ChaseMe Alarm Clock

Team B1: Peizhi Yu, Echo Gao, Yuhan Xiao

Presenter: Yuhan Xiao

Application Area

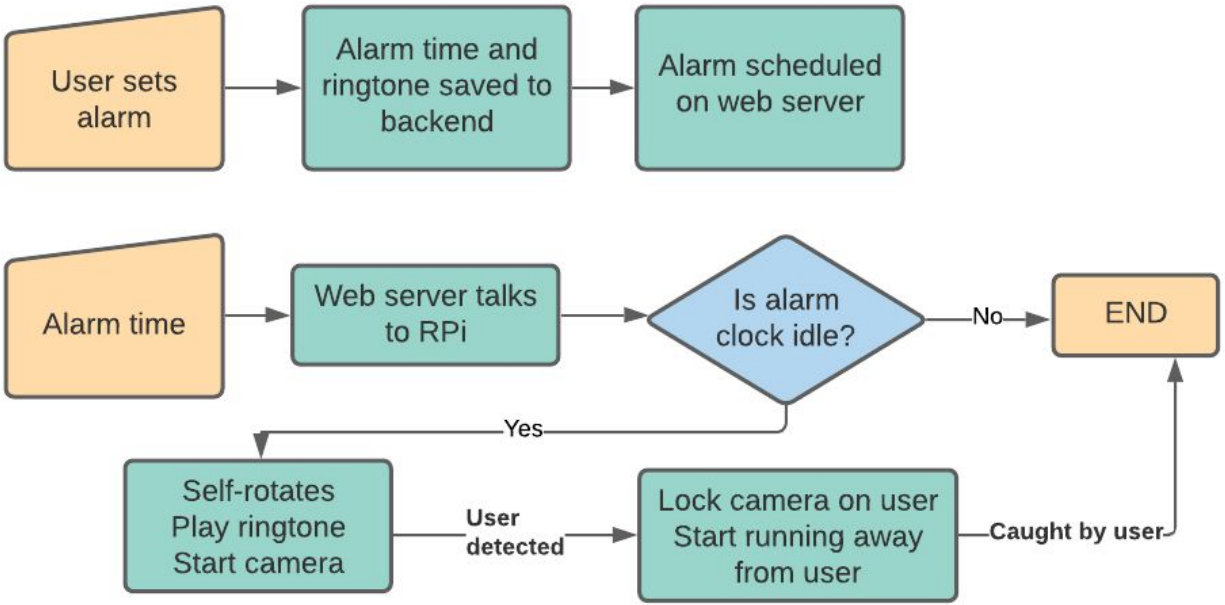
- A no-snooze alarm clock that you have to catch to turn it off
- Traditional/smartphone alarm clock:
  - easy to hit snooze
- “Clock on wheels”:
  - lack reliability
  - lack durability

“Clock on wheels”

