PillPopperPro

A0: MM Demangone, Aneesha Bhattacharjee, Taylor Koda
18-500 Capstone Design, Spring 2025
Electrical and Computer Engineering Department
Carnegie Mellon University

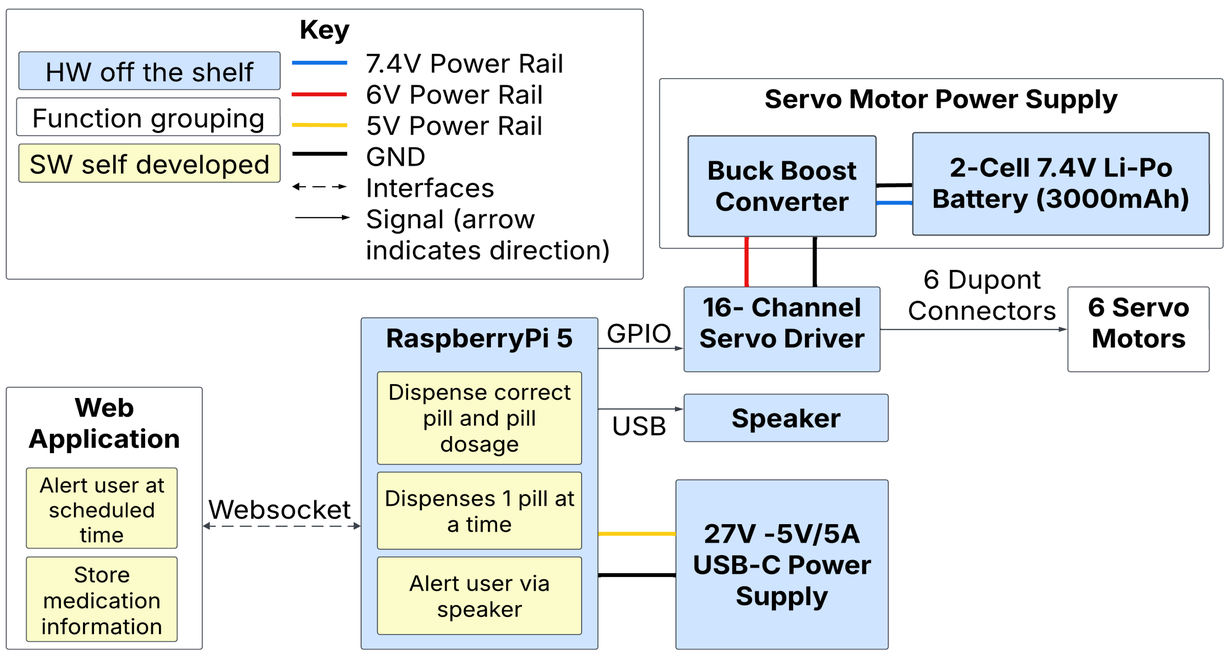
Product Pitch

Have you ever struggled managing your prescriptions - whether that is forgetting doses, taking the wrong dose, or having a difficult time opening pill bottles? If you said yes to any of these, then PillPopperPro is the solution for you! PillPopperPro automates the pill-dispensing process with a machine and integrated web application. PillPopperPro accurately dispenses up to 6 medications of different sizes and dimensions, at least 95% of the time. It dispenses the pill within 6 seconds of pressing the “Release” pill button. Check out www.pillpopperpro.com to see how easy the web app is to use!

Application Areas: Web-App Development, Embedded Systems, Robotics, Control Systems, and Signal Processing

System Architecture

Users can manage their account, store pill information, notify caretakers, and dispense pills all through PillPopperPro’s web app. When a user adds medication into PillPopperPro, they choose which compartment the pill will be stored in. For the sake of simplicity, let’s say the user stores *Prescription X* into *Compartment A.* When the user clicks the “dispense” button in the web app, the web app will communicate to the RaspberryPi 5 (RPi5) via web sockets. Specifically, the web app will notify the RPi5 to release a pill from *Compartment A*. The RPi5 signals for *Servo Motor A* (corresponding to *Compartment A)* to rotate, causing the pill to dispense. When a pill is dispensed, or a refill is required, the RPi5 will signal to the speaker to play the appropriate audio file. There are 2 external power supplies in our system: one for the 6 servo motors and another for the RPi5. The servo driver allows the servo motors to receive adequate power while receiving instructions from the RPi5



Conclusions & Additional Information

We have successfully met our aspiration: dispensing a pill with the click of a button in a web app! Review our status reports throughout the semester and our web app via the QR codes below. Throughout the course of this semester, three main lessons stuck out to us: 1.) 3D printing the housing is unrealistic due to the sizing dimension constraints of CMU’s Fab Lab 2.) trading out the RPi5 with an older RPi model, as the load cell modules (HX711) are not supported on the RPi5 3.) solid objects often get jammed when dispersed via funnel.

