MILESTONE 3

T-Mobile CareLink Project

Hypertension Medication Tracking

https://github.com/Xiao-Hannah/Tmobile-Carelink-Project.git
https://github.com/urnotvicky-li/CareLink-Portal.git
https://adorable-brigadeiros-c2240e.netlify.app/

GIX Student Team 7
University of Washington

Chang Li, Hannah Xiao, Marjorie Yang, Yishuai Zheng

UNIVERSITY of WASHINGTON | GLOBAL INNOVATION EXCHANGE

Problem Statement

Globally, hypertension affects over 1 billion people, yet poor medication adherence (30-40%) remains a major barrier to effective treatment, leading to serious health risks and increased healthcare costs. The most prevalent reasons include: forgetfulness, failure to perceive the need for medication, fear of side effects and low self-efficacy - placing additional stress on caregivers, especially for elderly patients.

Despite the scale of the problem, patients lack simple, connected tools to track medication, monitor blood pressure, and stay engaged with their care plans.

There is a clear need for an integrated hardware and software solution that leverages T-Mobile's connectivity to enable real-time tracking, reminders, and data sharing - empowering patients, reducing caregiver burden and improving health outcomes.

Solution Description

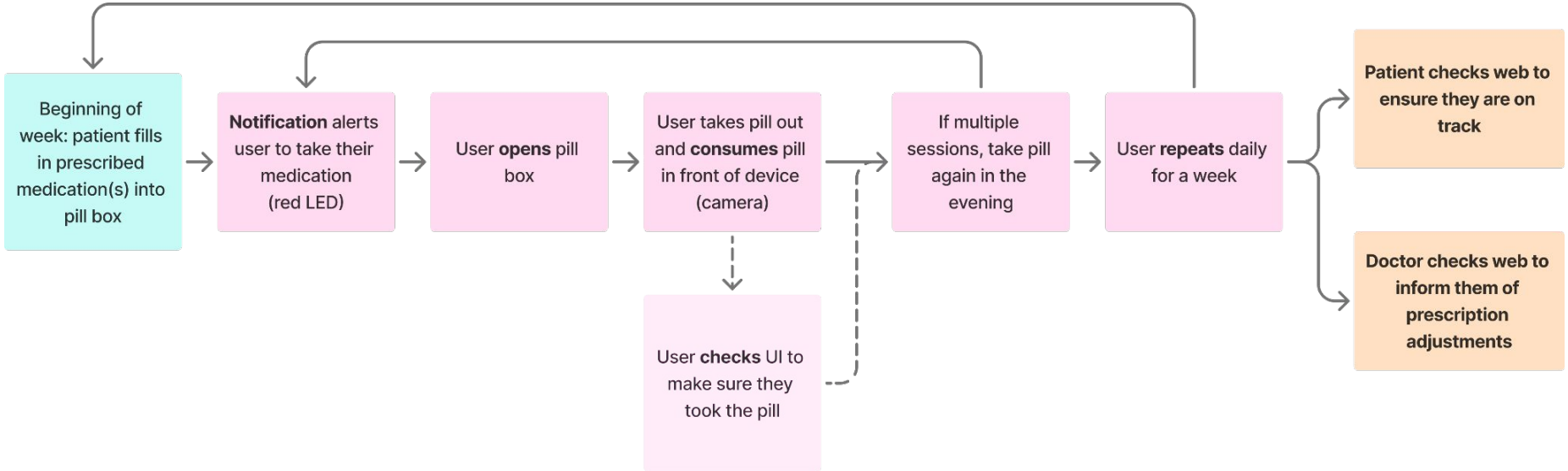
We are building a two-part system to track medication adherence and support hypertension care:

1. **Detecting Pill Removal by Photoresistors**
    - Embedded under each separated box of the pill box, when the pills are removed, the photoresistor will return a lower value as more light is absorbed.
2. **Verifying Pill Consumption with Camera**
    - A camera tracks gesture to monitors the consumption.
    - ML image detection ensures that ingestion (not just removal) is verified.