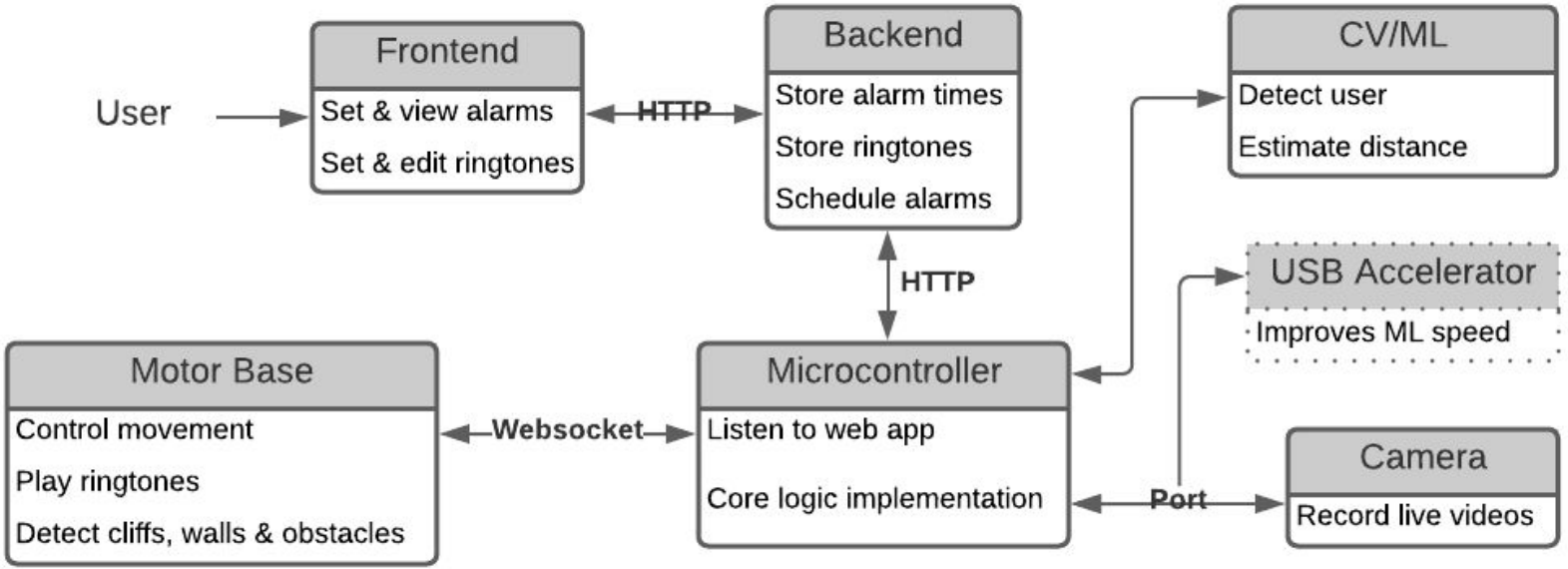
Solution Approach

- CV/ML
  - Human recognition
  - Distance estimation between clock and user
- Create 2 robot base
  - Edge/obstacle detection
  - Create2 Open Interface API
- Web App
  - Interface for users to set alarm time and ringtone

User Flowchart

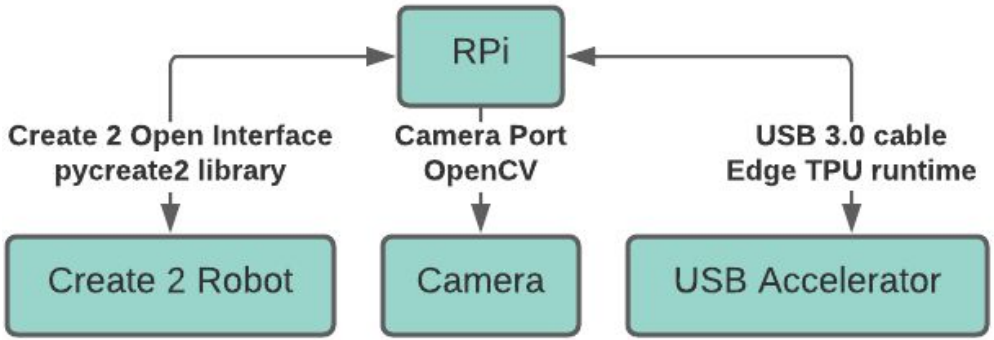


System Diagram

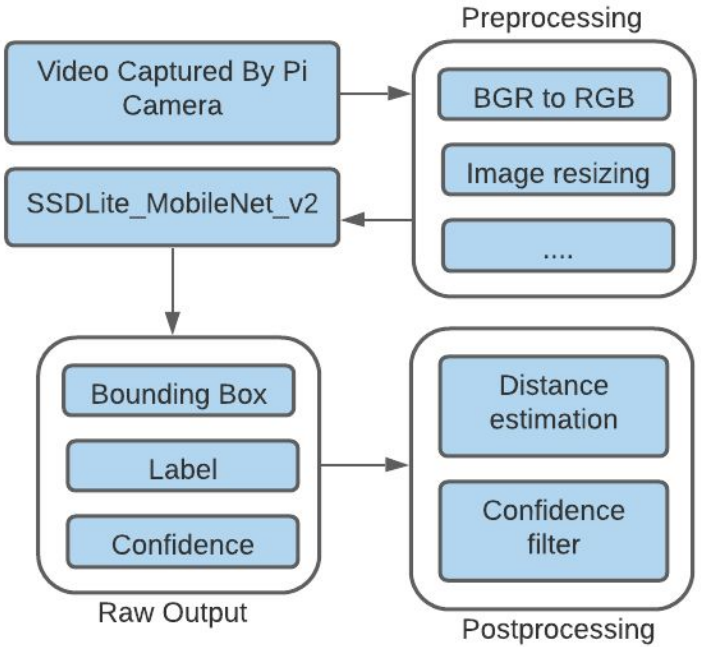


Implementation Plan: hardware setup

- Create 2 Robot
  - cliff detection
  - bump detection
  - song playing
- pycreate2 library
  - compatible w/ Open Interface
- Google Coral USB accelerator
  - Edge TPU coprocessor to speed up ML interference



