Introducing PetSTAR (System to Track And Report)

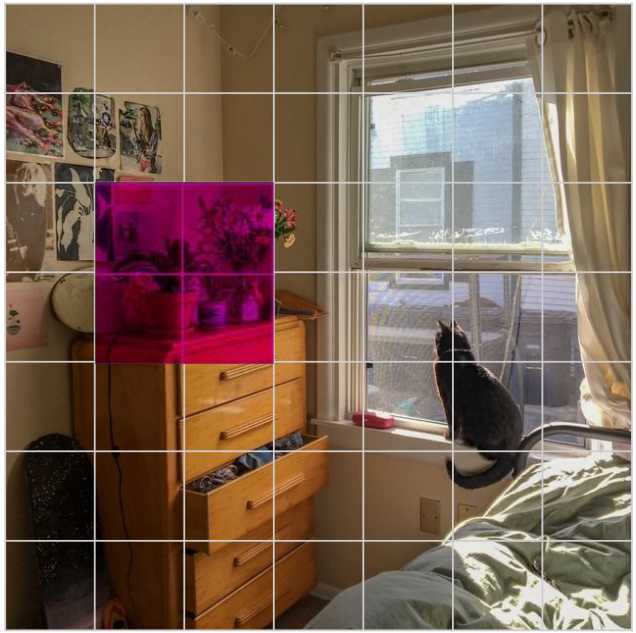
A more convenient approach to pet monitoring

The overall goal	Our use case requirement(s)
Detect and report when a pet goes somewhere it shouldn't	**False positives <10% of the time**
	zone report speed **<10 seconds**
Provide a log of pet activity, differentiating between animals if multiple	For each pet, logs are **>90% accurate**
Maintain system accessibility	system setup in **<5 min**
	system cost **<$100**
	>95% of users can accomplish tasks easily

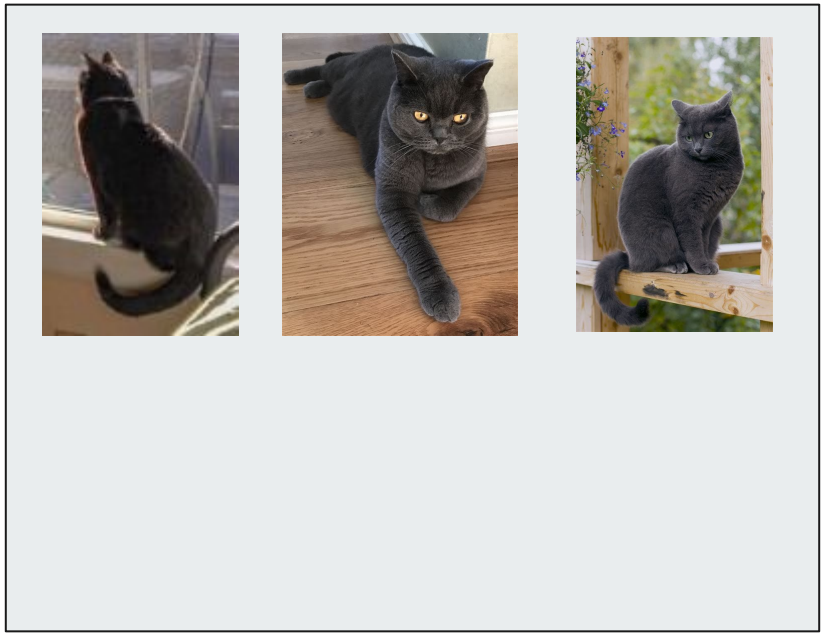
Design Requirements

The use case	The design requirement(s)
Report Speed <10 seconds	zone detection speed **<1 second**
	classification speed **<5 seconds**
Forbidden Zone false positives <10%	**tracking accuracy within 1 foot**
	classification accuracy **>90%**
Activity logs with >90% accuracy for each pet	tracking accuracy **within 1 foot**
	classification accuracy >90%
	new animal detection in **<5 seconds**
System cost <$100	Use a Raspberry Pi for primary computation
	Use a simple camera

Solution Approach: Step 1, user inputs data



User selects grid squares where the animal should not go, to be communicated to the tracking



