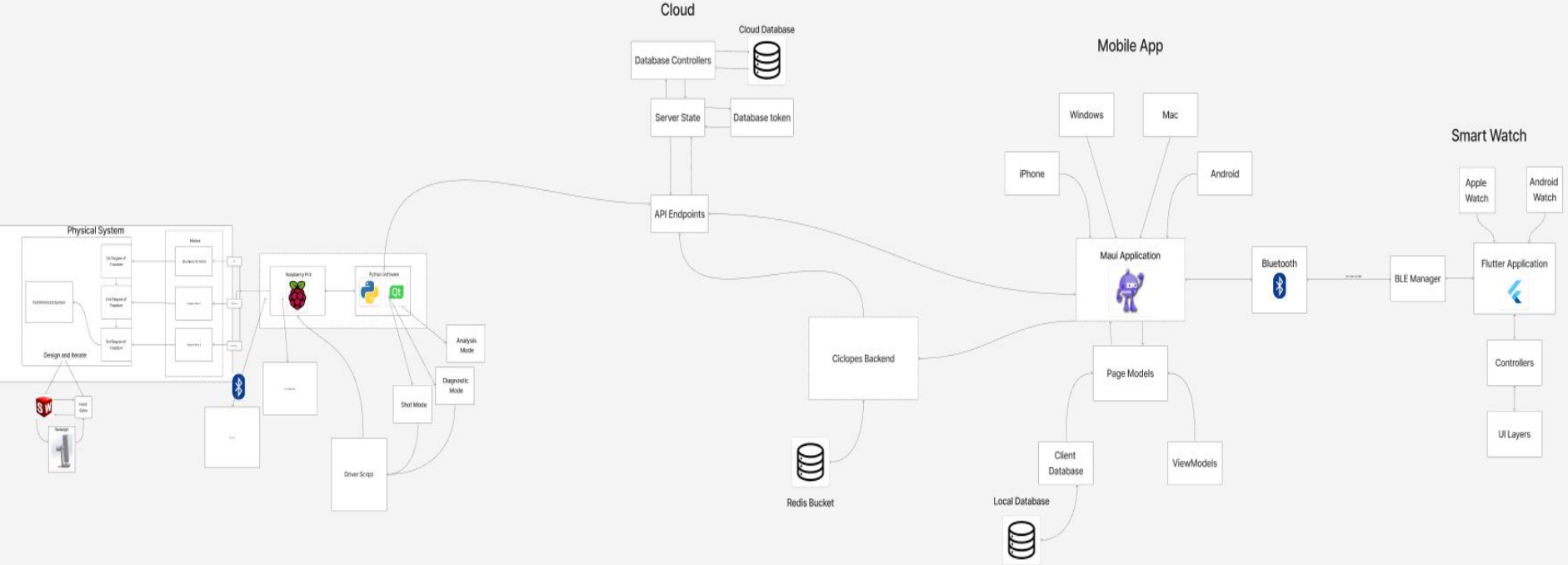
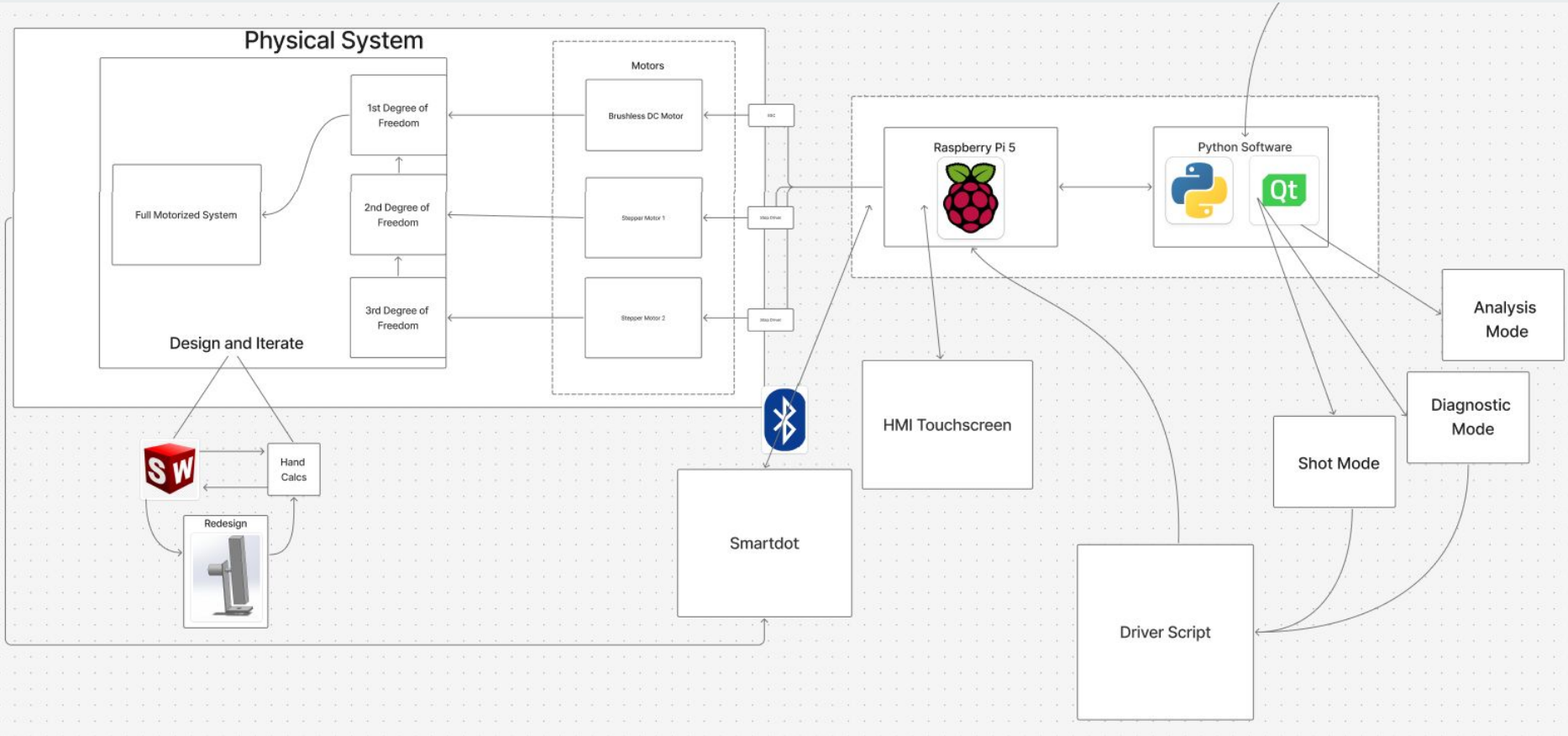
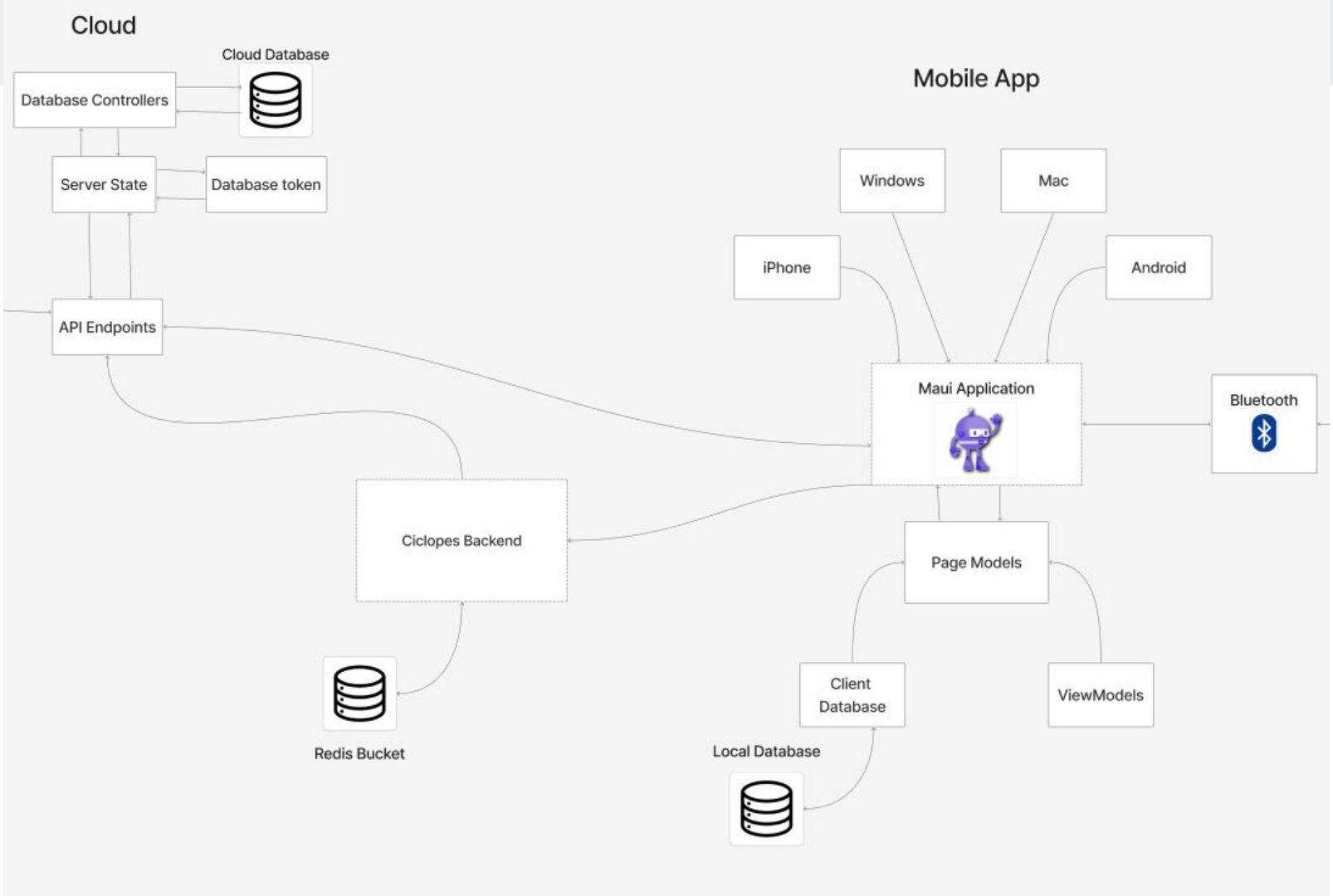
Milestone 2