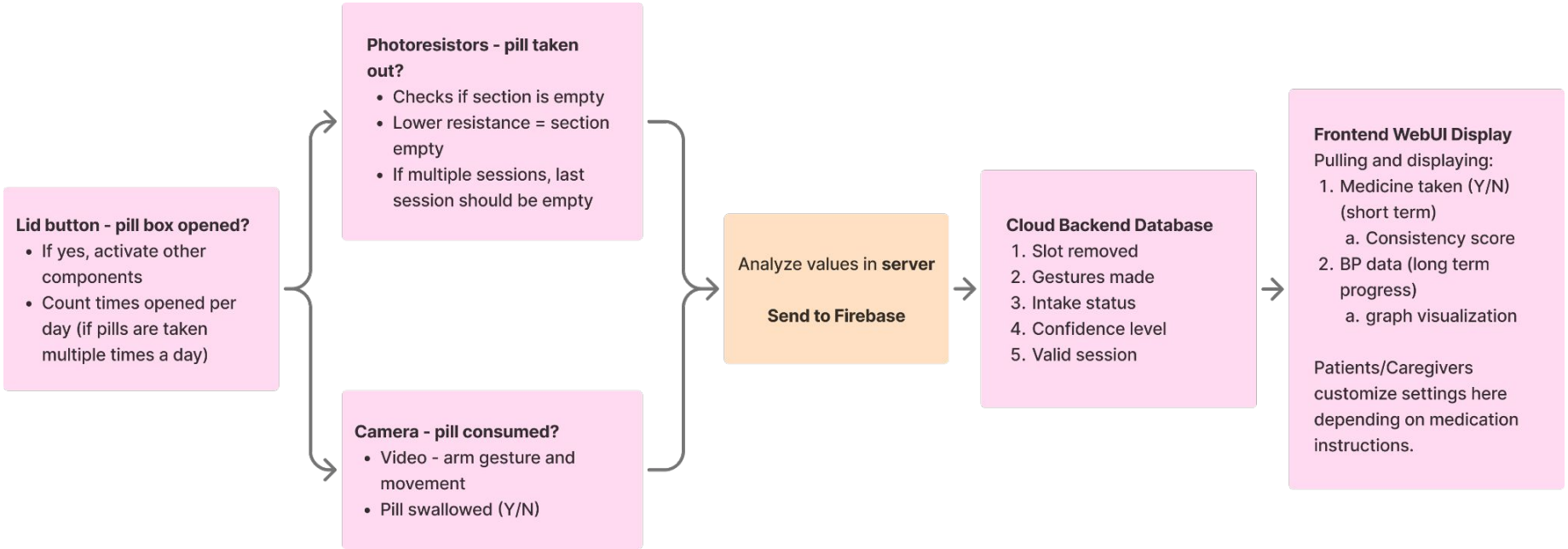
Patient & Doctor portals (health data tracking): All data is synced to a companion portal, allowing both patients and doctors to track medication adherence over time

T-mobile 5G Connection: The device connects to the cloud through T-Mobile's 5G network, which supports real-time data upload and remote monitoring without depending on local Wi-Fi.

User Flow Diagram



Data Pipeline



Enclosure Design

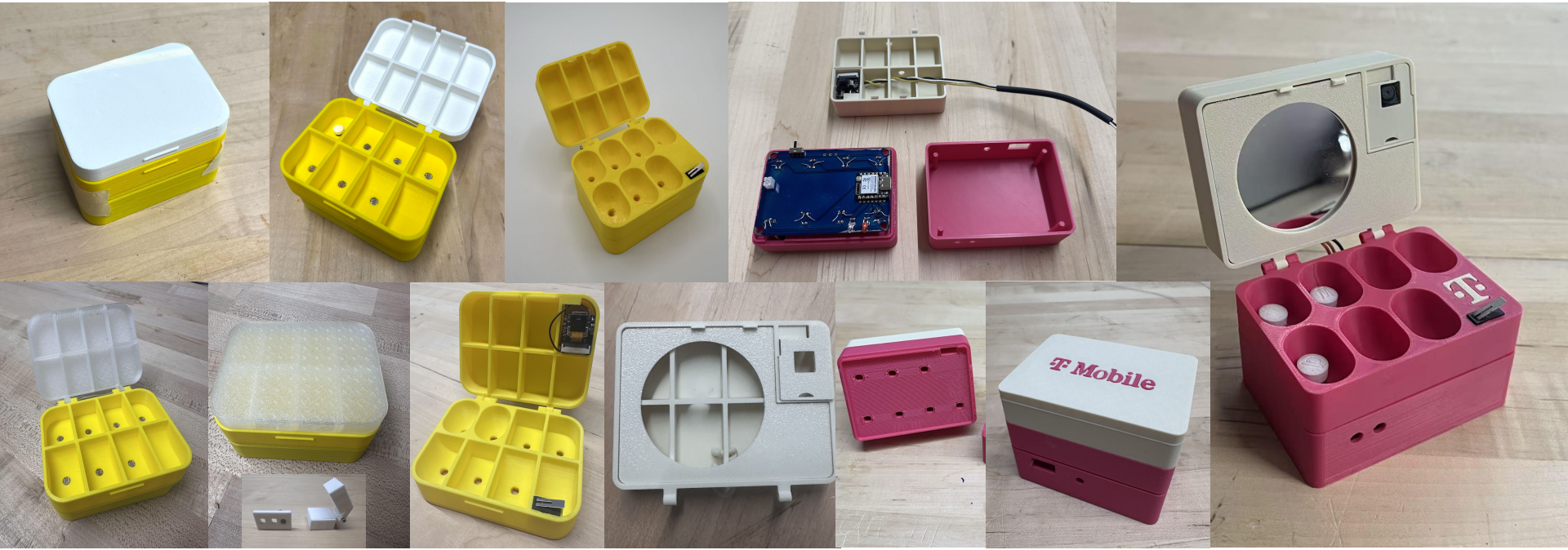
We designed a weekly-use pill box with a compact, square-shaped form, making it suitable for daily use at home or easy to carry on the go.

