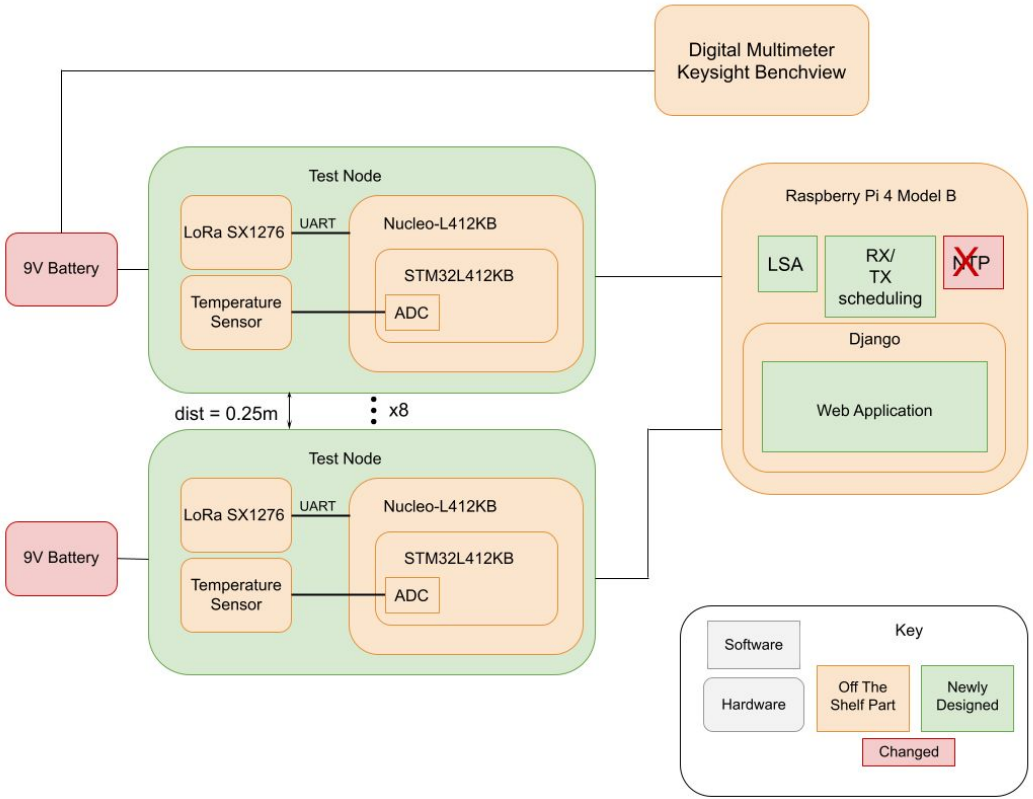
USE CASE REQUIREMENTS

Motivation: Locations like California experience multiple wildfires every year, spanning large areas (7,490 fires in 362,455 km)

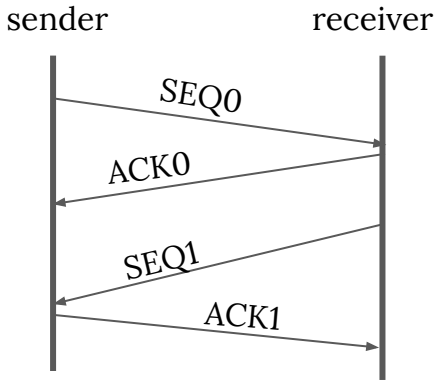
Goal: Notify wildland firefighters of wildfire locations

Use Case Requirement	Desired Quantitative Metrics
Fire Detection Accuracy	> 90-95%
Notification Timing	30 mins
Low Power	1 mo. maintenance