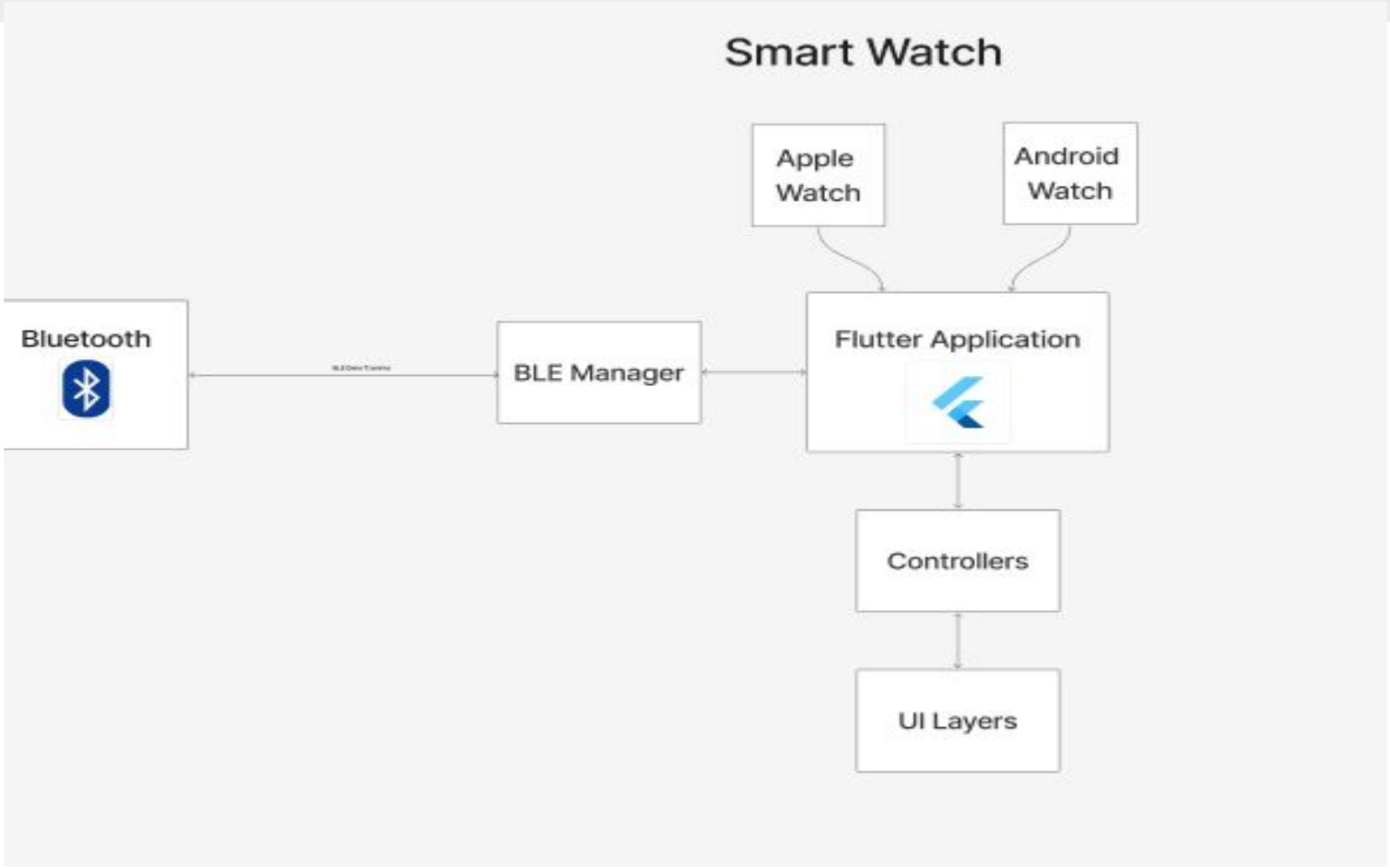
By: RevMetrix Team

HIGH LEVEL OVERVIEW

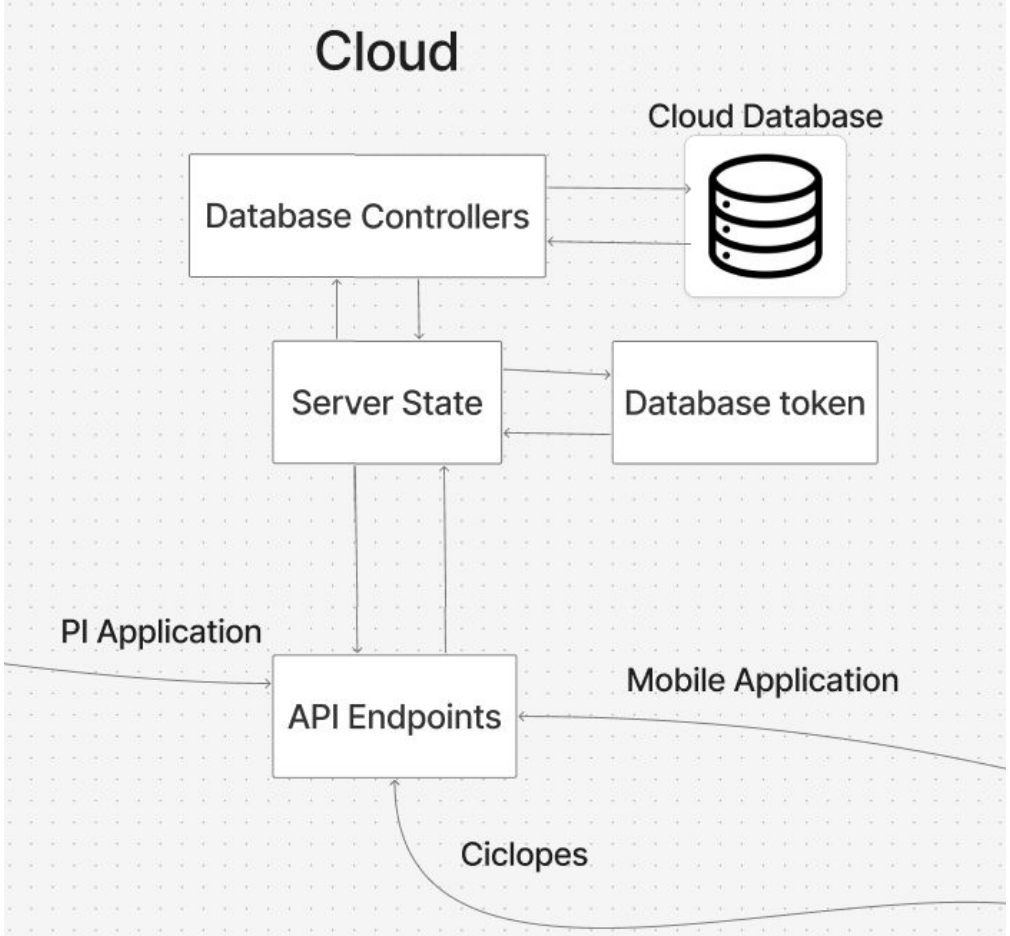



Physical System
Motors
1st Degree of Freedom
2nd Degree of Freedom
3rd Degree of Freedom
Full Motorized System
Design and Iterate
Brushless DC Motor
Stepper Motor 1
Stepper Motor 2
Raspberry Pi 5
Python Software
Qt
Analysis Mode
Diagnostic Mode
Shot Mode
HMI Touchscreen
Smartdot
Driver Script
Hand Calcs
Redesign

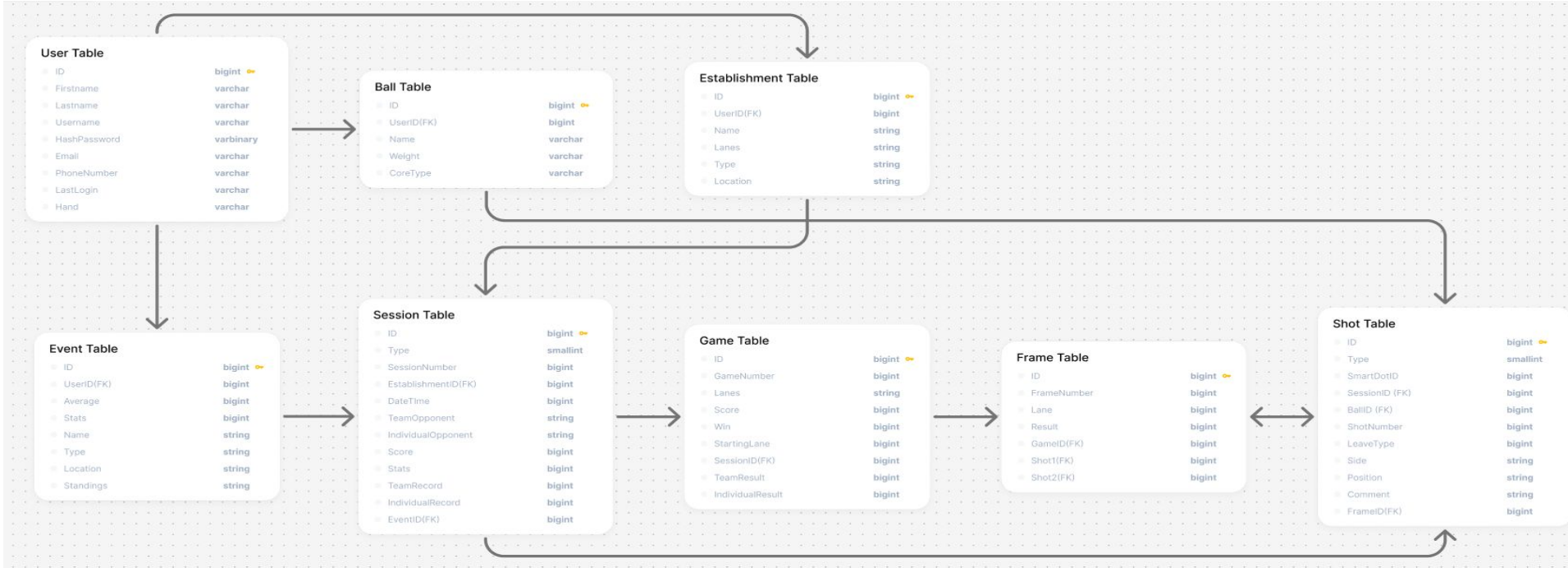

Cloud
Cloud Database
Database Controllers
Server State
Database token
API Endpoints
Mobile App
Windows
Mac
iPhone
Android
Maui Application
Bluetooth
Ciclopes Backend
Page Models
Redis Bucket
Client Database
ViewModels
Local Database


