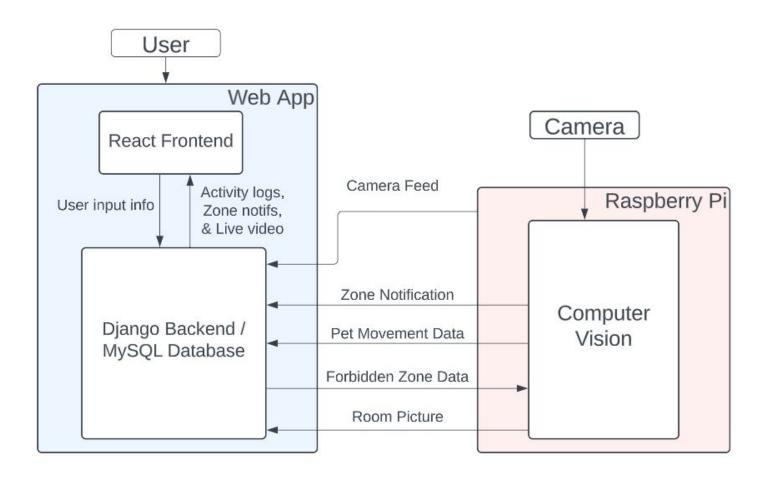
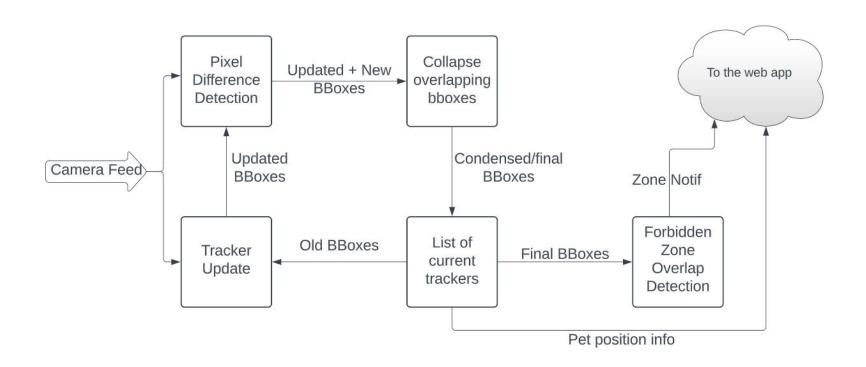
Use Case Requirements

The overall goal	Our use case requirement(s)	The design requirement(s)				
Detect and report when a pet goes somewhere it	False positives <10% of the time	Tracking accuracy w/in 1ft				
shouldn't	zone report speed <10 seconds	Zone detection w/in 1 sec				
Provide a log of pet activity		Tracking accuracy w/in 1ft				
	Logs are >90% accurate	New animal detection w/in 5 sec				
	system setup in <5 min	-				
Maintain system accessibility	system cost <\$100	Use RPi and simple camera				
	>95% of users can accomplish tasks easily	-				

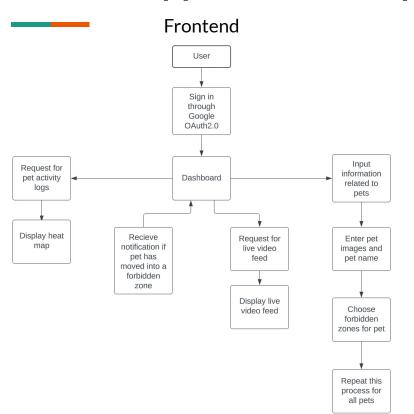
Solution Approach (Overall)



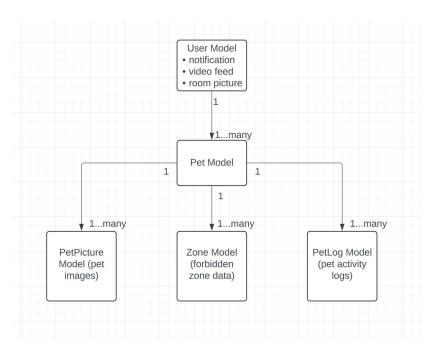
Solution Approach: Computer Vision



Solution Approach (Web App)

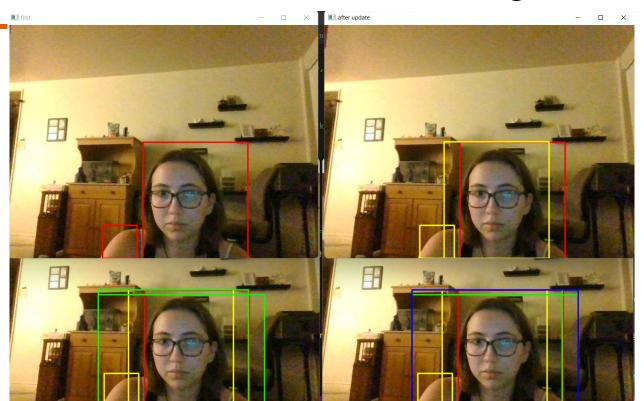


Backend



Complete Solution: Detection & Tracking

starting boxes from the last iteration (red)

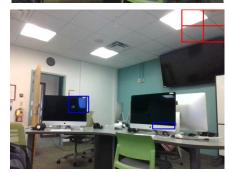


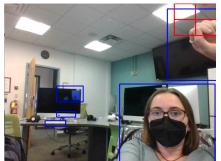
OpenCV tracker update (yellow)

