Week 5

Goals

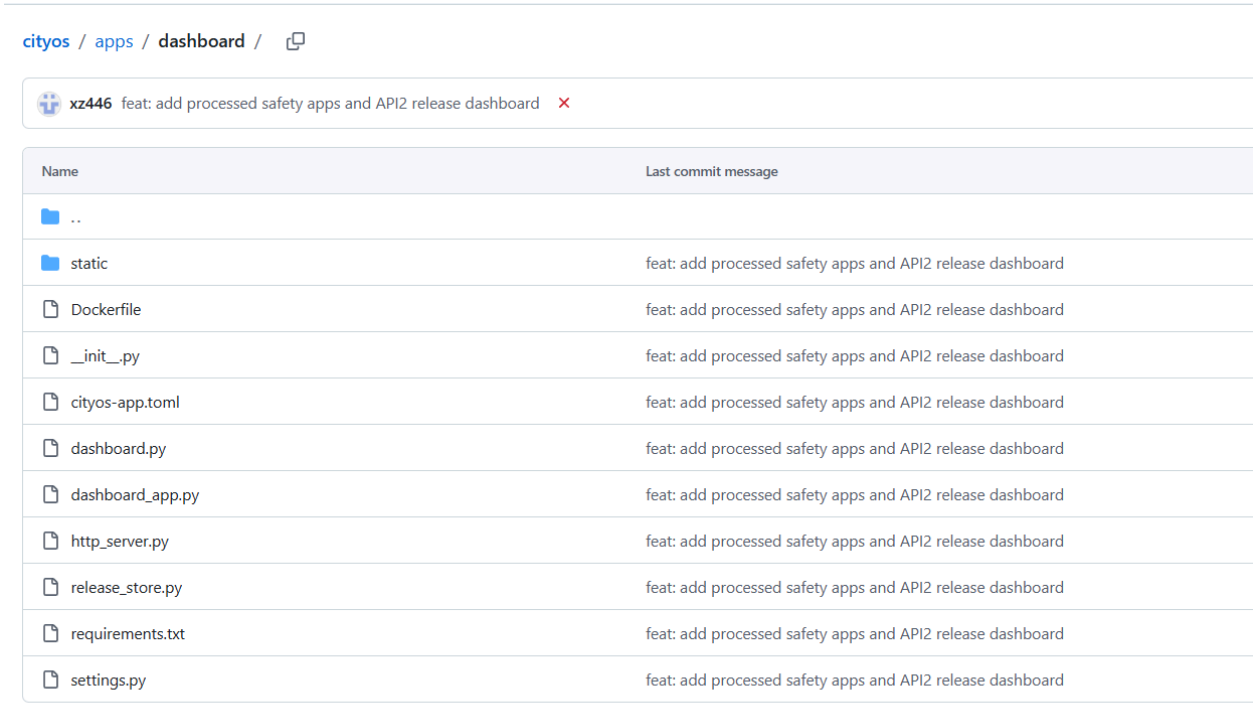
- **Still waiting on DeepSeek API Key**
- **Ran with new version of TraceFix**
- **Established CityOS synthesizer LLM agent**
- **Refined system architecture**

OpenCode Harness

We integrated the OpenCode Harness, a CLI tool that generates code, fixes bugs, and helps Trace-Fix create development plans and artifacts.

CityOS Synthesizer

We also developed the CityOS Synthesizer as a layer before CityOS to understand API 1, 2, and 3 restrictions, deploy agents as Docker containers, monitor runtime activity, and communicate with Trace-Fix and TeLLMe.



cityos / apps / dashboard /

xz446 feat: add processed safety apps and API2 release dashboard

Name	Last commit message
..	
static	feat: add processed safety apps and API2 release dashboard
Dockerfile	feat: add processed safety apps and API2 release dashboard
__init__.py	feat: add processed safety apps and API2 release dashboard
cityos-app.toml	feat: add processed safety apps and API2 release dashboard
dashboard.py	feat: add processed safety apps and API2 release dashboard
dashboard_app.py	feat: add processed safety apps and API2 release dashboard
http_server.py	feat: add processed safety apps and API2 release dashboard
release_store.py	feat: add processed safety apps and API2 release dashboard
requirements.txt	feat: add processed safety apps and API2 release dashboard
settings.py	feat: add processed safety apps and API2 release dashboard