Smart Watch
Apple Watch
Android Watch
Bluetooth
BLE Manager
Flutter Application
Controllers
UI Layers

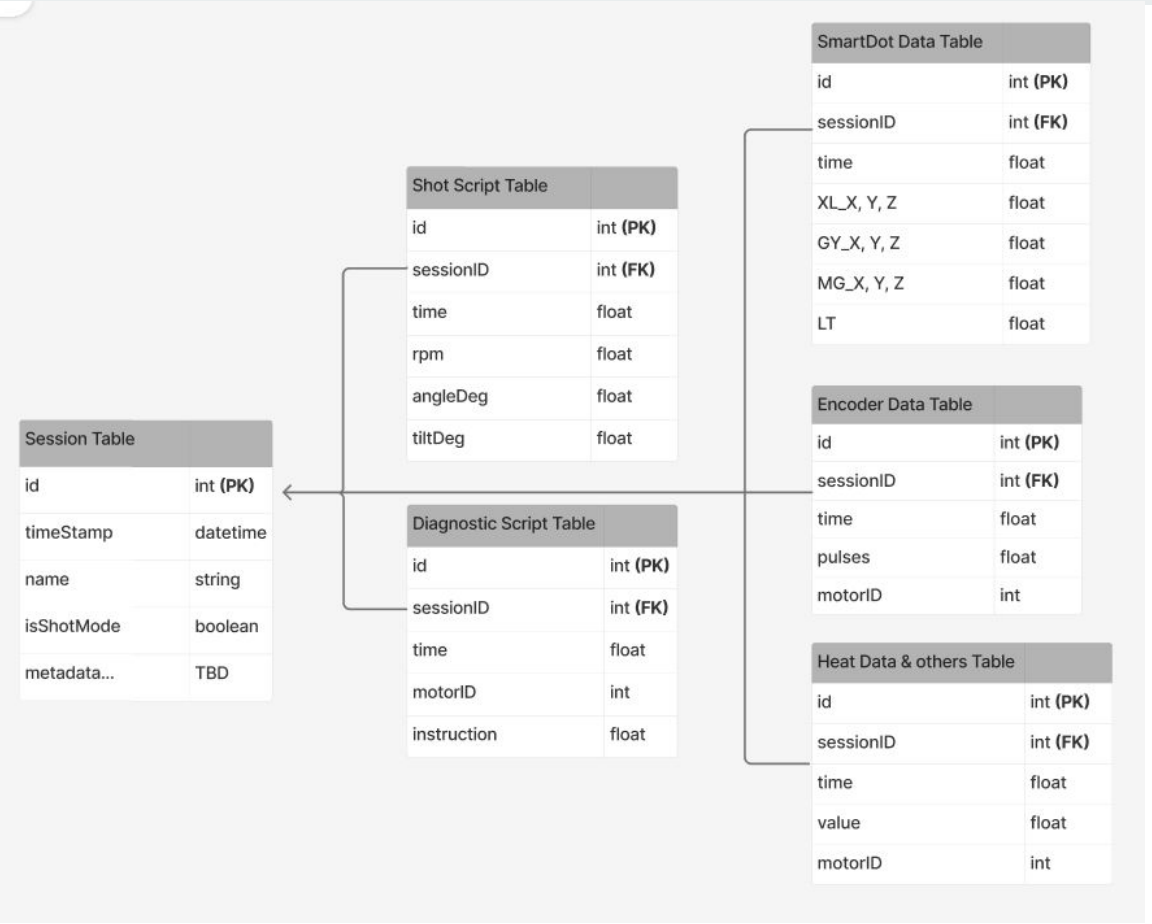
Backend/Cloud High-Level Overview



New (Mobile) Schema



New (Pi) Schema

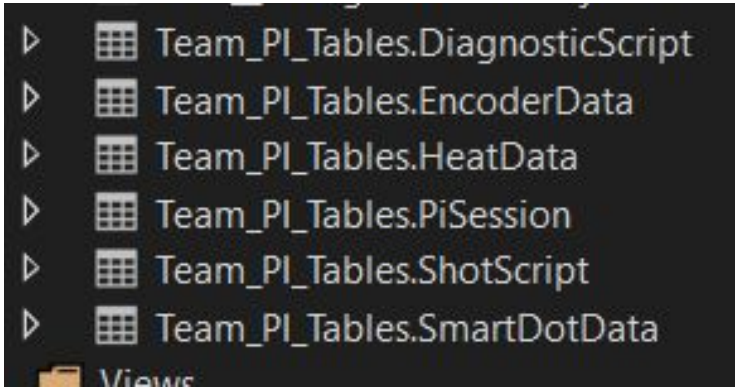


Cloud (MS2 Goals)

- Implement Pi Database
- Add API endpoints for new Pi Tables
- Master merge of Mobile & Cloud Database tables and API

Database Additions & Digital Ocean Update

- Added 6 new tables for Pi Team onto backend
- Pushed all of our API and DB changes from MS1 and MS2 onto the Digital Ocean droplet



RevMetrix-Cloud-Infrastructure - DigitalOcean Droplet Web Console - Google Chrome

cloud.digitalocean.com/droplets/381550500/terminal/ui/?os_user=root

```
  GNU nano 6.2                                  Fall2025DBChanges.sql
IF OBJECT_ID(N'[__EFMigrationsHistory]') IS NULL
BEGIN
    CREATE TABLE [__EFMigrationsHistory] (
        [MigrationId] nvarchar(150) NOT NULL,
        [ProductVersion] nvarchar(32) NOT NULL,
        CONSTRAINT [PK___EFMigrationsHistory] PRIMARY KEY ([MigrationId
    );
END;
GO

BEGIN TRANSACTION;
GO

IF NOT EXISTS(SELECT * FROM [__EFMigrationsHistory] WHERE [MigrationId]
BEGIN
    IF SCHEMA_ID(N'combinedDB') IS NULL EXEC(N'CREATE SCHEMA [combinedD
END;
```