User inputs images of their pet(s), to be communicated to the ML

Solution Approach: Step 2, identifying and monitoring

OpenCV detects animal movement via pixel differences

And sends the area of interest to the ML model to be identified

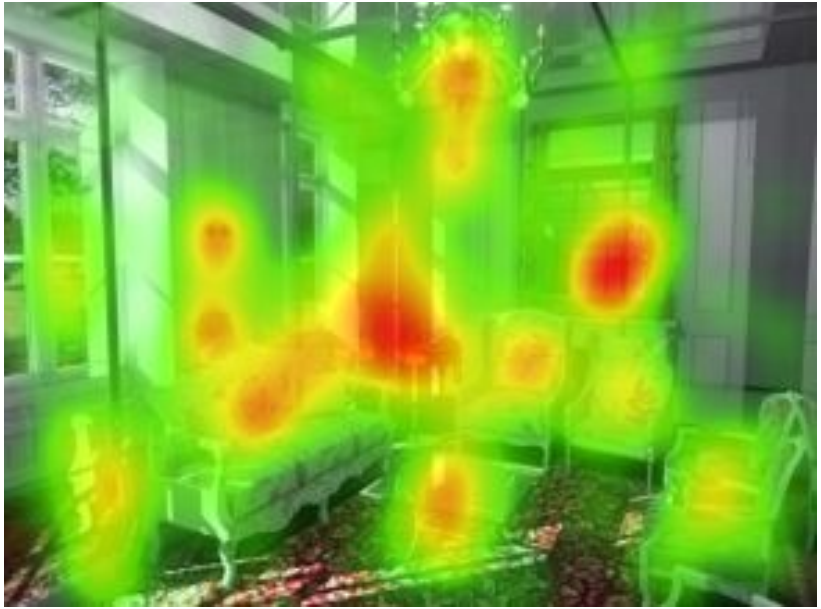
OpenCV is used to continue tracking the identified animal in the environment

If it overlaps with a forbidden square

ML validates the identity and sends a notification

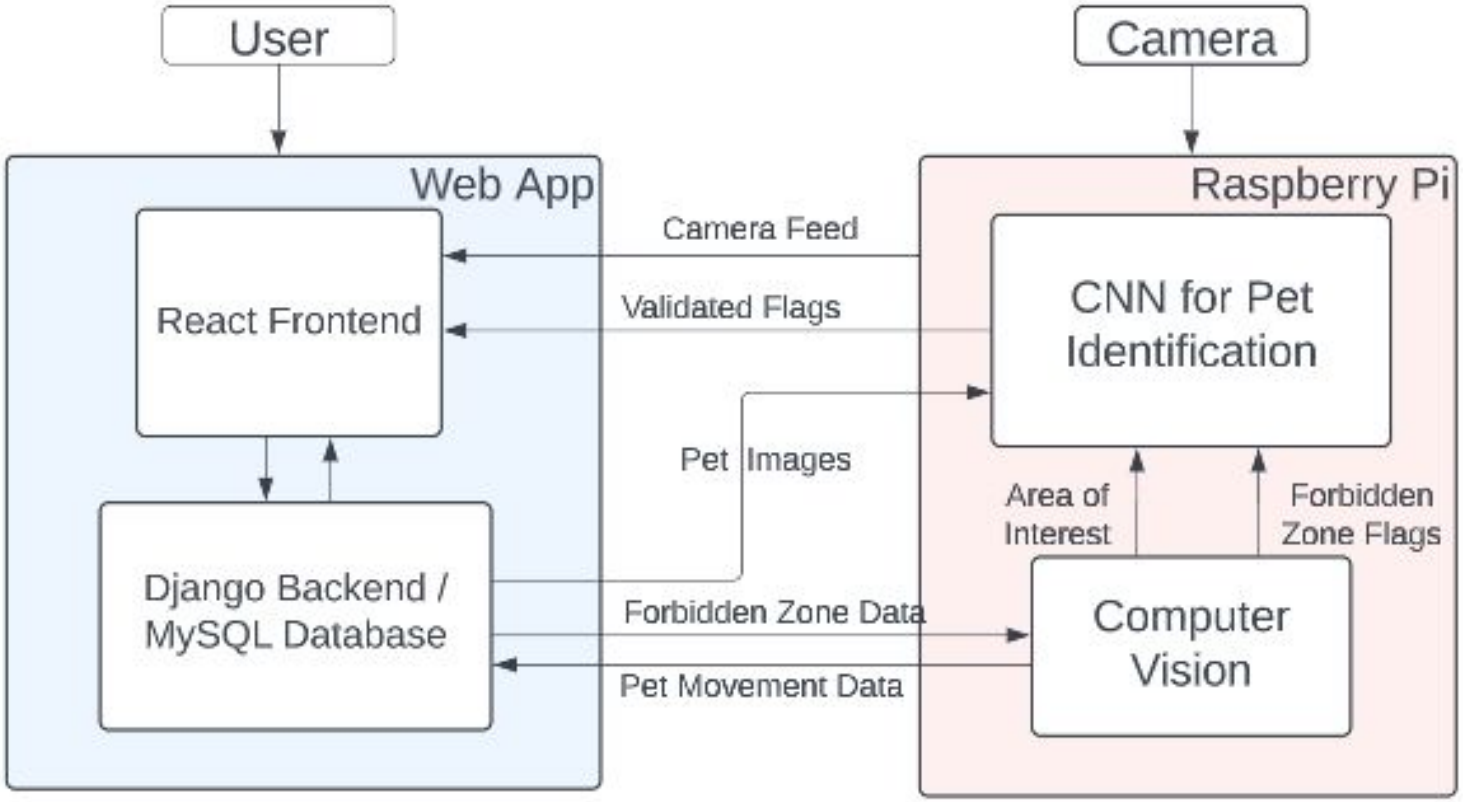
Solution Approach: Step 3, reporting back to user