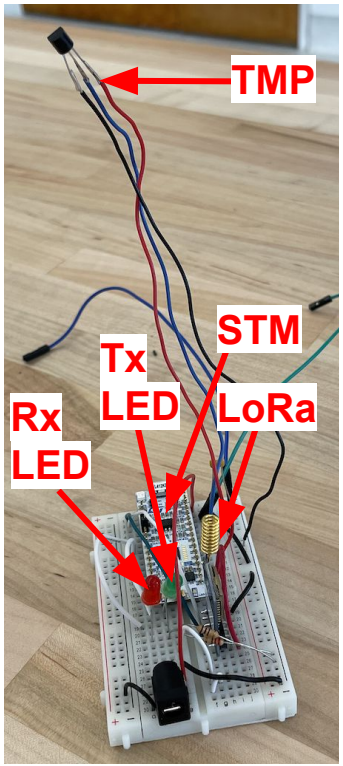
MODIFIED SOLUTION APPROACH/RESCOPING

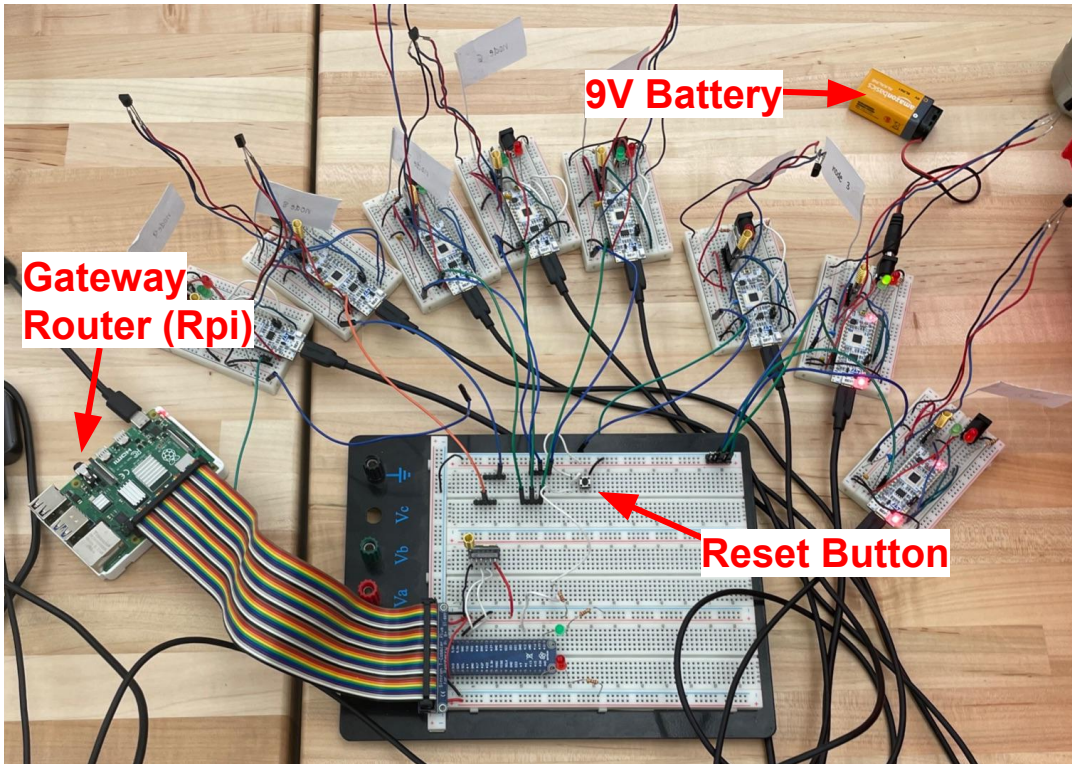


- Assuming a certain amount of packet losses
  - Assume at most N packet errors can occur in each phase
- Web app does not automatically reload when JSON is updated with new data
  - Manual reload
- Removed NTP from protocols