Status Reports
https://course.ece.cmu.edu/~ece500/projects/s25-teama0/

Web App
www.pillpopperpro.com

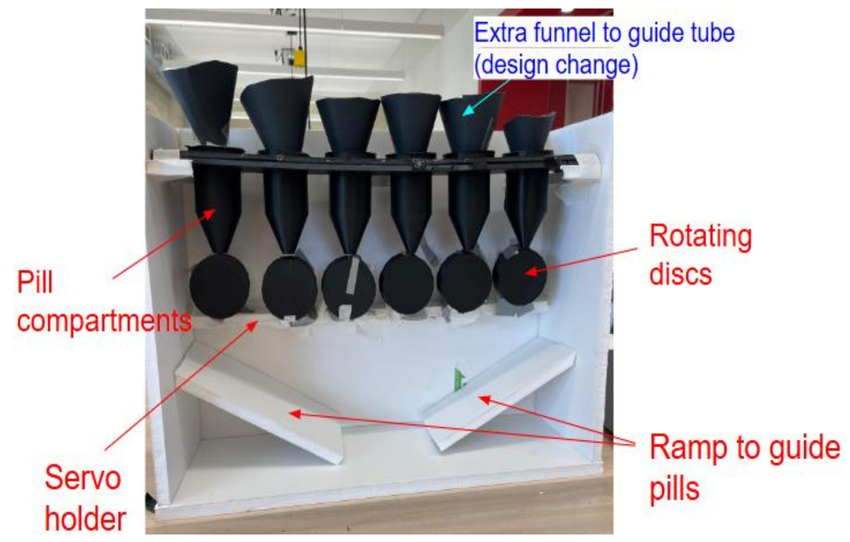
Besides buying an older RPi, we would advise a team that would like to continue this project, to look for a bigger 3D printing site or swap the current housing for a medical-grade, product-safe material.

System Description

Housing: The housing unit has 6 pill compartments, capable of dispensing the following diameters: 8mm, 7mm, 5.5mm, 4.5mm. The compartments are 3D printed. They funnel down into a rotating disk, connected to a servo motor, with two slots 180 degrees apart. The pill will fall into the slot, and as the servo motor rotates, fall onto dispensing area’s load cell.

Hardware: The RPi5 runs the hardware system: it plays .wav files via the speaker and controls servo movement via the 16-channel driver. The RPi5 runs on CMU Device WIFI, and you can SSH into the RPi5 accordingly. The RPi5 runs on a 64GB SDcard. For power supply, we used a 2-cell 7.4V LiPo Batteries (3000mAh) with a buck boost converters, as well as a standard 27W -5V/5A power supply.

Software: The web app domain is allocated via GoDaddy. HTML, JS, Django, and CSS provide functionality and page structure. mySQL and OAuth store user’s accounts and pill information. The Google Calendar API allows users to add pill reminder notifications directly to their calendar. Celery, celery-beat, and Django’s mail module are used to send the user emails daily. It uses the following deployment and SSL softwares: AWS EC2, daphne, nginx, certbot. Daphne, nginx, redis, celery, celery-beat, and redis are all daemonized programs.



System Evaluation

Each compartment dispenses the correct medication ≥95% of the time. We refined our design as follows: 1.) adding a tube to feed pills into the disc one at a time, as the funnel design caused pills to jam 2.) adding inserts into the rotating discs’ slots to ensure only one pill fell into the slot 3.) removing the load cell and its amplifier as readings were too noisy 4.) replacing 3D-printed body with a foam body, as the body’s dimensions exceed CMU FabLab’s limit. We rendered our web app across 6 common screen sizes, ensuring all components were accessible with a min. text size of 14px. We created and tested 3 unique user accounts to ensure each user could only access their information, not another’s information, and the information stored was accurate. All 3 users were accurately notified of pill refills and times to take the medication at the correct time zone. We refined our design as follows: 1.) adding email and Google Calendar API, as notifications only appeared in browser 2.) replacing Bluetooth with web sockets as some browsers don’t support Bluetooth 3.) adding SSL certificate so the web app was accessible across all browsers 4.) adding celery and celery beat to automate daily notifications.

Metric	Target	Actual
Correct pill dispensed	≥ 95%	≥ 96.67%
Correct dose dispensed	≥ 95%	≥ 96.67%
No. of compartments	6	6
Pills per compartment	< 30	< 20-60
Notification delay from scheduled time	< 60 sec.	< 3 sec.
Dispensing delay from user activation	<10 sec.	5.84 s
Pill dimensions	1–22mm	4-23mm

Electrical & Computer ENGINEERING
Carnegie Mellon University