The system is activated when the lid is opened. Inside, there are seven compartments to hold a week’s medication. Each compartment is equipped with a photoresistor at the bottom to detect whether the pill has been taken.

On the top-right corner of the lid, a small camera is embedded to verify whether the user has actually swallowed the pill. A mirror is positioned next to the camera to help users align themselves properly within the frame.

At the bottom of the device, two LED indicators provide visual feedback: a red LED reminds users to take their medication on time; a white LEDs to notify if all pills are removed or the week is over.

Enclosure Design - Iterations



Prototype Components: Photoresistors

Our system uses an ESP32 microcontroller connected to **seven photoresistors (LDRs)**, each positioned under an individual pill slot.

Data Flow:
ESP32 Sensors → EMA Filtering → Flask Server → Firebase

Hardware Components:

- 7 LDR (Light Dependent Resistor) sensors for pill detection
- Button control (GPIO44) for system on/off
- WiFi connectivity for cloud communication

Smart Detection System:

- Exponential Moving Average (EMA) filtering for stable readings
- Configurable thresholds per sensor slot

Prototype Components: Camera

The camera component is directly connected to the PCB for power + button signal.

Releasing the button activates the camera to collect 50 frames (10 FPS) after a 3 second delay. These 50 frames are sent to the flask server to be analyzed.

Data Flow:
ESP32 Cam → Flask Server → Firebase

Gesture Detection:

- On the flask server, we use MediaPipe to detect poses.
- These are then analyzed to determine whether gesture has been made.
- **Savitzky-Golay low-pass filter** is applied for cleaner signals.

Data Sent to Firestore

```python
# Compute intake status
if pills_removed is None:
    intake_status = "Not enough data"
    confidence = "Low"

elif pills_removed > 0:
    intake_status = "Taken"
    confidence = "High"

elif pills_removed == 0 and session_valid:
    intake_status = "Uncertain"
    confidence = "Medium"

elif pills_removed == 0 and not session_valid:
    intake_status = "Not Taken"
    confidence = "High"

else:
    intake_status = "Unknown"
    confidence = "Low"
```

confidence: "High"

gesture_count: 1

gesture_detected: true

gesture_timestamp: 1748917053

intake_status: "Taken"