TeLLMe

CityOS Integration

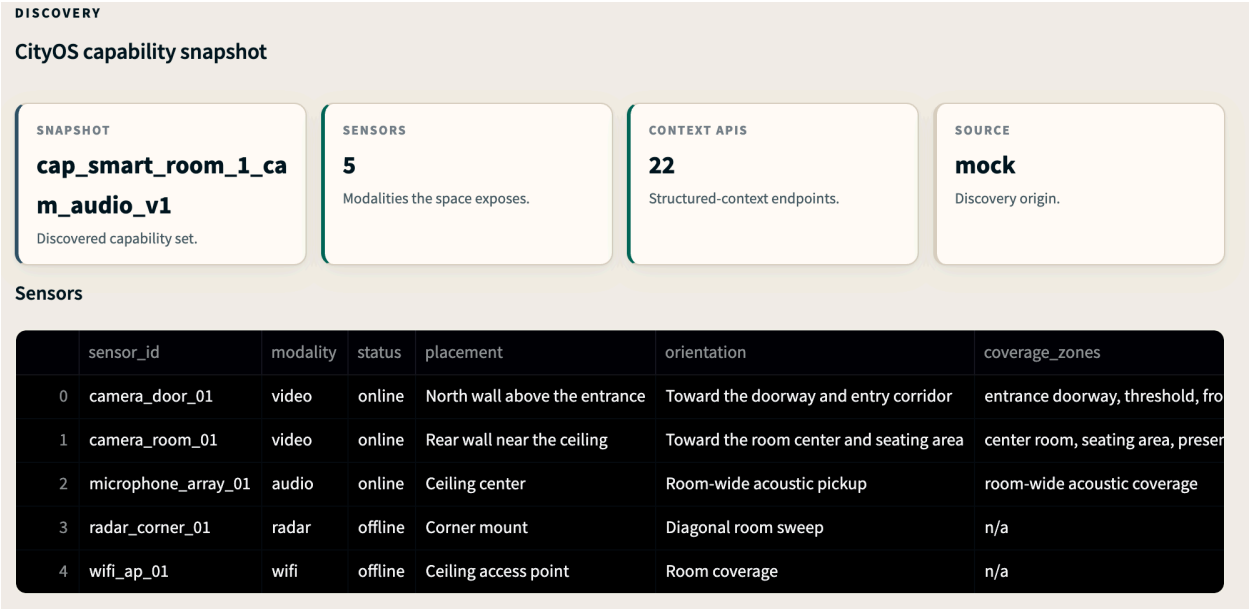
We continued CityOS integration by using it as a privacy layer that adds noise to data while protecting user identities, with agents running as separate applications alongside a runtime monitor that tracks their activity.

Goals:

- Began integrating TeLLMe into the TraceFix pipeline.
- Created a demo tool for the newest version of the stack.
- Enabled TeLLMe to generate capability-aware plans.

TeLLMe Integration

We connected TeLLMe to the updated TraceFix pipeline and created a demo tool for the new stack. TeLLMe can now generate plans based on the system capabilities available in CityOS.



	sensor_id	modality	status	placement	orientation	coverage_zones
0	camera_door_01	video	online	North wall above the entrance	Toward the doorway and entry corridor	entrance doorway, threshold, fro
1	camera_room_01	video	online	Rear wall near the ceiling	Toward the room center and seating area	center room, seating area, preser
2	microphone_array_01	audio	online	Ceiling center	Room-wide acoustic pickup	room-wide acoustic coverage
3	radar_corner_01	radar	offline	Corner mount	Diagonal room sweep	n/a
4	wifi_ap_01	wifi	offline	Ceiling access point	Room coverage	n/a