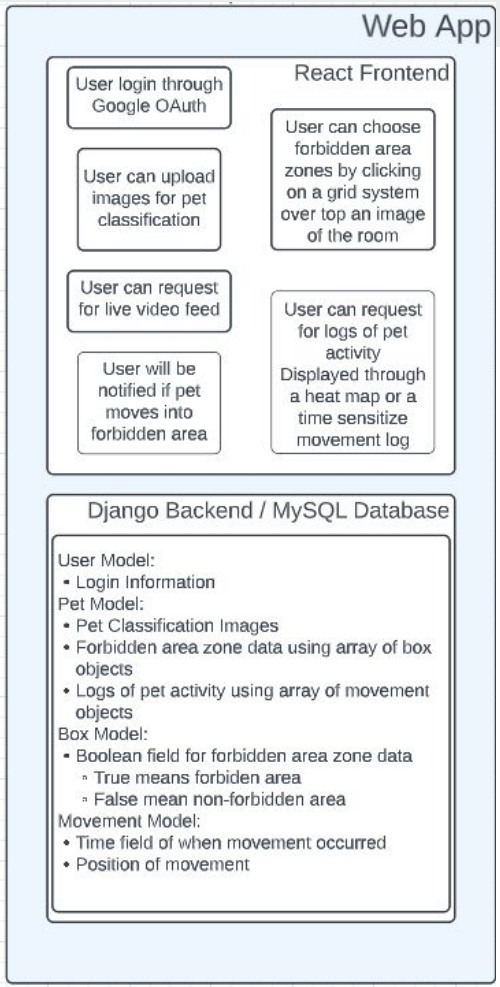
User receives a notification if pet enters a forbidden zone



User may request for logs of pet activity through a heat map or time sensitize graph