last_updated: June 2, 2025 at 7:17:47 PM UTC-7

ldr_timestamp: 1748917053

person_in_frame: true

pills_present: 4

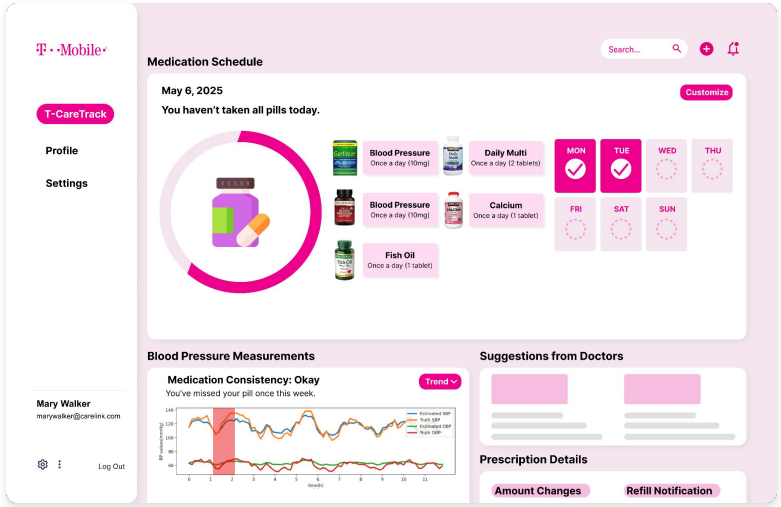
pills_removed: 3

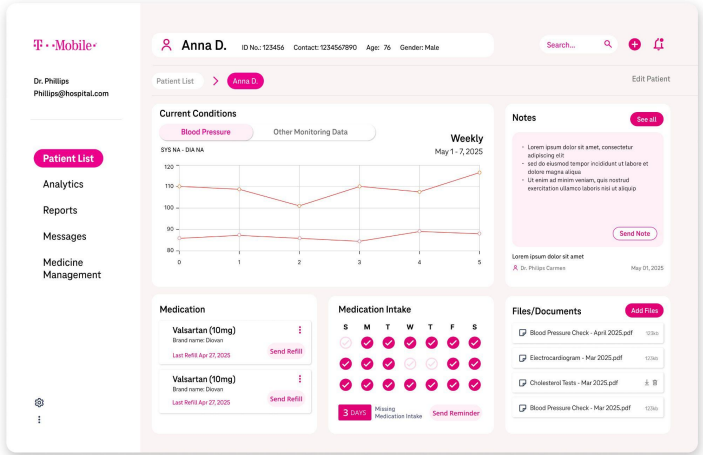
removed_slots

0 2

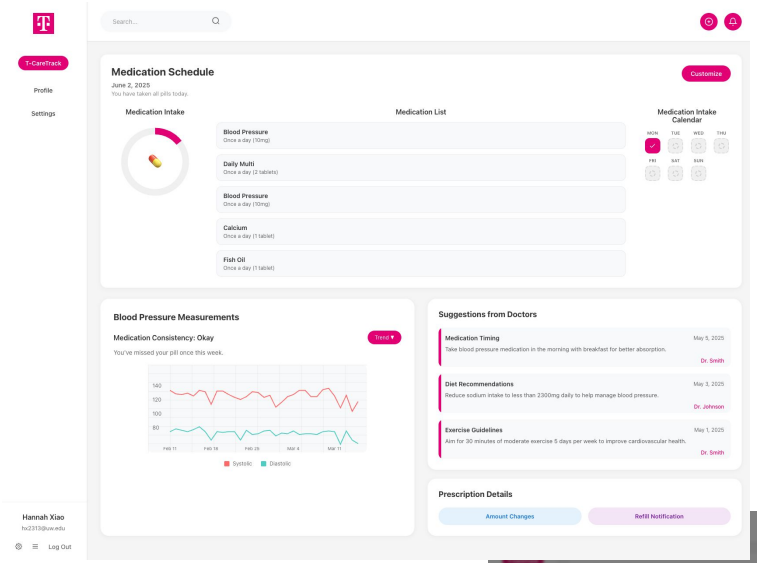
UI Design

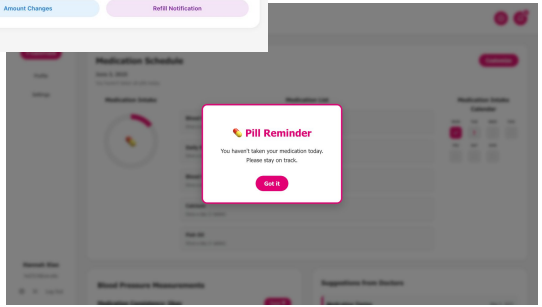
1. **Medication Schedule**
    a. Accounts for types of medications, frequency, and when they need to be taken.
    b. “Customize” button - for if the prescription changes.
2. **Blood pressure measurements**
3. **Hypertension progress**
    a. Medication consistency status
    b. Missed medication highlighted over BP trend visualization
    c. Tab to toggle week, month, year
4. **Notification from Caregiver**
5. **Suggestions**
6. **Medication refill buttons for notification**
7. **Notes from doctors**
8. **Patients’ documents**