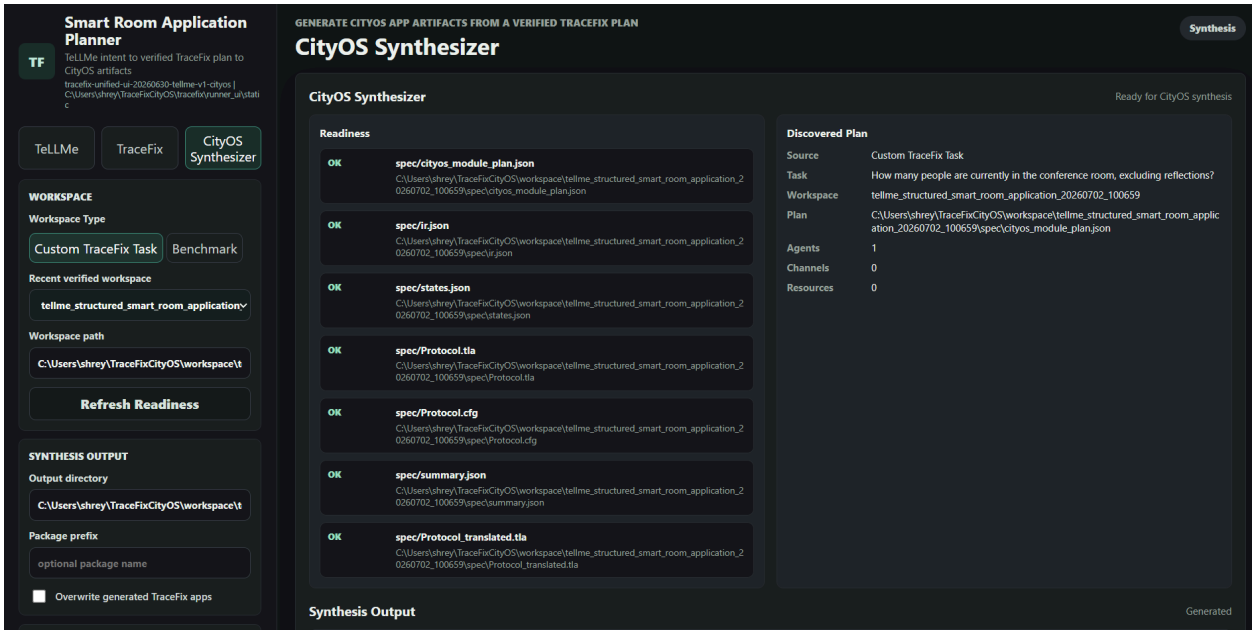
Week 6-7

Details

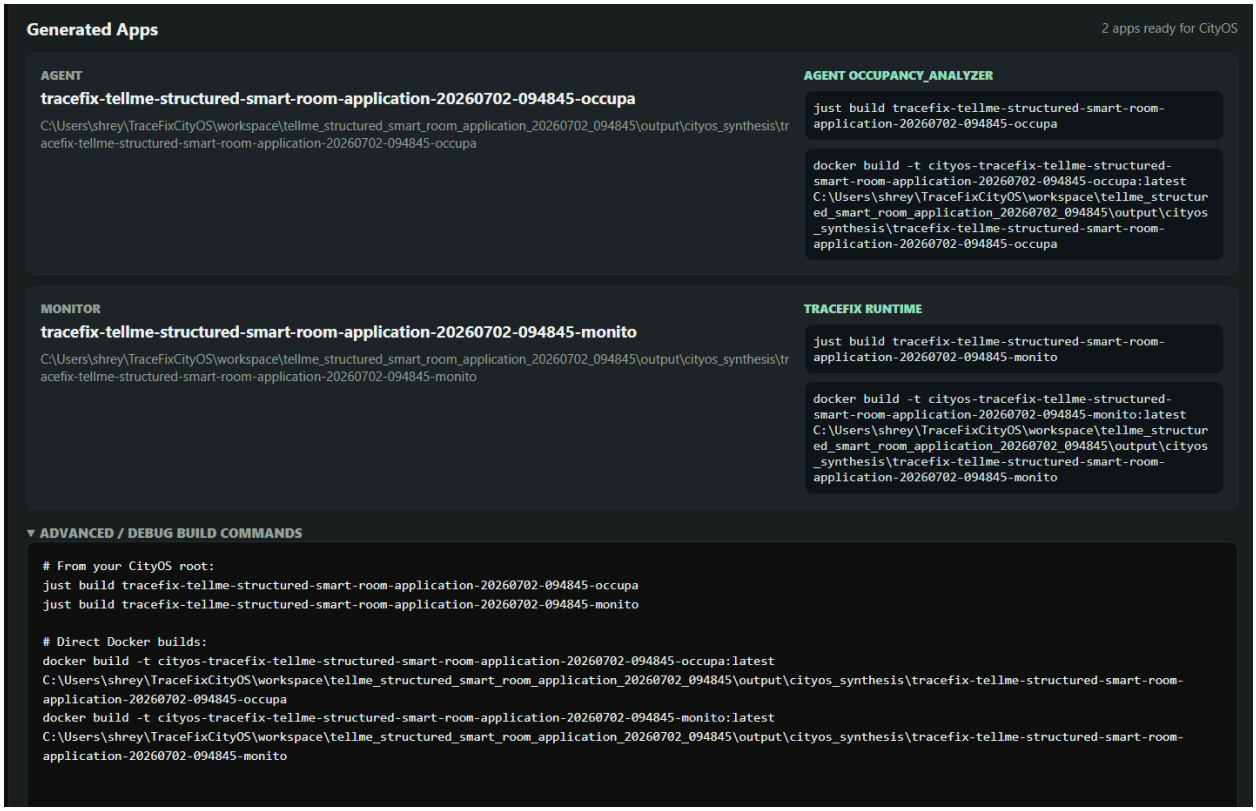
Goals

- Api Key Received: OpenRouter; Using glm 5.2
- TeLLMe–TraceFix–CityOS Synthesizer pipeline completed
- Revised pipeline optimized for time
- Implemented web server pipeline