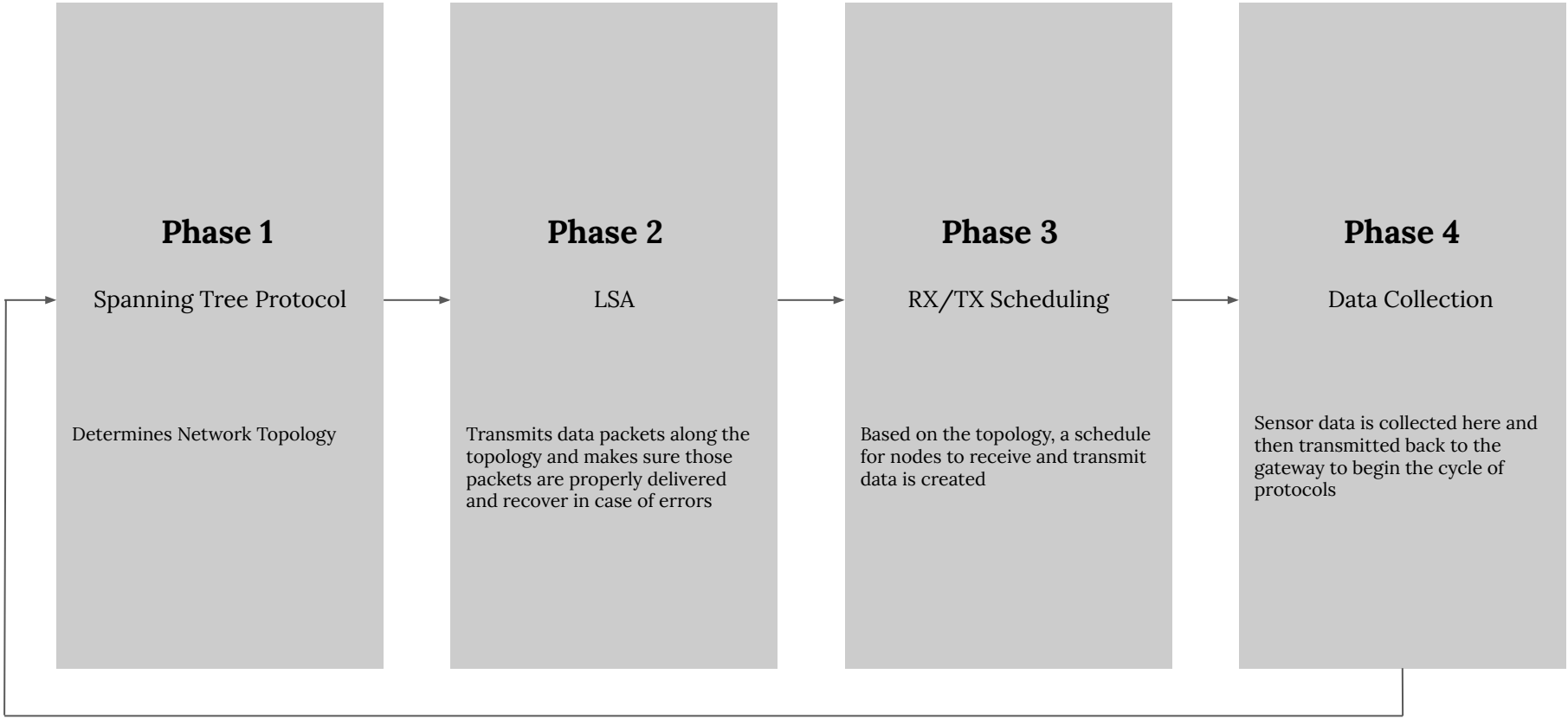
COMPLETE SOLUTION: NETWORK HARDWARE



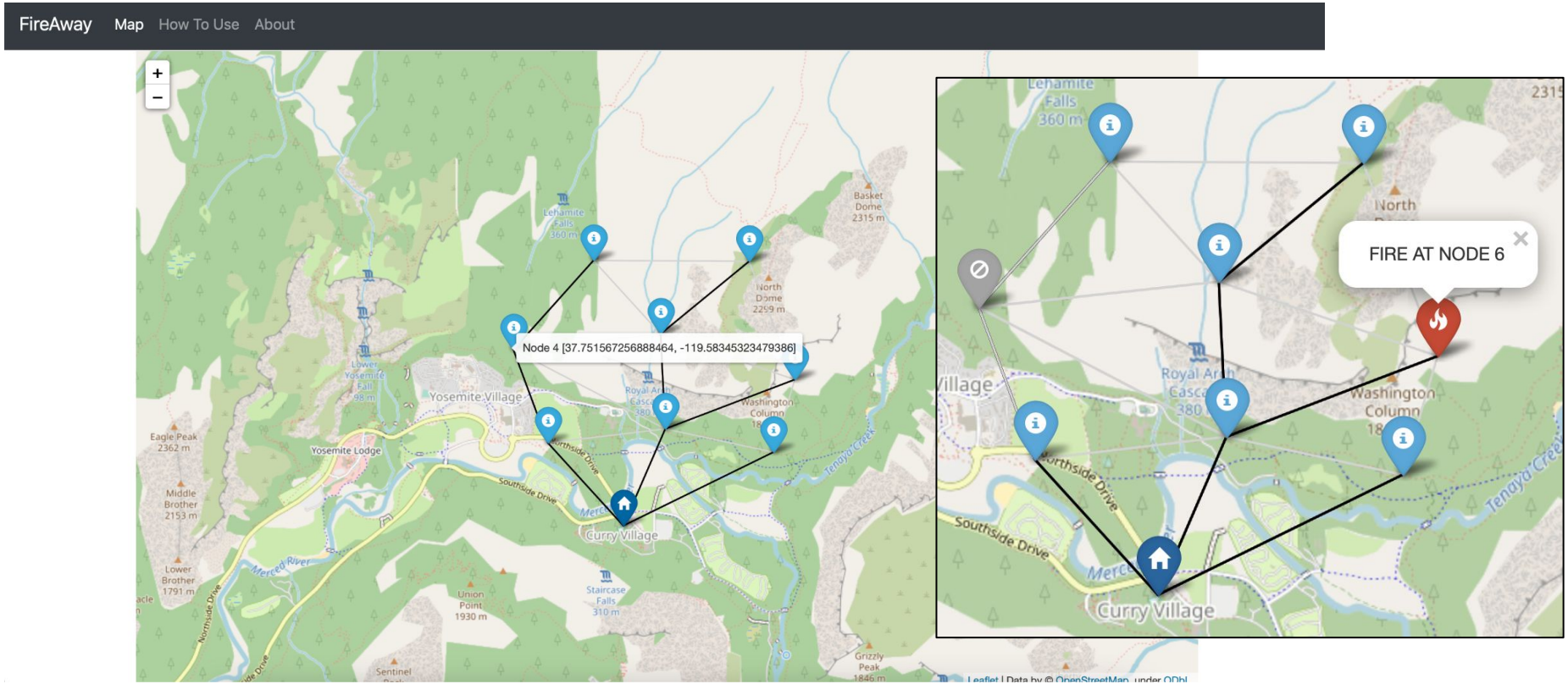


Demo Expectation: Nodes will be distributed around the room within 0.25m

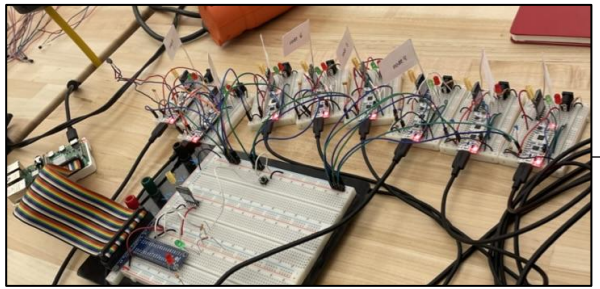
COMPLETE SOLUTION: NETWORK SOFTWARE



COMPLETE SOLUTION: WEB APPLICATION

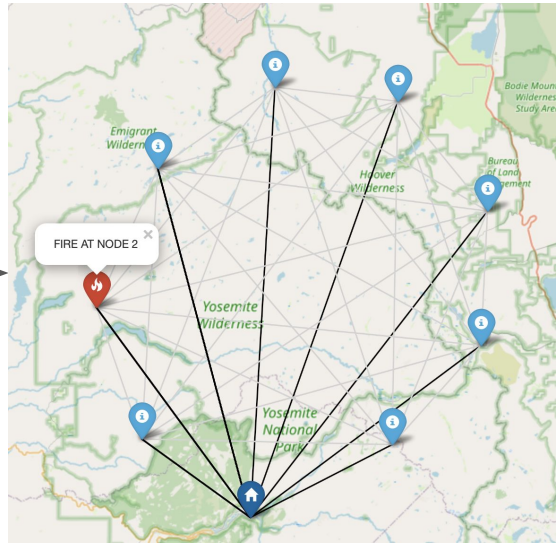


COMPLETE SOLUTION: SAMPLE RUN



Phases 1-3: Network Configuration

Phase 4: Data Collection (Heating of Node 2)