Patient Portal





Framework & Routing:

- React application with React Router for navigation
- Uses Context API (UserContext) to manage user roles (like doctor or patient) and login state across the app without repetitive prop drilling
- Connects to Firebase for user authentication and dynamic health data retrieval

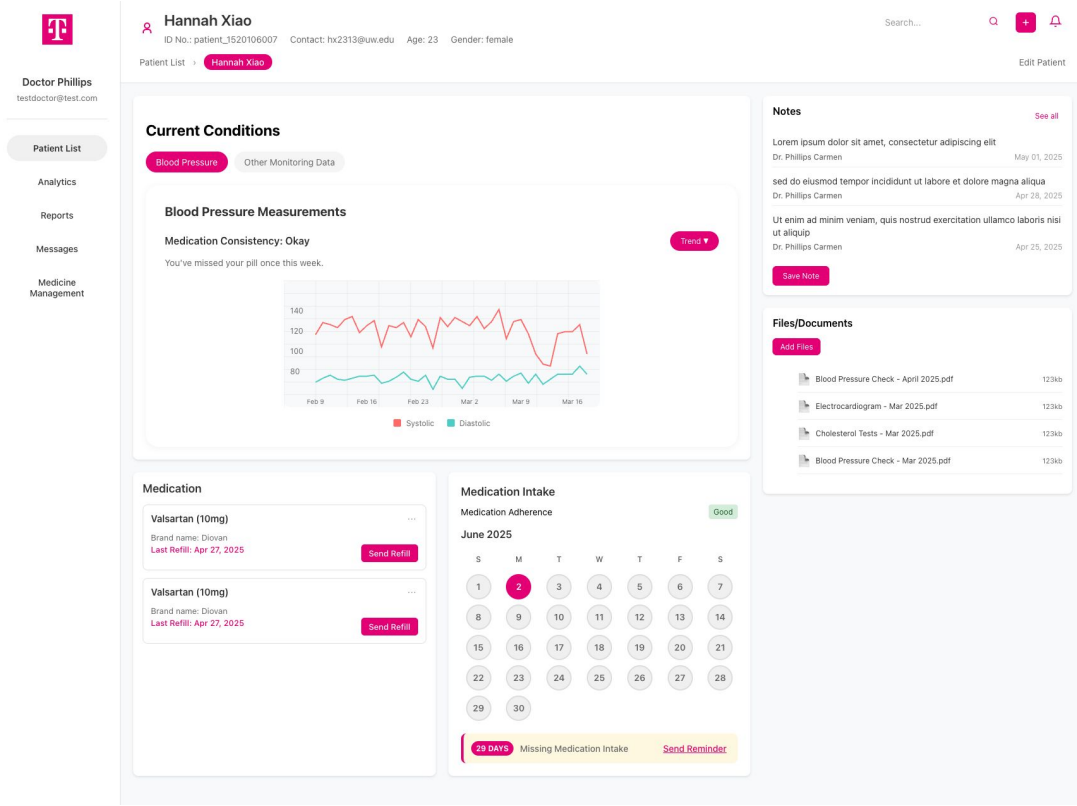
Navigation & Layout:

- Fixed sidebar with T-Mobile branding
- Search functionality
- User profile section
- Settings and logout options

Styling:

- Comprehensive CSS with T-Mobile Magenta color
- Modern UI elements (circular progress, cards, hover effects, trend charts)

Doctor Portal



Architecture & Framework

- React-based SPA with functional components and hooks
- React Router for navigation between different portal sections
- Context API (`UserContext`) for user state management

T-Mobile Styling Architecture

- CSS Grid for complex layouts (content areas, calendar)
- Flexbox for component alignment and spacing
- CSS transitions for smooth hover effects
- Design system follows T-Mobile's brand palette, using magenta highlights and light UI surfaces

Reflections

Hardware:
Incorrect voltage divider wiring issues on PCB, power bypassed photoresistors and directly went to resistors.
Solution: redesigning the schematic for proper voltage division between power and ground connecting to photoresistors and resistors, enabling successful operation with battery power.

Software:
Integrated real-time data from two devices (pill detection & gesture camera) into one Firebase.
Solution: solved by using **Flask server** to integrate two data streams into a unified **Firestore database** for a user, enabling a complete medication adherence tracking system.

Frontend UI:
Hard to coordinate a shared platform across teams due to three separate Firebase backends with large amount of data and difficult cross-navigation.
Solution: keep individual backends and route users based on credentials.