New BBoxes from detected movement(s) (green) Collapse overlapping boxes: final result(s) in blue

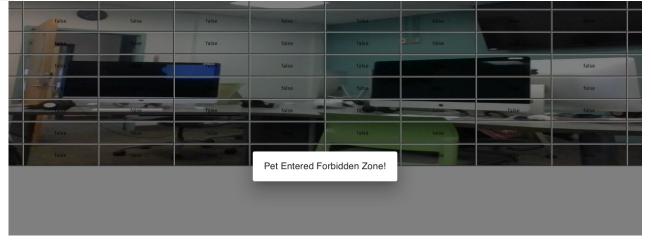
Complete Solution (Forbidden Zone Demo)



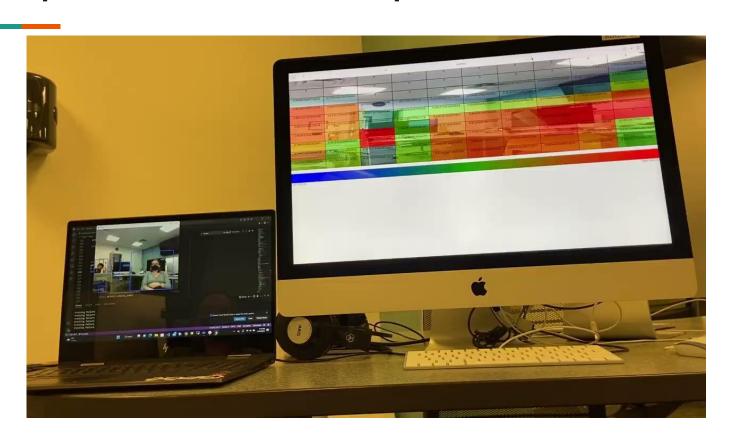








Complete Solution (Heat Map Demo)



Complete Solution (Other Features)

- Deployment
 - Frontend deployed through AWS S3 + Cloudfront
 - Backend deployed through AWS EC2 and Apache servers
- Live Video
 - User can request for live video feed of the room
 - CV -> Web App images encoded into byte arrays
- Login using Google OAuth2.0
 - Used for authentication and security purposes

Testing - Notification Speed



Test	Method	Goal	Result
Pet enter forbidden zone -> CV detection	Slow motion video (pet enters zone irl vs CV video feed)	< 1 second	0.625 seconds
Pet enter forbidden zone -> user notification	Slow motion video (CV video feed to notification)	< 10 second	1.125 seconds

- CV trade off is communication with the Web App more frequent requests = more accurate data on the web app, but lowers the frame rate
- Web App limiting factor is polling rate (frequency of GET requests) from frontend to backend to get notification data (second result ~ polling speed)
 - Tested with GET requests / 1 second
 - Trade off is higher polling rate means faster notification to user but more server resources spent on requests

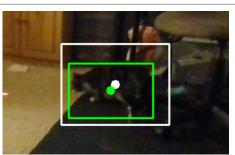
Testing - Tracking



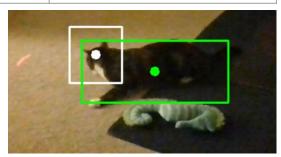
Test	Method	Goal	Result
Accuracy of tracker location(s)	Compare CV generated bounding box to human- chosen bounding box; Display the difference bbox centers, and estimate the distance	Within 1 foot	Generally around 3-6 inches
Detection speed of new animals	Slow motion video - pet enters frame irl -> bounding box appears	>5 seconds	~0.75 seconds on average

White: CV code

Green: human chosen











Test	Method	Goal	Result
System Cost	Adding user costs	< \$100	Raspberry Pi + Raspberry Pi Camera = \$80
System Setup	User Testing (10 participants)	< 5 min for each participant	TBD
Accomplishment of Website Tasks	User Testing (10 participants)	9 out of 10 users can do 4 core tasks successfully with minimal guidance	TBD

Project Management

Task	2/13	2/20	2/27	3/6	3/13	3/20	3/27	4/3	4/10	4/17	4/24	5/1
Major Deliverables/Deadlines				S								
Interim Demo				3								
Final Presentation				Р								
Poster PDF												
Video Demo				R								
Web App (Brandon)												
Heat map of pet activity (frontend for pet activity logs)												
Deployment of Backend				N								
Design frontend for notifying users that pet entered forbidden area zone				17. 77.								
Display live video feed for user on request				G								
User can login through Google OAuth				В								
Design and implement full user flow of website				Ь								
Deployment of Frontend				R								
Computer Vision & rPi (Rebecca)												
Explore openCV video tracking on laptop				E								
Get internet tutorial code to run on rPi				Α								
Rough draft of CV code on laptop - spot & track												
Get our CV code onto rPi				K								
Debugging/CV algorithm refinement												
Integration (Brandon & Rebecca)												
CV and Web App can communicate (on laptops)												
Rough forbidden zone detection and alerts												
Finalize method of CV/Web App communication (requests instead of sockets)												
Heat Map integration												
Live video feed												
Full system testing (First laptops, then pi) (simple cases)												
Longer/more realistic tests												