Data Notification

TEST, VERIFICATION, & VALIDATION: CHANGES

- Turn on **heat gun** next to sensor and check that response time is within 30 mins
  - Match → Heat Gun: Safety and Consistency
- **No longer assume that a node failing/going offline is a fire**
  - Not an assumption that is realistic to make
  - Detecting node failure within 30 mins of node going offline remains
- Maximize low power modes in the framework of our network protocol:
  - Turn nodes on for a period of time and measure how much battery is drained
- If clock synchronization fails, we were going to use GPS to calibrate RTC
  - We no longer are doing clock synchronization and are using ACKs to recover from any packet errors and fails

TEST, VERIFICATION, & VALIDATION: TEST CASES

- Topology Tests:
  - 2-node exchange (one node being gateway)
  - 4-node tree
  - 8-node line topology
  - **8-node fully connected → Test being used currently**
  - 8-node mesh topology (FINAL/IDEAL)*
- Types of Tests:
  - Control (No fire set)
  - 1 fire being set
  - Multiple fires being set*
  - Node going offline/dead*

*Tests that are currently not working/not been tested yet

TEST, VERIFICATION, & VALIDATION: RESULTS

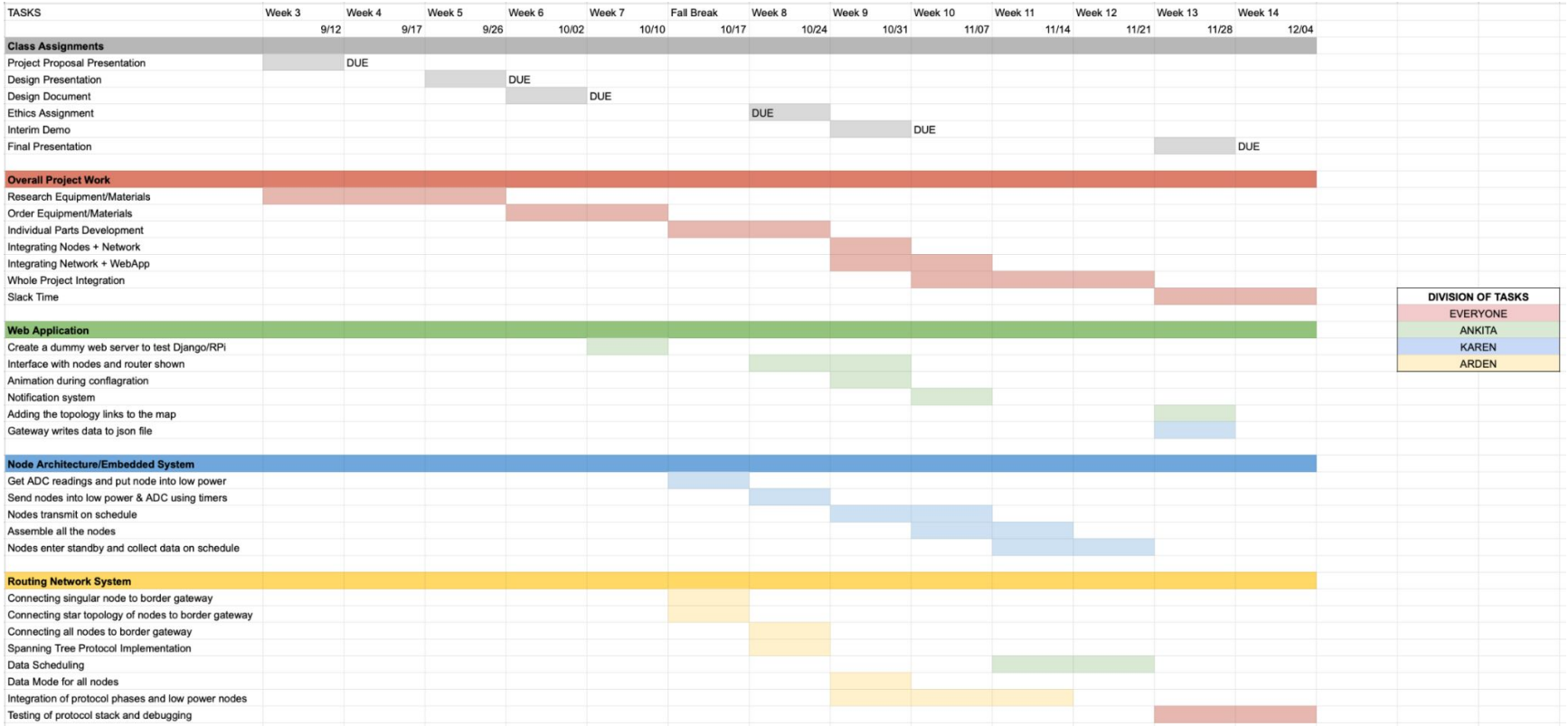
Design Requirement	Desired Quantitative Metrics	Results
Fire Detection Accuracy*	> 90-95%	75%
Notification Timing/Web App displays fire location*	30 mins	7-8 mins
Detect node failure	<= 30 mins of node going offline	(TO BE TESTED)
Maximize low power modes (measure how much battery has drained)	battery should last for 1 mo.	(TO BE TESTED)