UNIVERSITY of WASHINGTON | GLOBAL INNOVATION EXCHANGE

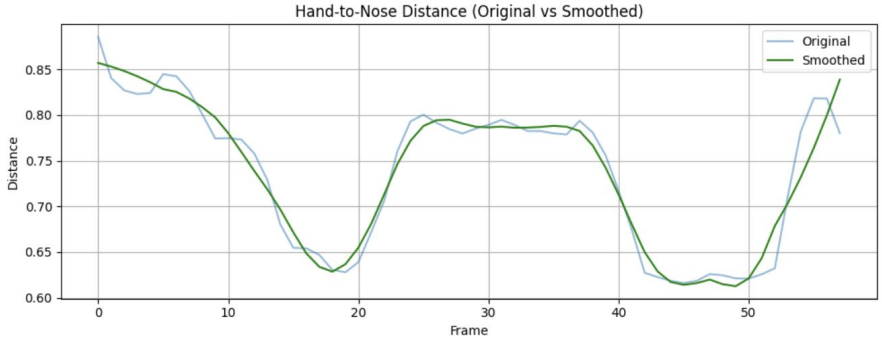
Appendix: Gesture Tracking

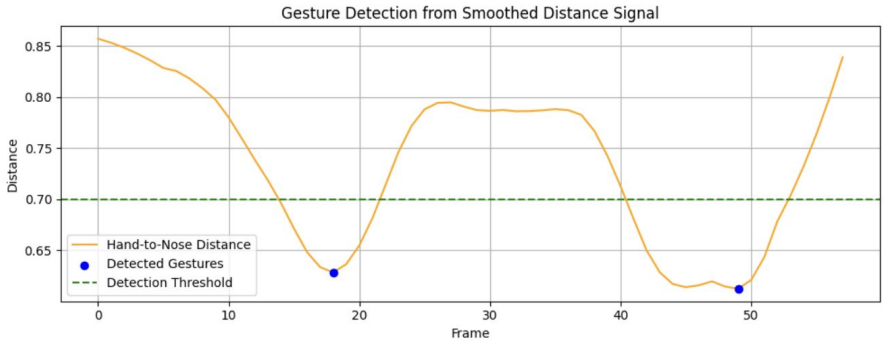
The gesture detection works by tracking the distance between the hand and the nose (not mouth as it might be easily covered while eating the pill). These body parts are detected using MediaPipe, where the center of each part is tracked as a point.

By tracking this Euclidean distance of these two points across the frames, we are able see the dips where the distance is the shortest.

DSP:

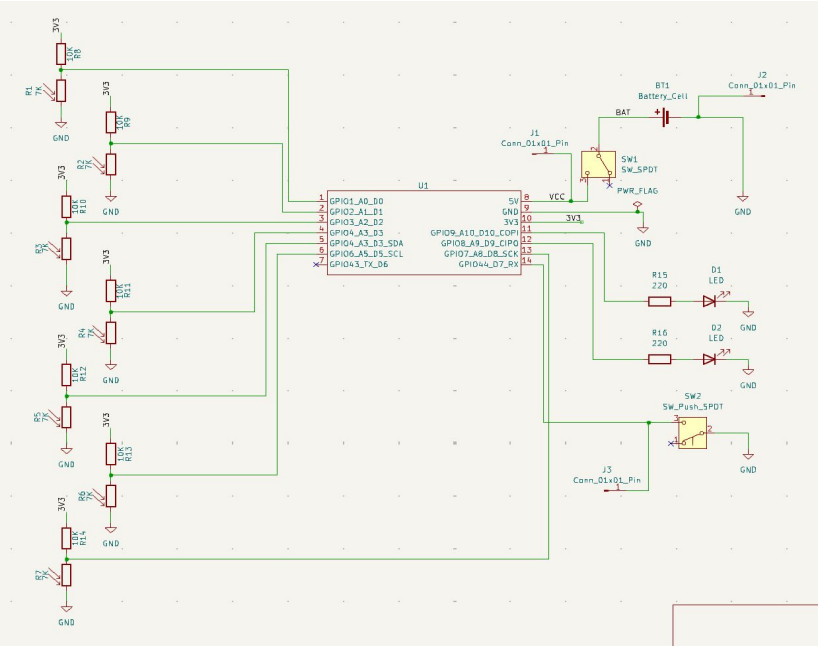
- **Savitzky-Golay low-pass filter:** reduces noise while preserving the overall gesture shape.
- **find_peaks() from SciPy:** detects gestures by identifying the dips.
- **find_peaks() constraints:** ‘distance = 10’ ignores dips less than 10 frames apart and ‘height = -0.7’ only detects peaks under 0.7.