Screenshot of our SQL script file in the Liquibase folder of our DO droplet

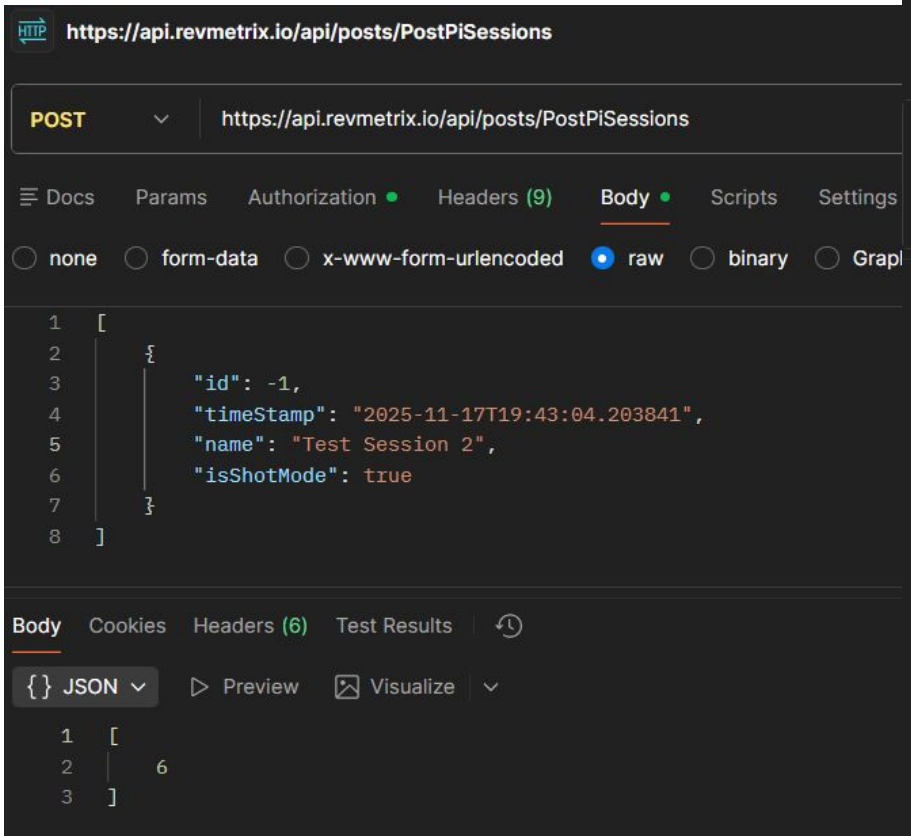
API Improvements/Additions

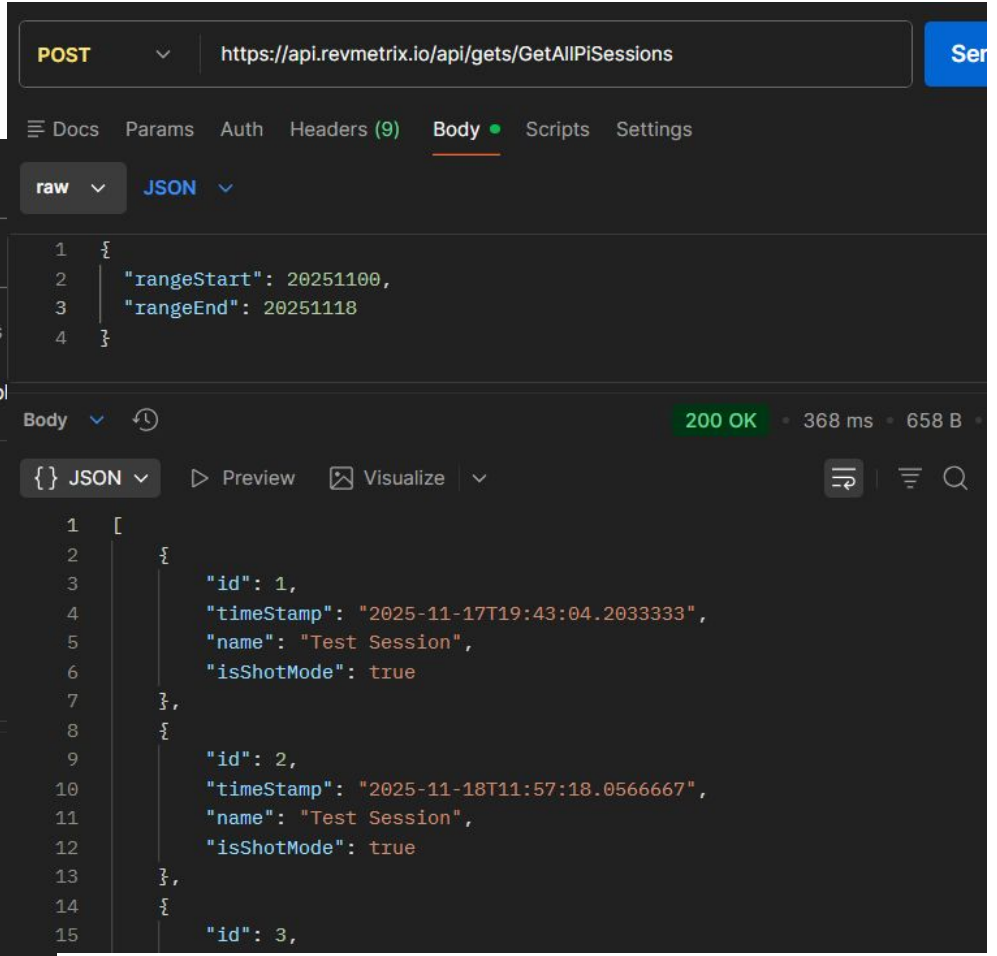
- 12 new endpoints added
- Create/Edit calls for all PI tables added
- Sorted results and parameterized retrieval

POST	/api/posts/PostPiDiagnosticScripts
POST	/api/posts/PostPiEncoderData
POST	/api/posts/PostPiHeatData
POST	/api/posts/PostPiSessions
POST	/api/posts/PostPiShot
POST	/api/posts/PostPiSmartDotData

GET	/api/gets/GetAllPiDiagnosticScriptBySession
GET	/api/gets/GetAllPiEncoderDataBySession
GET	/api/gets/GetAllPiHeatDataBySession
POST	/api/gets/GetAllPiSessions
GET	/api/gets/GetAllPiShotsBySession
GET	/api/gets/GetAllPiSmartDotDataBySession

API Improvements/Additions Cont.





Cloud (Future: Final Presentation)

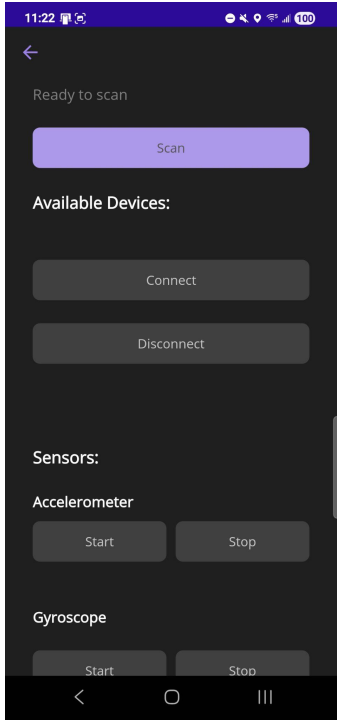
- Implement an API interface on the mobile application to interact with our newly created Database Tables and API Endpoints
- Continue to improve/modify/add to the API as needed by both the Cellular and Pi teams
- Add Edit/Delete endpoints to existing Tables that need them

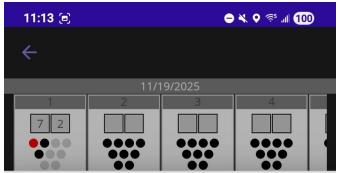
Mobile App (MS2 Goals)

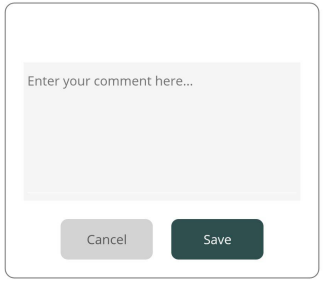
- Improve shot page
  - Refactor Shot page logic
  - Comment button functionality
  - Store dropdown selections
  - Edit shots capability
- Port Java Api to C#
- Cloud Database Connection
- Improve/ add unit tests

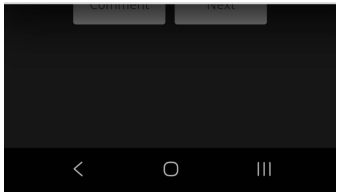
Mobile App (MS2 Achievements)

- Improve shot page
  - Refactor Shot page logic
  - Comment button functionality
  - Store dropdown selections (Some)
  - Edit shots capability
- Basic stats UI
- Video page resolution selection
- SmartDot is connecting to the App





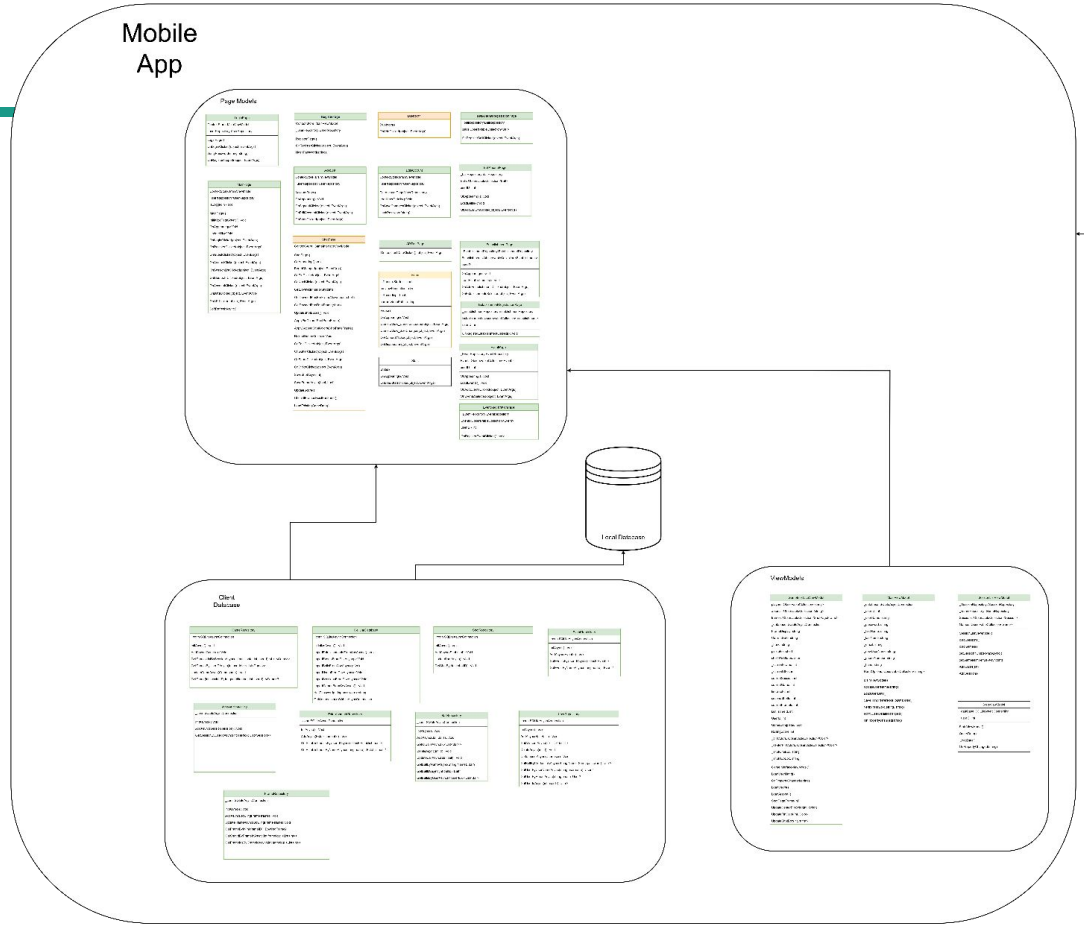


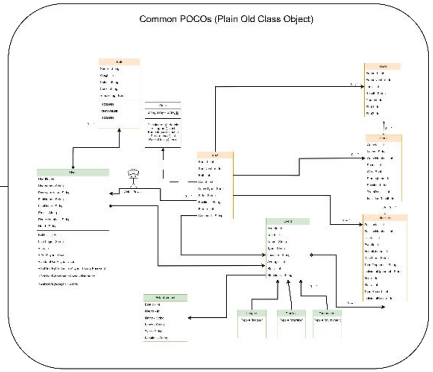


Mobile App (Future: MS3)

- Stats Page Functionality (Some)
- Finish Cloud database connection
- Upgrade to .Net 10
- Improved Unit Testing