*Results collected by running 10 tests with a 8-fully connected topology

TEST, VERIFICATION, & VALIDATION: TRADE-OFFS

- Acknowledgements (ACKs)
  - Transmission a lot more reliable
    - Sender now knows if receiver received a packet
  - Sending more packets = more time, more power
  - Reliability > Low Power in terms of priority

PROJECT MANAGEMENT: SCHEDULE



TASKS	Week 3	Week 4	Week 5	Week 6	Week 7	Fall Break	Week 8	Week 9	Week 10	Week 11	Week 12	Week 13	Week 14
	9/12	9/17	9/26	10/02	10/10	10/17	10/24	10/31	11/07	11/14	11/21	11/28	12/04
Class Assignments													
Project Proposal Presentation		DUE											
Design Presentation				DUE									
Design Document					DUE								
Ethics Assignment							DUE						
Interim Demo									DUE				
Final Presentation													DUE
Overall Project Work													
Research Equipment/Materials													
Order Equipment/Materials													
Individual Parts Development													
Integrating Nodes + Network													
Integrating Network + WebApp													
Whole Project Integration													
Slack Time													
Web Application													
Create a dummy web server to test Django/RPi													
Interface with nodes and router shown													
Animation during conflagration													
Notification system													
Adding the topology links to the map													
Gateway writes data to json file													
Node Architecture/Embedded System													
Get ADC readings and put node into low power													
Send nodes into low power & ADC using timers													
Nodes transmit on schedule													
Assemble all the nodes													
Nodes enter standby and collect data on schedule													
Routing Network System													
Connecting singular node to border gateway													
Connecting star topology of nodes to border gateway													
Connecting all nodes to border gateway													
Spanning Tree Protocol Implementation													
Data Scheduling													
Data Mode for all nodes													
Integration of protocol phases and low power nodes													
Testing of protocol stack and debugging													

DIVISION OF TASKS
EVERYONE
ANKITA
KAREN
ARDEN

WHAT'S LEFT? & LESSONS LEARNED

- Fixing Bugs
  - Make sure to check the extensiveness and reliability of a component's documentation
  - Always expect that integration of multiple parts reveals bugs/flaws
- Further testing
  - Testing if a node "dies"
  - Testing Low Power
- Adjusting Power Consumption and Timing:
  - More testing needs to be done and walking through working code to see if any timing and power usages can be pared down