Neville	3/17/22	3/22/22	6	0%
3.5	Web App Front End: UI/UX v2.0	Kanvi	3/23/22	3/25/22	3	0%
3.6	Duplicate Two Module System	Minu	3/23/22	3/29/22	7	0%
4	**Final**					
4.1	Final Field Implementation	All	3/30/22	4/1/22	3	0%
4.2	Final Field Testing	All	4/2/22	4/8/22	7	0%
4.3	Final Assessment of Results	All	4/9/22	4/15/22	7	0%
4.4	Final Paper and Presentation	All	4/16/22	4/22/22	7	0%