Mobile UML Overview



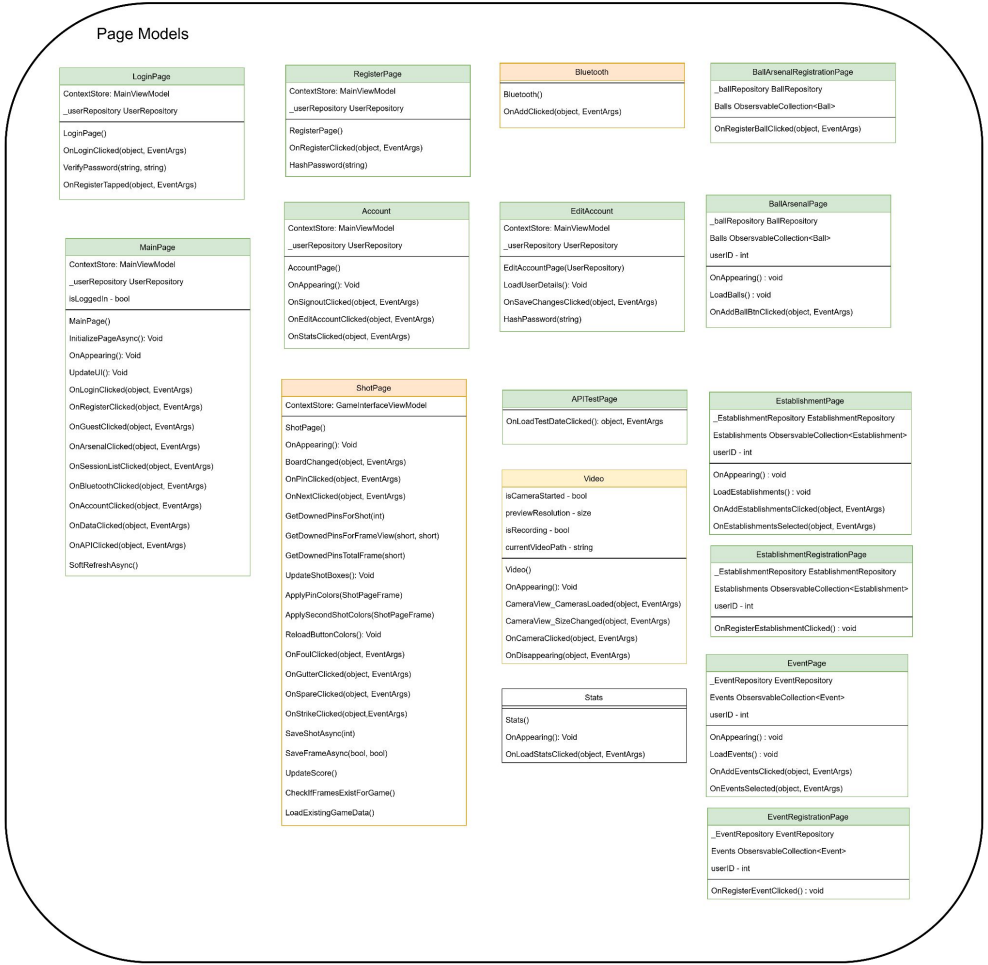


Implemented

Partially Implemented

Not Yet Implemented

Mobile UML - Page Models

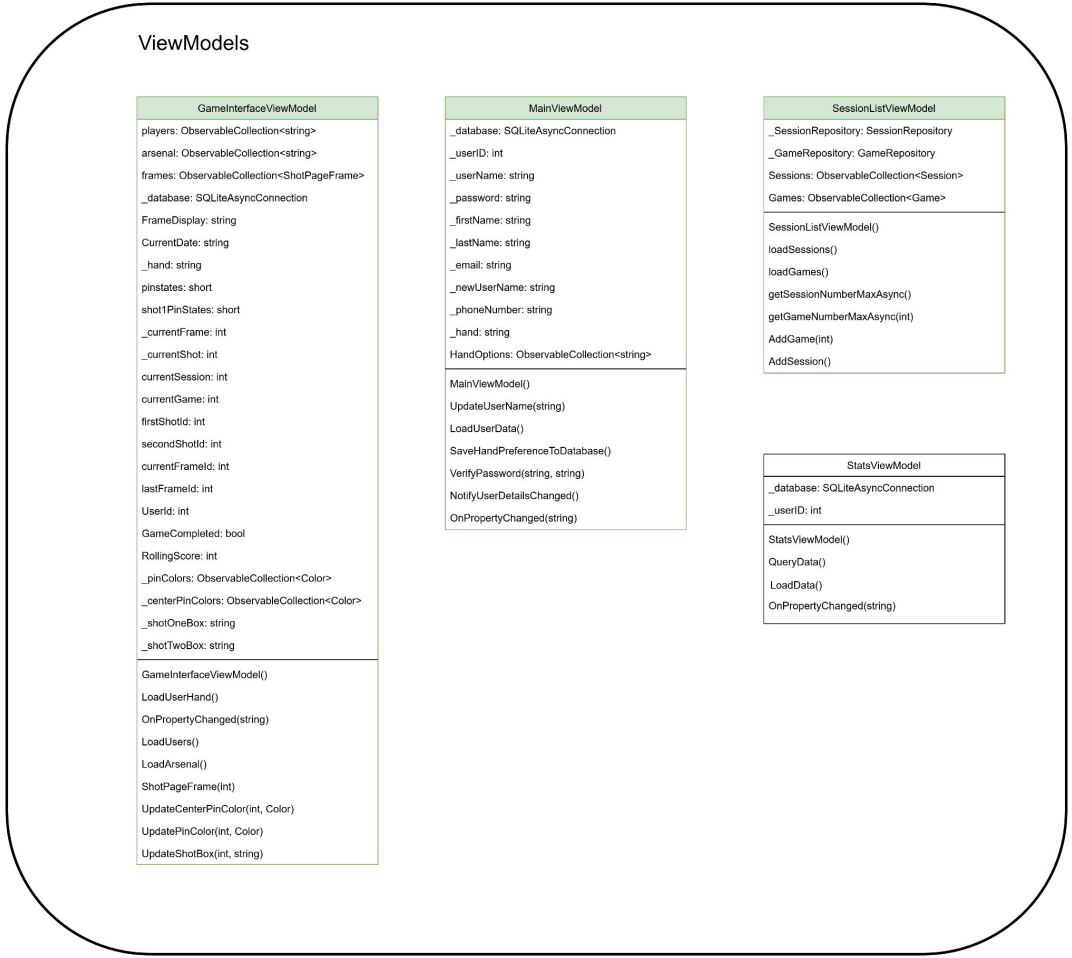


Implemented

Partially Implemented

Not Yet Implemented

Mobile UML - View Models



Implemented

Partially Implemented

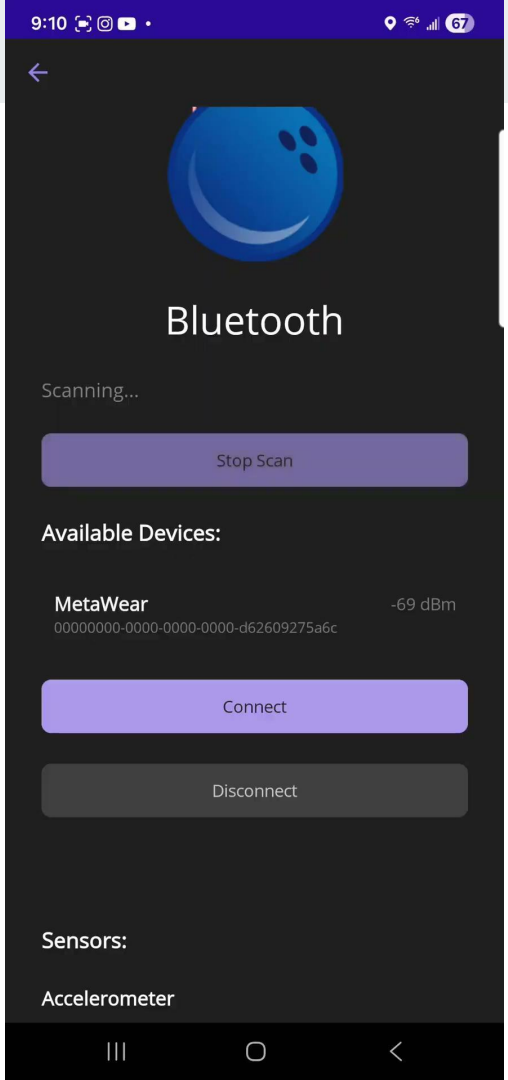
Not Yet Implemented

Mobile UML - Database

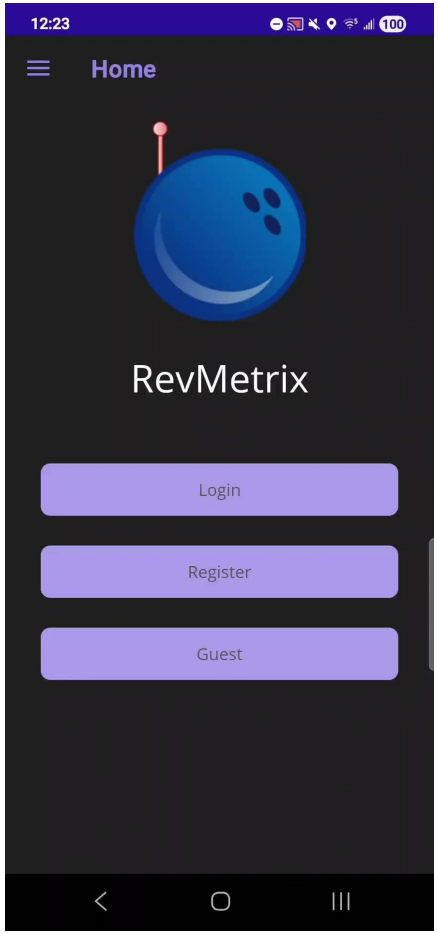
Client
Database

Smartdot Connection Established

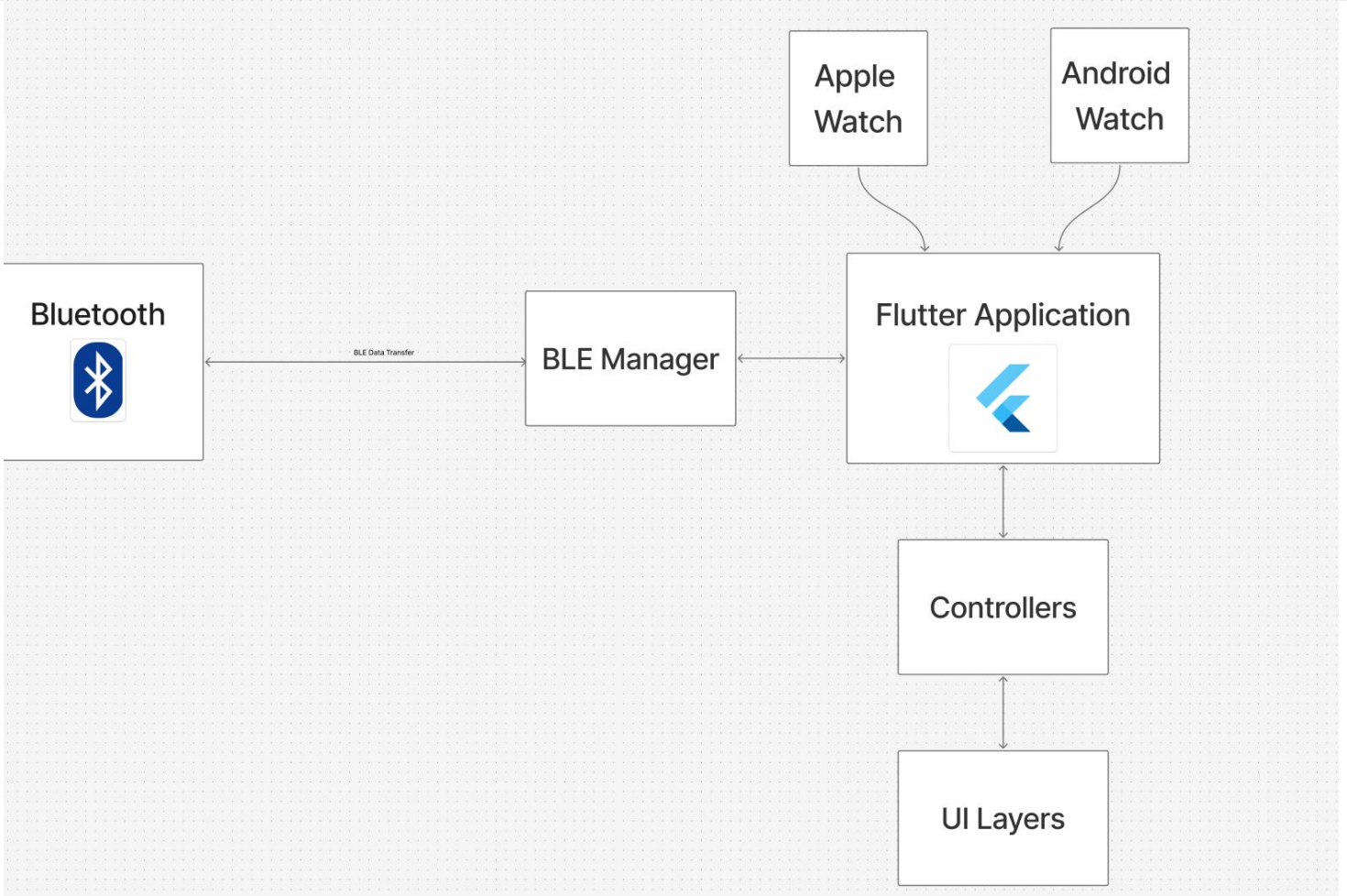
- Java -> C# did not work
- Connecting and Communicating directly with the components on the board
- Accelerometer: 0x03 (BMI270 on MMS), Gyroscope: 0x13 (BMI270 on MMS), Light Sensor: 0x14 (LTR-329ALS-01), Magnetometer: 0x15 (BMM150)



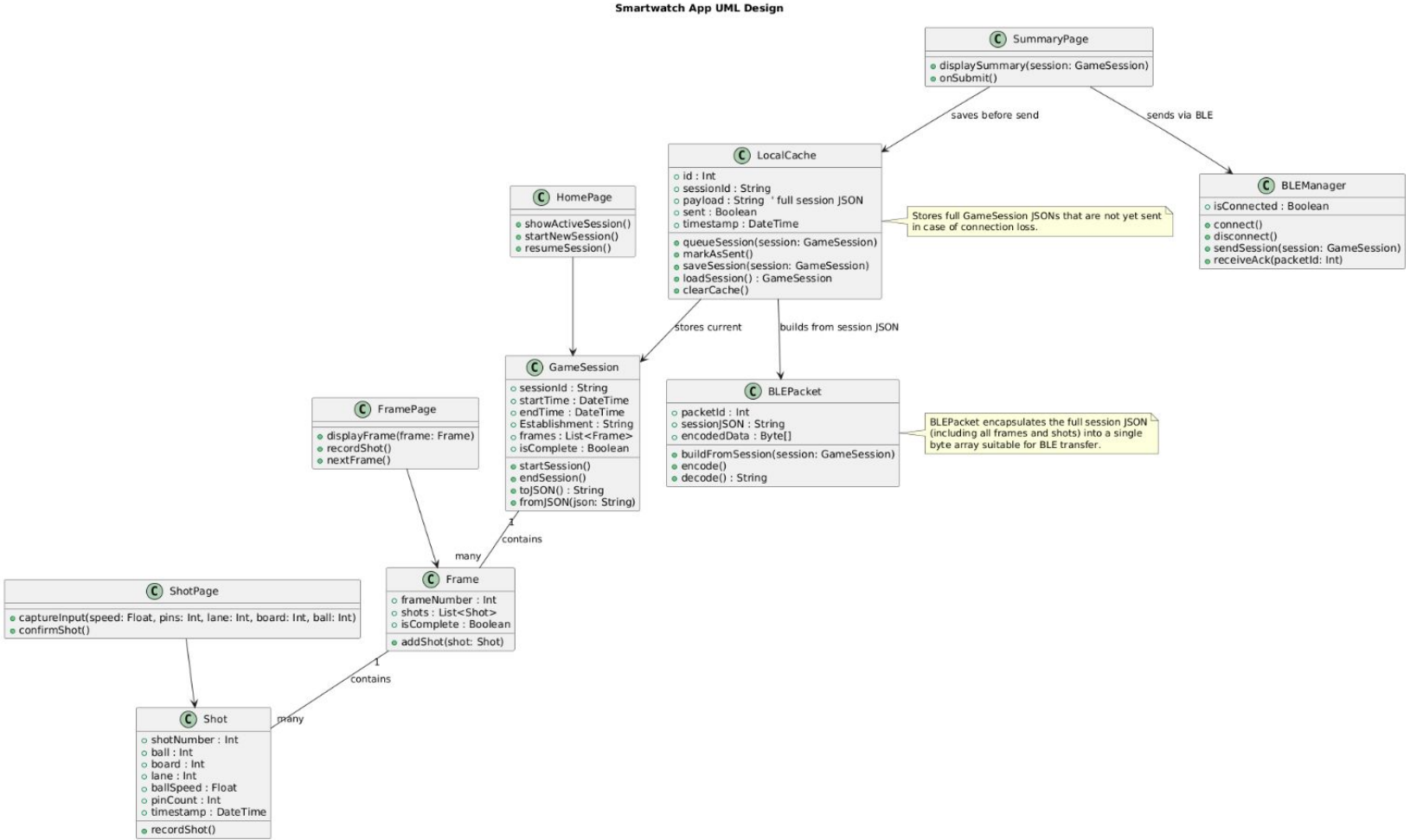
Mobile App - Demo!



Smart Watch High-Level Overview