Implementation Plan: CV/ML pipeline

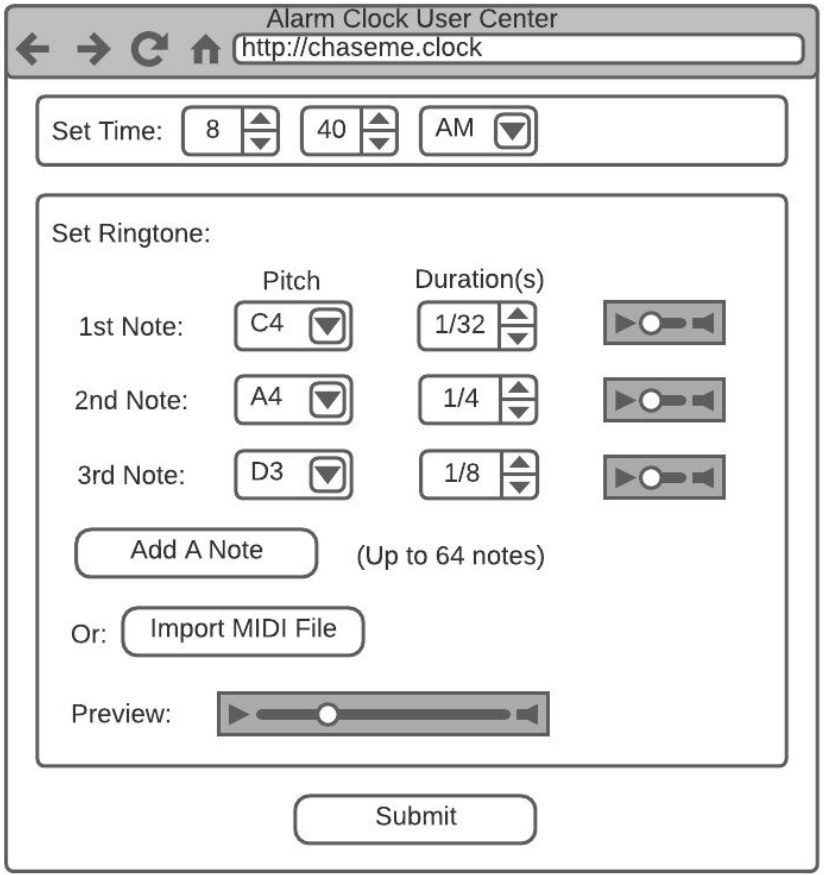


Human Recognition

- TensorFlow Lite
- SSDLite_MobileNet_v2
  - suitable for running on smaller devices
  - available in Python

Implementation Plan: web app

- MERN stack
- sample-player package for playing notes
- AWS EC2 for deployment
- crontab for task scheduling



Validation

Requirement	Metrics
Low latency in communication	delay from web app sends signal, to robot starts working < 1s
	delay from facial recognition to chase starting < 0.25s
Fast image processing	ML pipeline FPS > 10, Latency < 0.1s
Accurate human detection	Human recognition accuracy > 90%, false positive rate < 10%
	Camera can rotate 360 degrees, detect user coming at different directions
Accurate distance estimation	Distance estimation accuracy > 75%, starts chase when user w/i 1m radius
Successful obstacle avoidance	Rate of bumping into wall or obstacles the second time < 20%
Effective wake	chase duration > 30s, chase overall linear distance > 5m

Risk Factors

- To mitigate high latency in human detection
  - Quantize ML models: fp32 -> int8
  - Add heat sinks/fans to reduce overheating
- To improve low accuracy in human detection
  - Add more preprocessing steps, e.g. transformation

* Assumptions
  - Unknown environment
  - Unknown no. of obstacles
  - Only one user involved
  - Enough lighting for CV


week1
week2
week3
week4
week5
week6
week7
week8
week9
week10
week11
week12
week13
week14
week15
Task
8/31
9/7
9/14
9/21
9/28
10/5
10/12
10/19
10/26
11/2
11/9
11/16
11/23
11/30
12/7
Milestone
Abstract
Proposal Presen
1st status report
Design presenta
Design document
Demo in lab
In-lab demo
Final presentatic
Final report
Proposal&Report
Abstract
Proposal
Proposal Presentation
Report
Demo
Final
Computer Vision
Learn Yolo/Realsense & Order related hardware
Test&run on PC (human recognition)
Migrate onto Nano2
Test&Optimize
Fine-tune
Control Flow Diagram
[Insert new rows above this one, then hide or delete this row]
Web Application
User Interaction Diagram
UI Mockup
Object Diagram
Frontend Development
Backend development
Frontend/backend integration + testing
Hosting on AWS cloud
Hardware
Learn I/O of Create2 robot base
Communication between Nano2 and robot base
Test & Sweep parameters
Optimization
Hardware Diagram
[Insert new rows above this one, then hide or delete this row]
Integration
System Diagram
Combine Processing unit with base
Test I
Receive/React to siganal sent from webapp
Test II
Final Test
[Insert new rows above this one, then hide or delete this row]
Whole Team
Page Yu + Echo Gao
Yuhan Xiao
Echo Gao
Page Yu

Hardware CV/ML Web App

Questions?