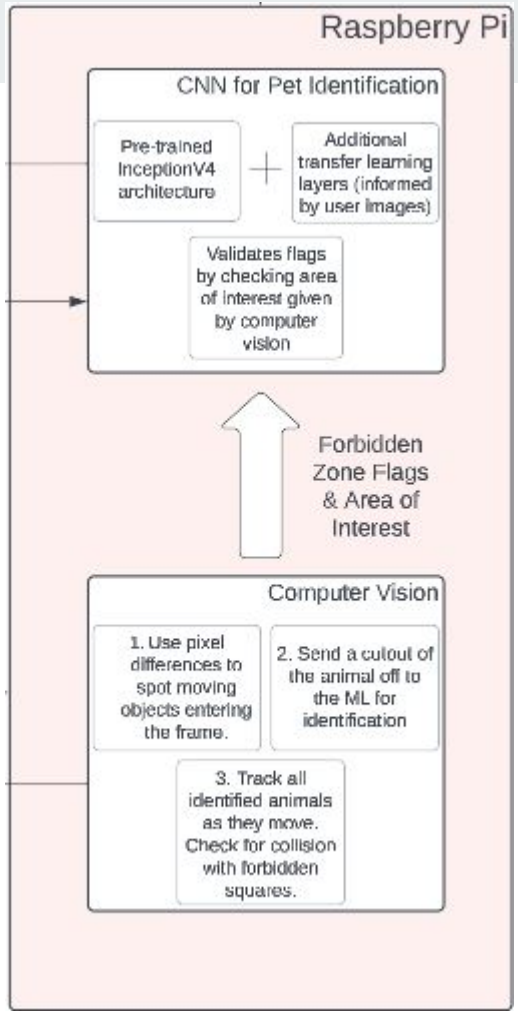
Top Level System Specification





The More Detailed Break Down: Web Application

- React on the frontend
  - Choose forbidden area zones
  - Request for pet activity logs
  - Notify if pet moved into forbidden zone
  - Request to see live video feed of pets
- Django on the backend
  - Using Django models to store data related to user options
- Safety concern is privacy issues related to displaying live video feeds of user's rooms on the website
  - Will use security tools given by Django



The More Detailed Break Down: Raspberry Pi

- OpenCV
  - Notice the animal when it enters
  - Send a cutout to the ML model for identification
  - Track the identified animal as it moves
  - Alert if it overlaps a forbidden area
- CNN
  - Pre-trained animal identification with InceptionV4
  - Incorporate user images for greater accuracy
  - Identify animals as they're noticed
  - Validate identity if forbidden zone flag is raised

	Hardware
	Software

Implementation plan

Off the shelf	Adapted from previous projects	From scratch
Raspberry Pi 4	OpenCV image tracking	Forbidden zone detection and flagging
RPi compatible camera	Implementing a CNN	Additional CNN layers for transfer learning
InceptionV4	User upload images	Frontend overall composition
React + Django	Login with Google OAuth	Backend Django models