Smart Watch UML



Smart Watch (MS2 Goals)

- Establish BLE Comms
  - Connect the watch to the phone
- Update models with relevant information
- Make views dynamic
- 2 phase shot input

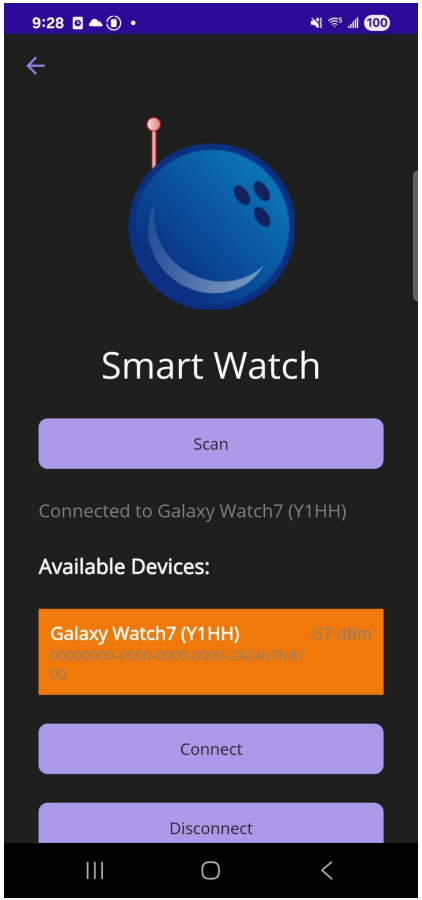
Smart Watch (MS2 Achievements)

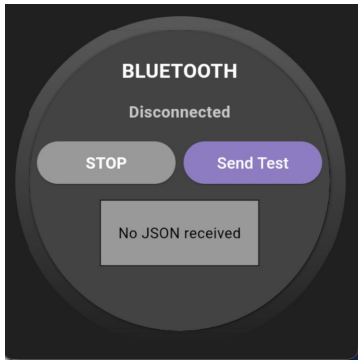
- Added Bluetooth comms
  - BLE peripheral (watch) hosting custom GATT server
  - BLE client (phone) using Plugin.ble
- Updated models with relevant data
- Made Game and Frame/Shot views dynamic
  - The game views now react to the number of game objects
- Added frame view logic
  - Can now only access the frames up to the active frame
  - Processes the frame/shot objects for persistent data
- Updated shot input to 2 phase
  - Pre shot, stance, board, lane, ball, start recording
  - Post shot, pins select, shot speed, strike/spare and gutter/foul selection