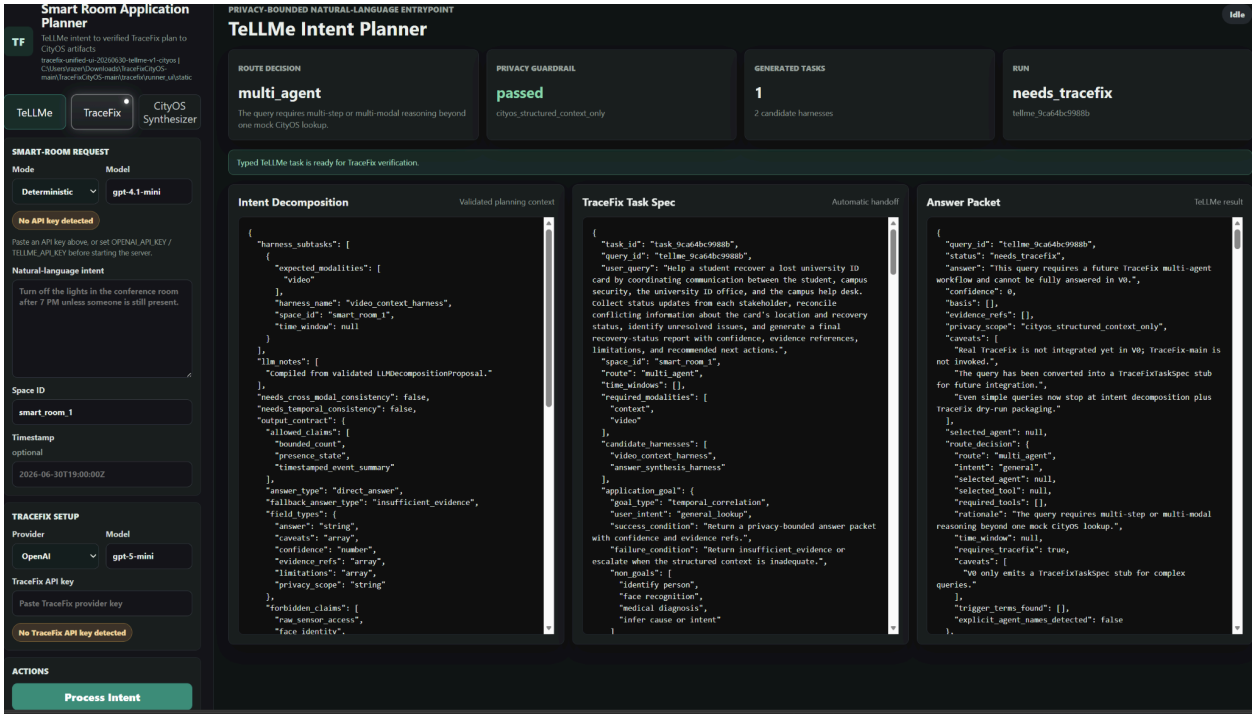
Completed CityOS Synthesizer



Completed Docker Commands



Completed Pipeline



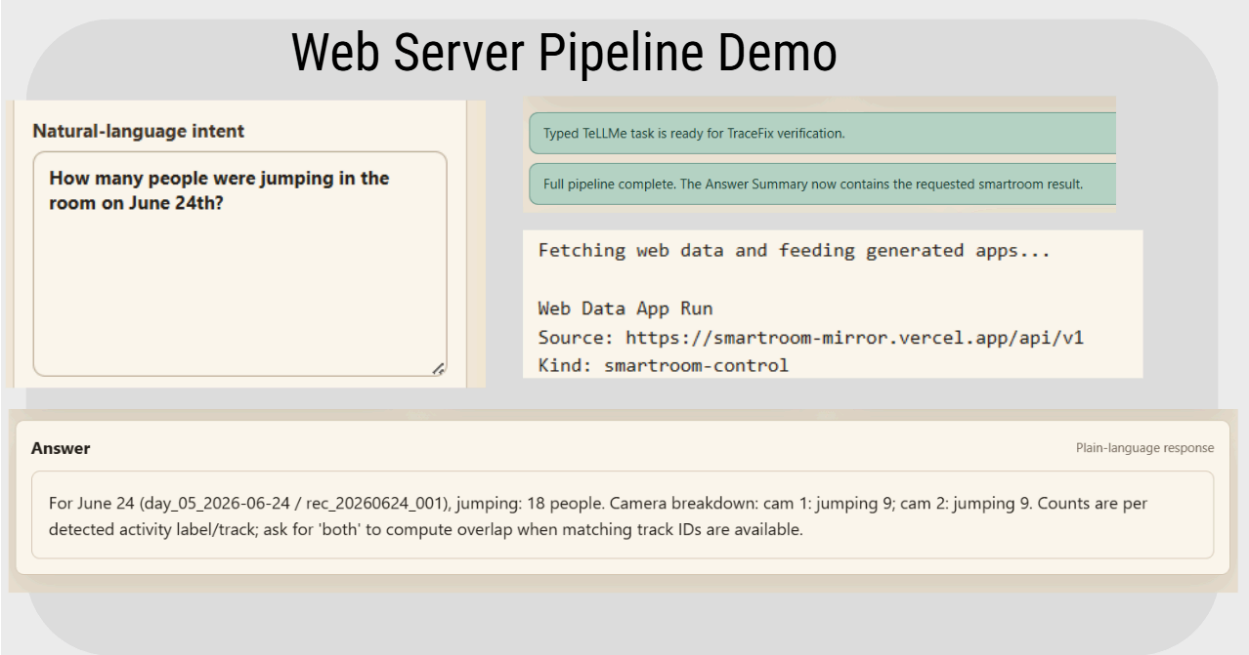
OpenCode Issues

We also improved the OpenCode Harness by using template-based protocols, reducing runtime from about 20 minutes to 20–30 seconds while continuing to verify all protocols with TLC before synthesis into CityOS apps.