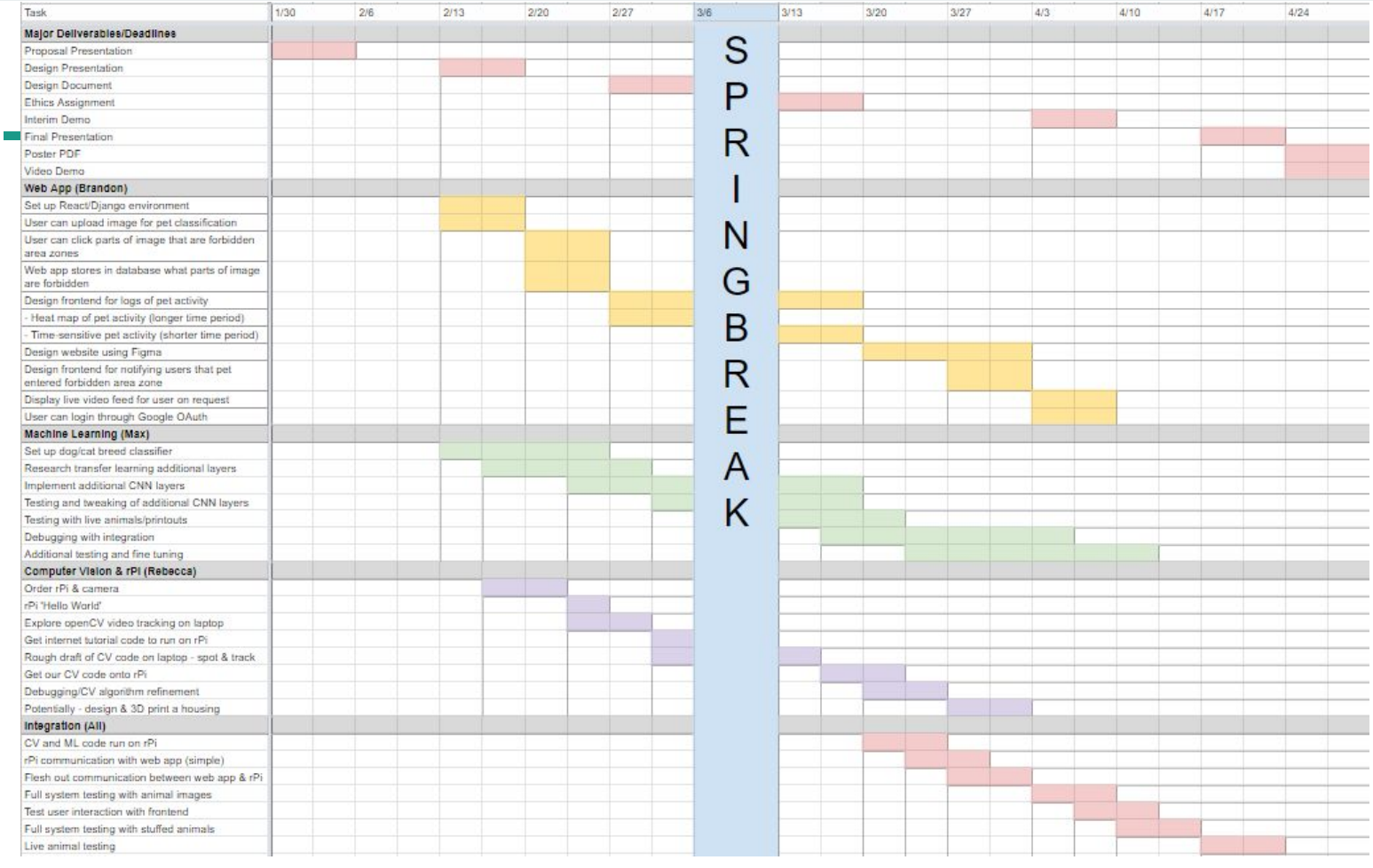
Testing: Computer Vision and Machine Learning

- Phases: images, stuffed animals, and live animal(s)
- Goal: meet our use case requirements - specifically accuracy and latency
- Success: goals are met across repeated trials and various species
- Mitigation: redefine scope or investigate other frameworks, if refining parameters is insufficient

Testing: Web Application and Overall System

- Phases: user testing and user surveys
- Goal: create an intuitive and user friendly web application and overall system
- Success:
  - Web App: enough users are satisfied with tasks or able to complete tasks on the web application
  - System: setup is easy and inexpensive, speed between components of system is satisfactory
- Mitigation: reduce time taken at slowest parts of system, revise tasks in web application to improve user experience

Schedule


Task
1/30
2/6
2/13
2/20
2/27
3/6
3/13
3/20
3/27
4/3
4/10
4/17
4/24
SPRING BREAK
Major Deliverables/Deadlines
Proposal Presentation
Design Presentation
Design Document
Ethics Assignment
Interim Demo
Final Presentation
Poster PDF
Video Demo
Web App (Brandon)
Set up React/Django environment
User can upload image for pet classification
User can click parts of image that are forbidden area zones
Web app stores in database what parts of image are forbidden
Design frontend for logs of pet activity
- Heat map of pet activity (longer time period)
- Time-sensitive pet activity (shorter time period)
Design website using Figma
Design frontend for notifying users that pet entered forbidden area zone
Display live video feed for user on request
User can login through Google OAuth
Machine Learning (Max)
Set up dog/cat breed classifier
Research transfer learning additional layers
Implement additional CNN layers
Testing and tweaking of additional CNN layers
Testing with live animals/printouts
Debugging with integration
Additional testing and fine tuning
Computer Vision & rPi (Rebecca)
Order rPi & camera
rPi 'Hello World'
Explore openCV video tracking on laptop
Get internet tutorial code to run on rPi
Rough draft of CV code on laptop - spot & track
Get our CV code onto rPi
Debugging/CV algorithm refinement
Potentially - design & 3D print a housing
Integration (All)
CV and ML code run on rPi
rPi communication with web app (simple)
Flesh out communication between web app & rPi
Full system testing with animal images
Test user interaction with frontend
Full system testing with stuffed animals
Live animal testing