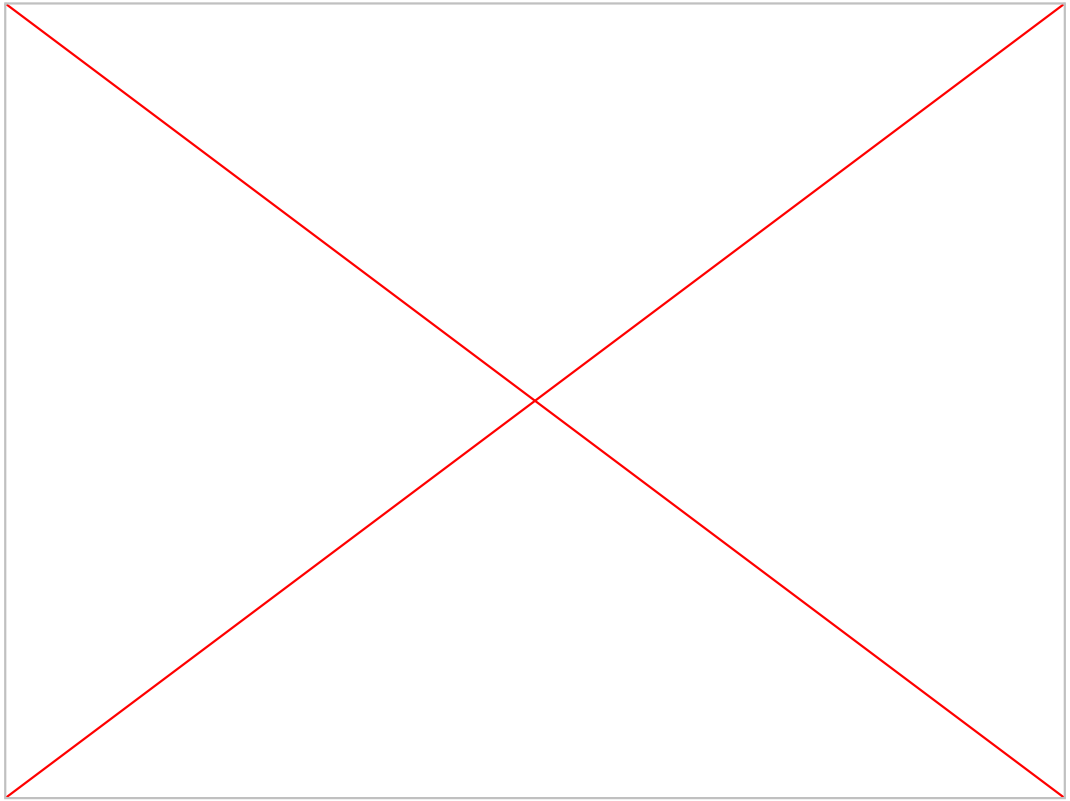
Smart Watch (Future MS3)

- Local Storage
  - Objects already organized and populated for local storage integration
  - Implement save to local storage
- BLE research
  - Packet work to send all data via BLE
  - Find limitations
  - Research what exists already
- Have minimally working system for game test

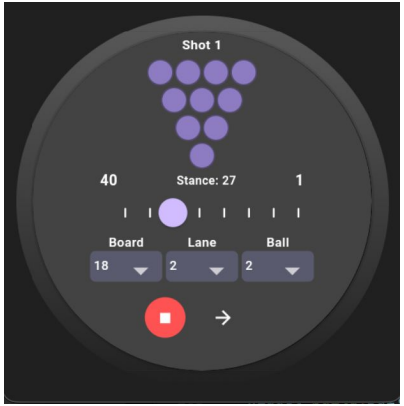
Smartwatch Bluetooth Demo





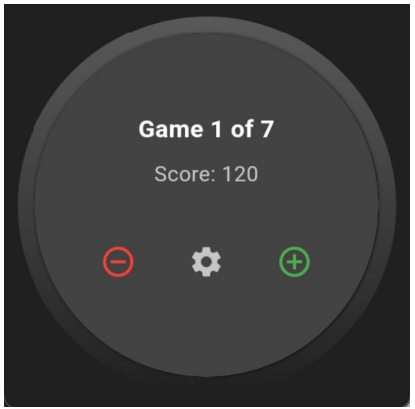


Smartwatch Functionality Demo



Pre shot

Post shot



- Adaptive Games
- Adaptive Frames

Ciclopes - Goals for MS2

- Generate episodes of more realistic trajectories for testing
  - Export with per frame BEV positions and kinematics labels
- Develop homography warping functionality
  - Test with labeled BEV positions as ground truth
- Develop BEV episode buffer -> kinematics / features functionality
  - Visualizations? What is actionable data?
  - Post processing tricks (eg. frame interpolation, detection hacks)
- Run end to end segmentation -> warp -> features testing on labeled episodes

Ciclopes - Accomplishments for MS2

- Generate episodes of more realistic trajectories for testing
  - Export with per frame BEV positions and kinematics labels
- Develop homography warping functionality
  - Test with labeled BEV positions as ground truth
- Develop BEV episode buffer -> kinematics / features functionality
  - Visualizations: Render trajectories
  - Actionable Data: Velocity at specific points, Break(curve) statistics
  - Frame Interpolation: Cubic Spline vs Piece Wise Linear Approximation
- Run end to end segmentation -> warp -> features testing on labeled episodes

Ciclopes - Deep Learning Overview

Inference

Extract features from image

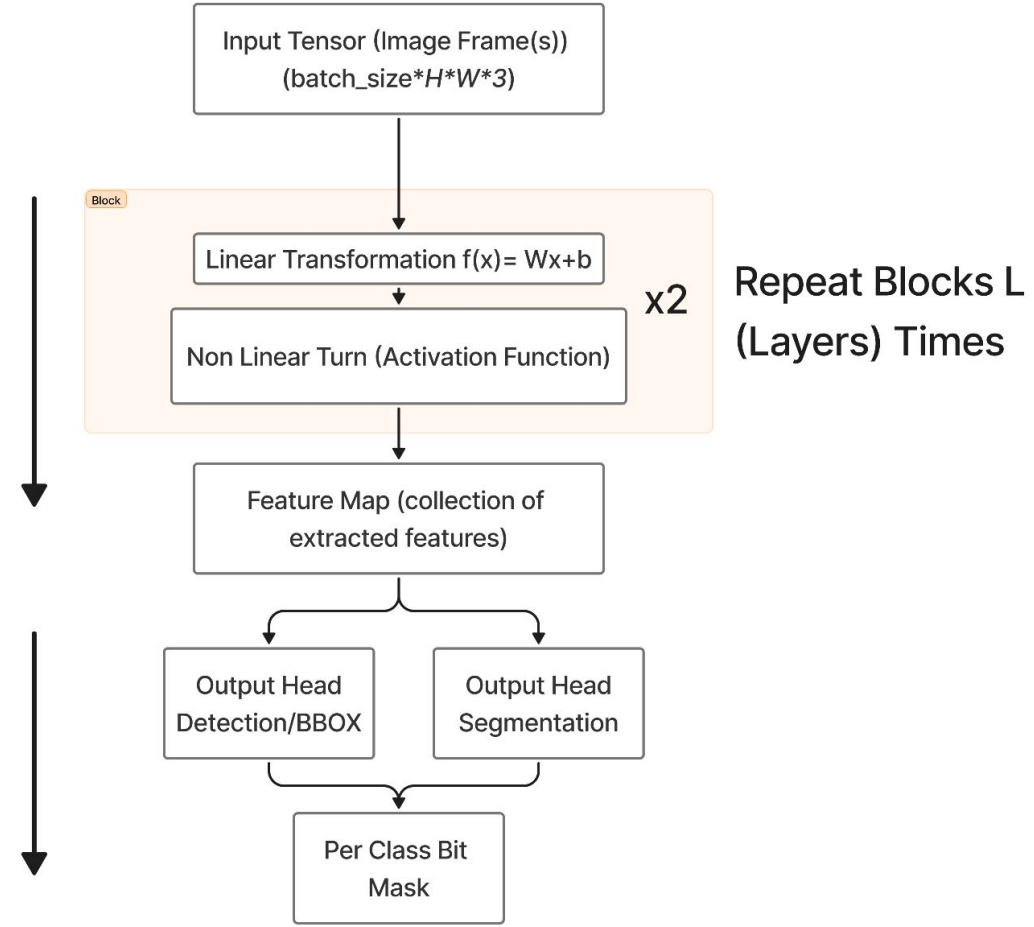
- Low Layers(eg. corners→edges→colors)
- Middle Layers (eg. object parts, shapes)
- High Layers (eg. “Ball”, “Lane”)

Project Representations of each layer to common Dmodel and mix fuse into final feature map

Representations Ex.

- Input (2048×2048×3)
- Low (1024×1024×6)
- Middle (256×256×24)
- High (16×16×384)