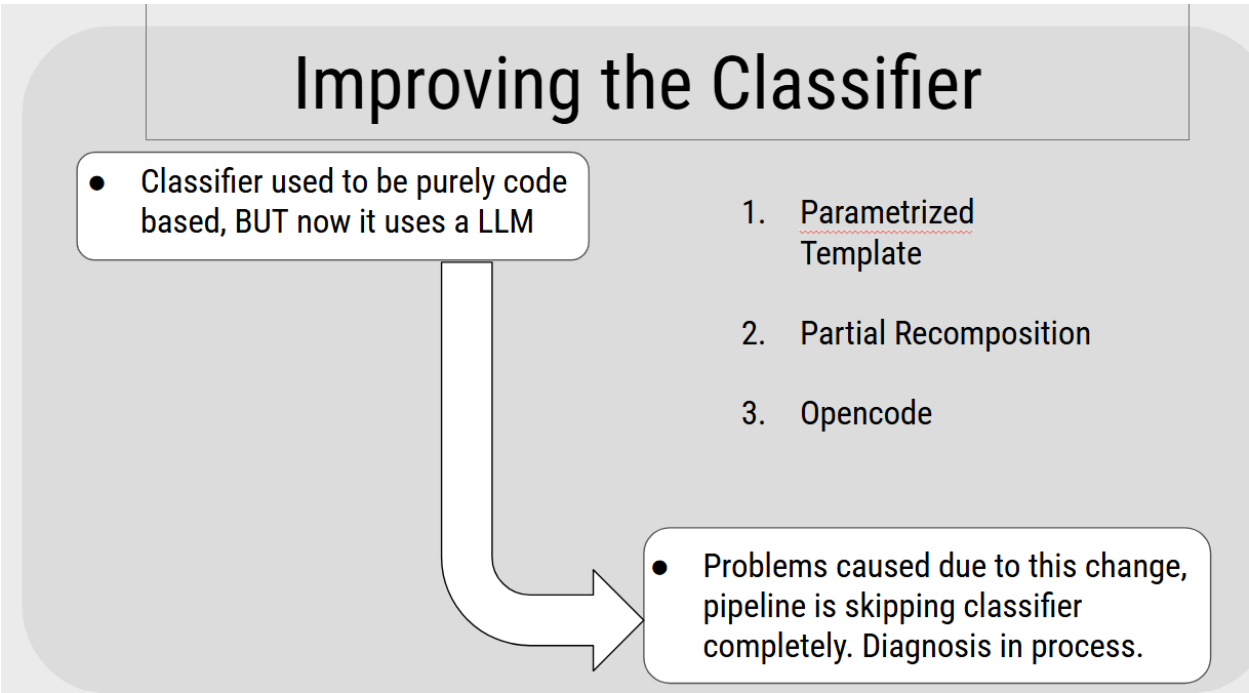
API Endpoints

We implemented the web server pipeline by creating an API endpoint that stores smart room data and allows the web-harness agent to retrieve information and generate payloads for the agents. The agents process the input frame to produce responses, and we added a human tracking ID to the API for improved tracking. We also demonstrated the web server pipeline and continued development of the classifier.

Web Server Demo



Worked on Classifier



TeLLMe

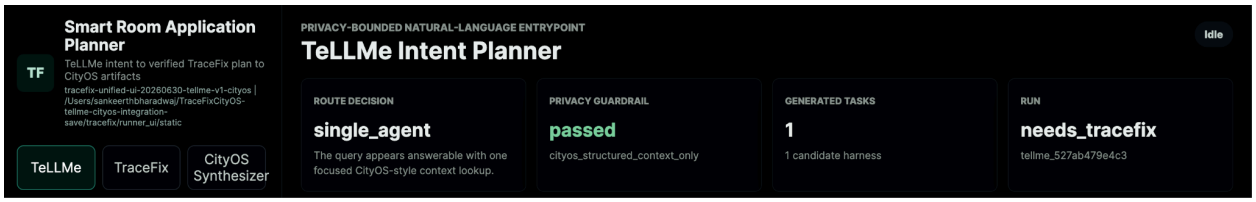
Week 6:

Goals

- Completed TeLLMe's integration with the pipeline.
- Improved how TeLLMe manages context and system constraints.
- Began working on expanding its context window and adding CityOS tools.

TeLLMe Integration

We completed TeLLMe's pipeline integration. TeLLMe now manages the user's context and constraints, while TraceFix handles agent design, verification, and execution.



Week 7:

Goals

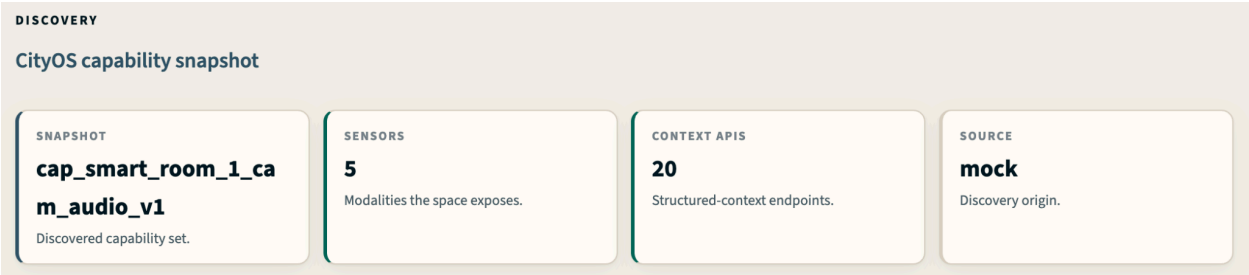
- Shortened the TaskSpec sent to TraceFix.
- Continued developing the CityOS Discovery Tool.
- Improved how TeLLMe receives available CityOS capabilities.