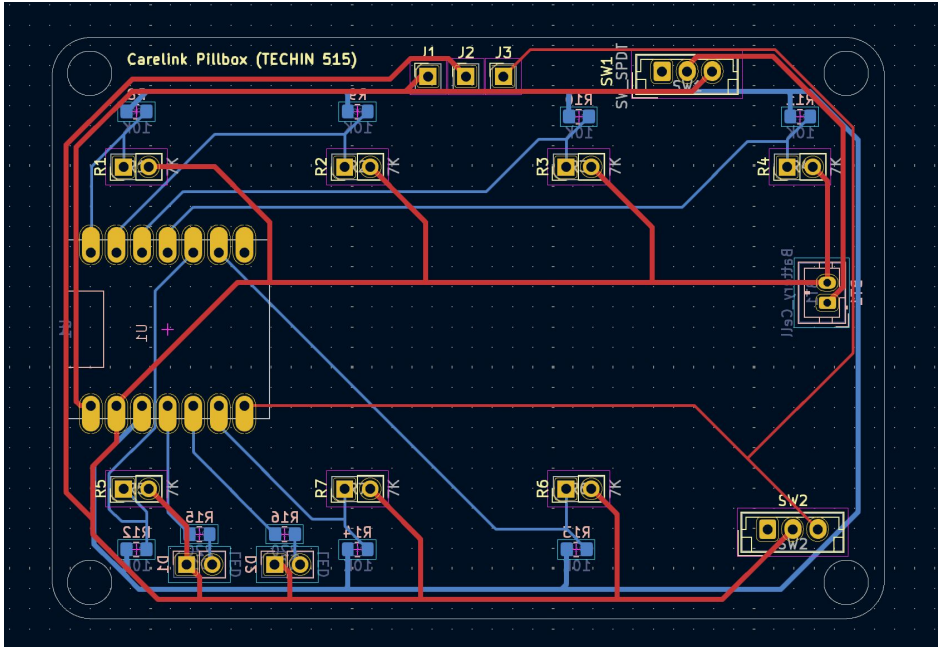





```
Detected gestures: 2
Gesture start frames: [18 49]
```

Appendix: Final PCB Design





Budget

Month	Item	Notes	Link to the item	Unit Price	QUANTITY	Actual Expenses	Status	Note
April	Single Point Load Cell- 100g	Non-Amazon	https://www.phid	$2.00	2	$29.06	Arrived	Note
April	Seeed Xiao ESP32C3	Non-Amazon	https://www.digik	$7.00	2	$9.98	Arrived	Note
April	HiLetgo Photodiode	Amazon	https://www.ama	$5.78	1	$7.06	Arrived	Note
April	NOYITO ADS1232	Amazon	https://www.ama	$14.99	1	$16.27	Arrived	Note
April	HiLetgo ADS1256	Amazon	https://www.ama	$16.98	1	$18.26	Arrived	Note
April	SparkFun Load Cell Amplifier	Returned	https://www.ama	$10.95	1	$12.07		Note
April	Xiao ESP32S3	Amazon	https://www.ama	$29.15	1	$29.15	Arrived	Note
May	Xiao ESP32S3 Camera	Amazon	https://www.ama	$25.91	1	$25.91	Arrived	Note
May	Lithium Ion Battery - 3.7V 200	Non-Amazon	https://www.adaf	$12.50	1	$20.17	Arrived	Note
May	SUNLU 3D Printer Filament PL	Amazon	https://www.ama	$18.98	1	$24.21	Arrived	Note
May	SUNLU PLA 3D Printer Filame	Amazon	https://www.ama	$18.98	1	$21.68	Arrived	Note
May	Flexible Adhesive Mirror Shee	Amazon	https://www.ama	$7.49	1	$7.49	Arrived	Note
May	EEMB Lithium Polymer Batter	Amazon	https://www.ama	$12.66	2	$27.90	Arrived	Note
May	Super Z Outlet Mini Round 2" I	Amazon	https://www.ama	$11.75	1	$12.95	Not yet	Note
						$262.16		

Remaining Budget: $87.84 *[350 - $262.16]*

Link to up-to-date team budget

Readme File

Milestone 3

In this final milestone, we focused on integrating both photoresistor and camera subsystems into a unified medication tracking platform, supporting real-time doctor-patient interactions.