TaskSpec Reduction

The previous TraceFix prompt was too long, so we identified the key components and shortened the TaskSpec while keeping the information needed to process the user's request.

CityOS Discovery Tool

We worked on the CityOS sensor discovery tool. A capability only reaches TeLLMe when it is registered, available, permitted by privacy settings, and located in the requested space.



Week 8

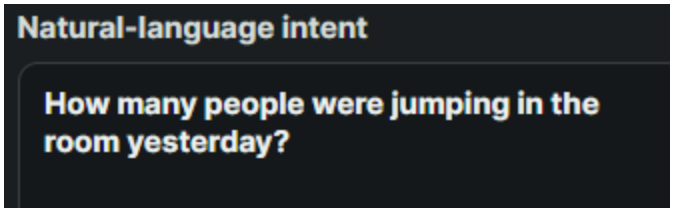
Goals

- Continued testing the web server endpoint and found more problems to work on for the rest of the weeks:

Problems Found

- **Specific Dates**
  - Multiple Date Issue
  - Relative Keywords (Yesterday, Last week, etc)
- **Overlap Calculations**
  - # of people
  - # of actions
- **Word Issues**
  - Put on a hat vs. wearing a hat

Example Output




```
Recording: day_13_2026-07-15 /
rec_20260715_003
Selection: selected newest recording because
the question did not include a specific date
Cameras: cam2-d455, cam2-d435
Recordings available: 5
```

Implemented OCSORT/ReID

What is it?

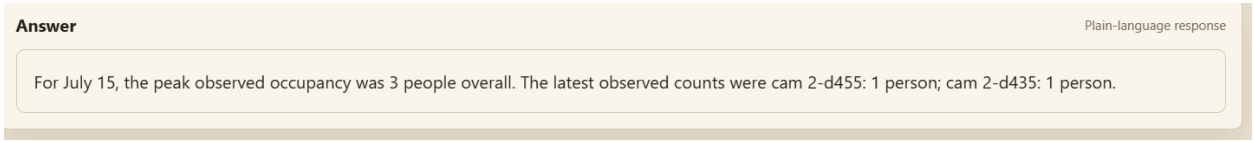
- A visual tracking algorithm
- Detects objects in video frames
- Runs on top of YOLO

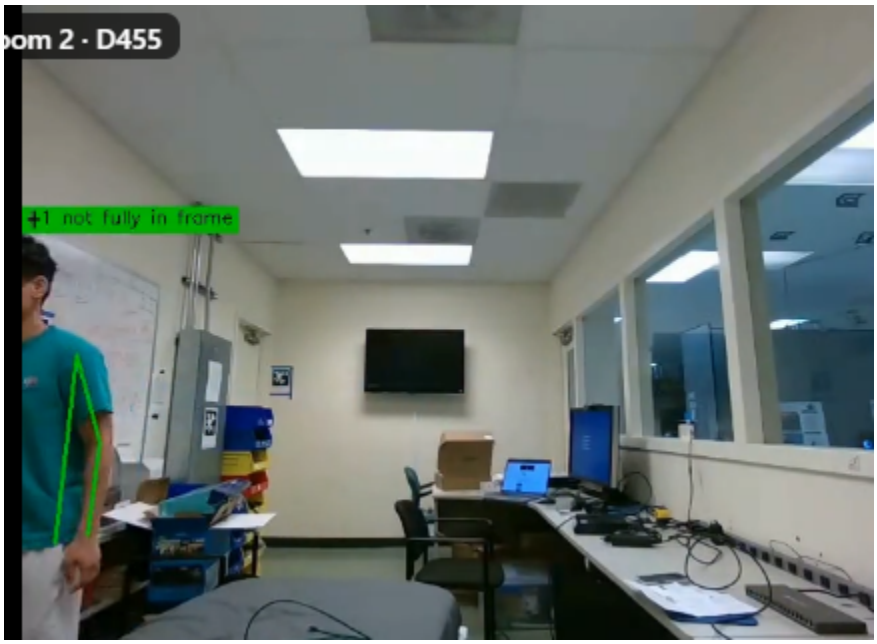
Why are we using it?

- It helps keep track of objects when they leave and re-enter the frame
- Helps counts objects
- Gives each object its own ID.

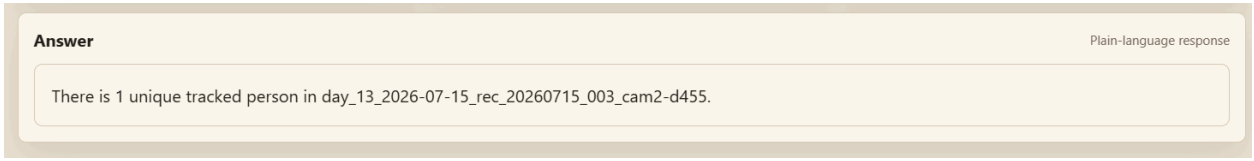
Example Output

Before: Not Implemented OCSORT





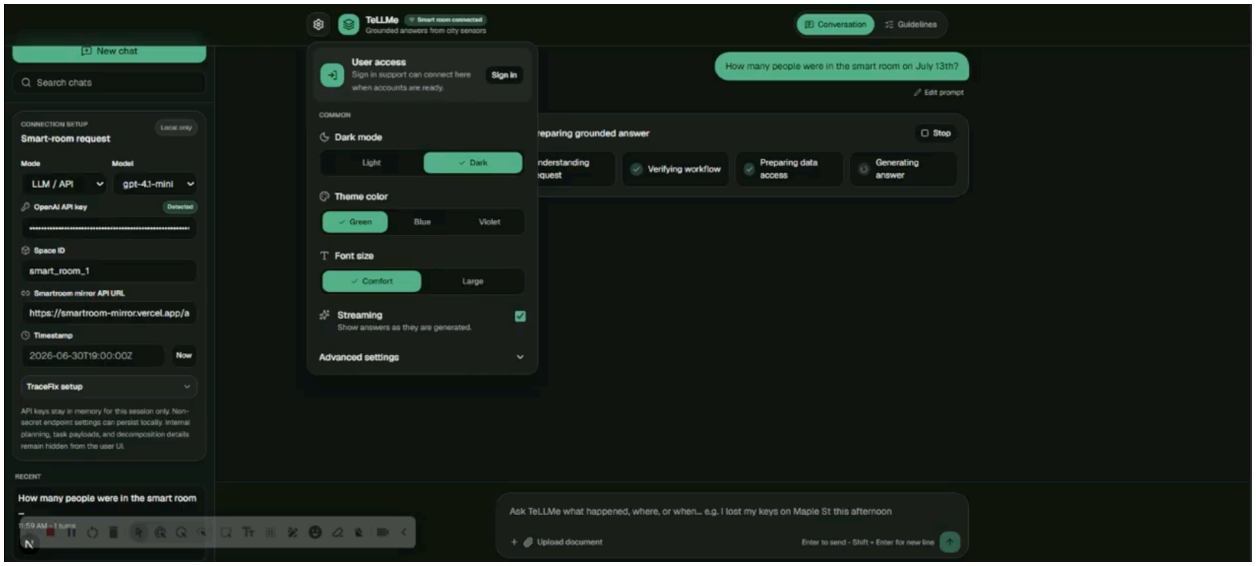
After: Implemented OCSORT



TeLLMe

Goals

- Added deterministic checkpoints throughout the TeLLMe pipeline.
- Built a bridge between TeLLMe and CityOS.
- Researched how TraceFix output could be converted into deployable CityOS applications.
- Created a new and improved TeLLMe user interface.



TeLLMe Pipeline Improvements

We added deterministic checkpoints to validate outputs at different stages of the TeLLMe pipeline and built a demo connection to CityOS. We also researched an outlet that converts verified TraceFix plans into CityOS Docker applications and translates CityOS Servicer state machines into Python code. In addition, we developed an improved TeLLMe interface for submitting questions and viewing evidence-backed answers.