Full System Integration

We deployed two complementary sensing modalities:

- Photoresistors (LDRs) detect physical removal of pills from each compartment.
- Camera Module captures and verifies pill-taking gestures via MediaPipe (hand-to-nose movement).

Data from both sensors are timestamped and sent to a cloud-based FastAPI/Flask backend. If both LDR and gesture detection occur withi a defined time window, the system confidently marks the pill as "taken." Otherwise, it flags the action as "Not Taken" on both patient and doctor portal, and the LED light on the pill box will flash red to notify. If either the photoresistor or the camera fails to detect correctly, the system marks the status as "Uncertain," which there is a chance to allow the patinet/caregive to manually confirm medication intake status on the patient portal. After a 7-day period, the LED on the pillbox will blink white to remind the patient/caregiver to refill the pillbox.

Deployment

- All components are housed in a custom PCB and a 3D-printed enclosure, powered via battery.
- The system (The camera and the photo resistors) is automatically activated when the pillbox is opened, triggered by an embedded switch mechanism placed inside the bottom left part of the pill box.
- The device connects to the cloud through T-Mobile's 5G network, which supports real-time data upload and remote monitoring witho depending on local Wi-Fi.
- Backend supports real-time Firebase integration, enabling potential EHR sync in the future.

Frontend Visualization

We interated on and finalized the UI design for both patient and doctor dashboards and implemented responsive dashboards using React, HTML, CSS, and JavaScript within the Cursor development environment to visualize and receiving real-time data collection from the pillbo We also integrated our portals with two other teams sponsored by the T-Mobile CareLink+ project, allowing both patients and doctors to sign in through one unified platform.

Key features include:

Daily Pill Intake Tracking:

- Displays pill status by time and compartment using a simple, color-coded calendar view.

Key features include:

Daily Pill Intake Tracking:

- Displays pill status by time and compartment using a simple, color-coded calendar view.

Weekly Adherence Summaries:

- Visualizes weekly medication patterns using backend-generated data and intuitive charts.

Blood Pressure Summary:

- Provides a quick overview of the patient's recent blood pressure readings for contextual health insights.

Medication Adherence Reminders:

- Automatically sends reminders to patients based on missed doses or irregular intake patterns.

Doctor-to-Patient Messaging:

- Allows doctors to send personalized notes or feedback directly to patients, which can be reviewed in the

Hardware -> Firebase

For hardware, ensure you connect the two components from Milestone 2 so that the LDR component is conn Additionally, power the two components with a shared 3.7 / 2000 mAh battery and switch. To get this running

Server Side Steps

1. Go to your Firebase, and download the credentials.
2. Navigate to `server` folder from terminal.
3. Add the credentials JSON file to the `server` folder.
4. Make sure you are replacing the file path for the credentials in the `flask_server.py` code.
5. Set up a virtual environment with the following sequence (note - please ensure python 3.10 has been ins some packages required)

```
rm -rf venv
python3.10 -m venv venv
source venv/bin/activate
pip install --no-deps --no-cache-dir -r requirements.txt
```

6. Run `flask_server.py`
7. Turn on the switch and open the lid when the red light turns on!

T-Mobile Carelink WebApp

This is a React-based web application designed for the T-Mobile Carelink platform. It features separate portals for patients and doctors to manage and track medication adherence securely.

Follow these steps to run the project locally.

1. Clone the repository

```
git clone git@github.com:urnotvicky-li/T-mobile-carelink-doctorportal.git
cd T-mobile-carelink-doctorportal
```

2. Install Dependencies

```
npm install
```

3. Set Up Firebase Environment Variables

```
Create a new file named .env in the root directory: touch .env
```

Then add the following lines to the .env file, replacing the values with your Firebase project credentials:

REACT_APP_API_KEY=your-api-key

REACT_APP_AUTH_DOMAIN=your-app.firebaseapp.com

REACT_APP_PROJECT_ID=your-project-id

REACT_APP_STORAGE_BUCKET=your-app.appspot.com

REACT_APP_MESSAGING_SENDER_ID=your-sender-id

REACT_APP_APP_ID=your-app-id

You can find these values in your Firebase Console:

- Go to your project