Choose Dimension to project to and fuse into feature map



Pre Processing Overview

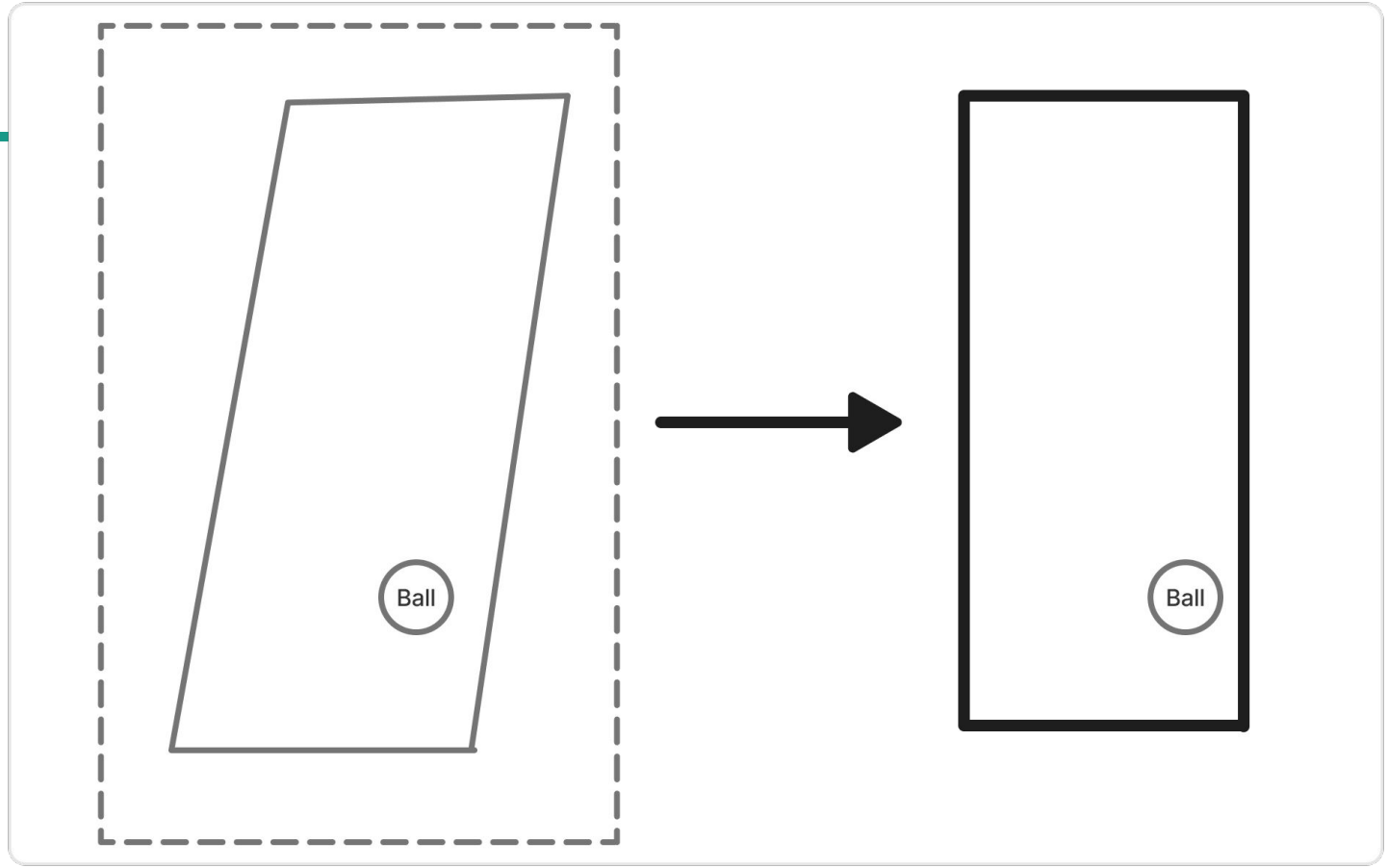
1. Runs YOLOv11-seg to segment ball and lane masks

- Returns buffer of dicts (JSON shape) containing lane and ball masks

Post Processing Overview

1. Calculate BEV Homography Transformation
    a. Uses segmented lane mask, and known values
    b. Computes transformation to reuse for the ball
    c. Ball must be in contact with the lane, and separation causes drift
2. Compute linear mapping of BEV units (pixels) to real world units (meters)
3. Compute centroids in BEV coordinates
4. Compute velocities / accelerations at different points down the lane (0%, 25%, 50%, 75%, 100%)
5. Compute total break, and open to compute more special features
6. Return data structure with a buffer of ball centroids in BEV, vel/accel values, break total, and open for more

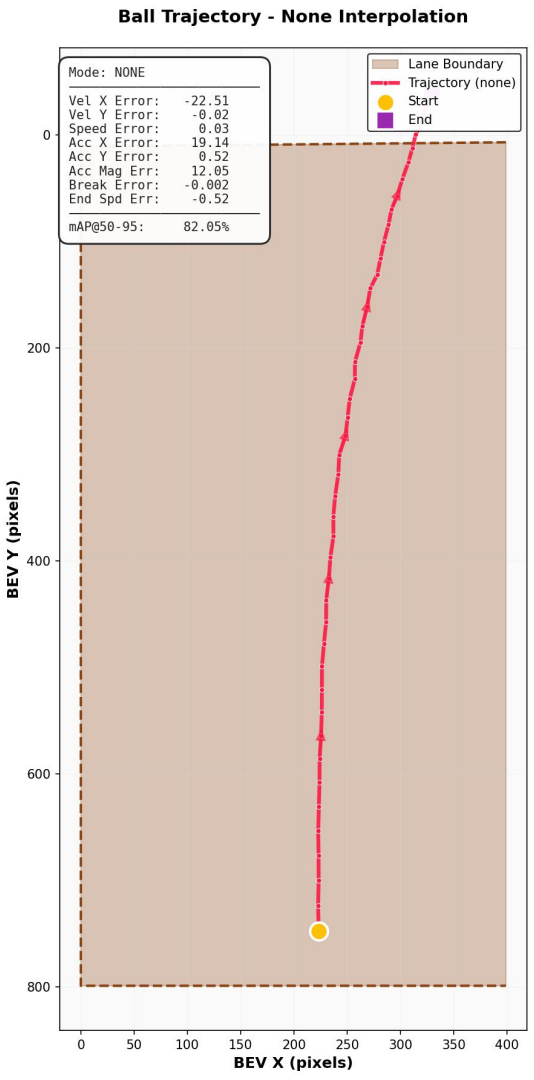
Homography Explanation



End to End Results

Metric	None	Linear Approximation	Cubic Spline
mAP@50-95	0.820	0.820	0.820
Avg Total Break Error	0.003	0.003	0.003
End Speed Error	~0.522	~0.531	1.411 - 0.632
Vel MAE	0.383	0.429	0.683
Accel MAE	9.33	3.23	12.77

Example Rendering



Where To From Here

Now:

- Technique for centroid extraction confirmed
- Segmentation of ball and lane is all that is needed

Future:

- Sim2real / real world working vision model
  - Difficult
  - Noisy
  - Expensive (Effort/Time wise)

SAM3 - Segment Anything

Pros:

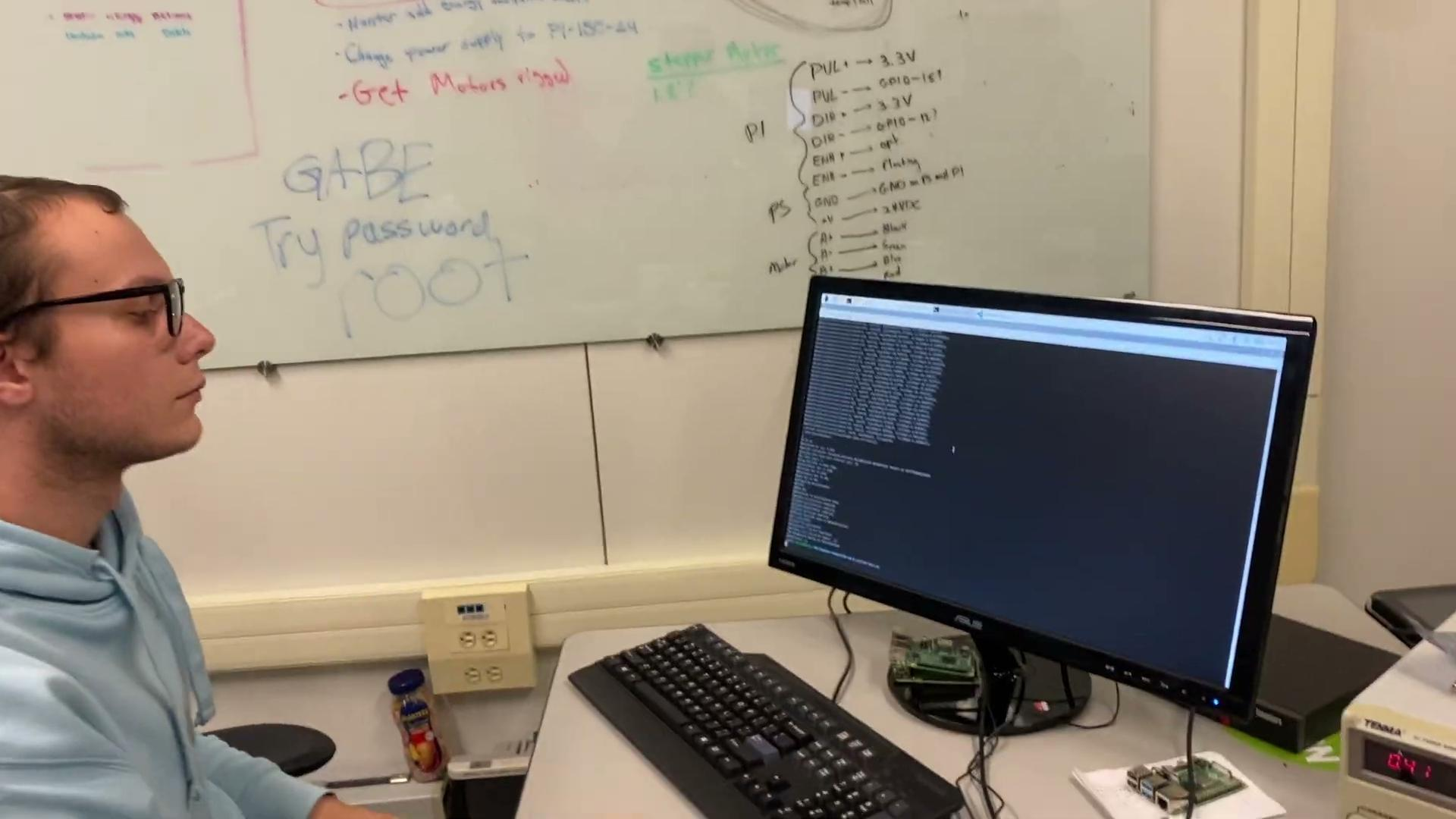
- Text based input (Software 3.0)
- Will segment balls and lanes / other things
- Native 2D pose detection → 3D points to render

Cons:

- Will segment all lanes / balls
- Larger, no option to fine tune

Ciclopes - Goals for MS3

- Document engine
  - Cleanup ugly/unused code
- Finalize research and concrete plan for sim2real
- Research SAM3/3D and how that could be leveraged
- Finalize plan for application implementation to timestamp frames to process, and detect ball off the lane / other edge cases (ex. Not visible)
- Prepare for implementation in real world use


- Change power supply to PI-15V-4A
-Get Motors rigged
GtBE
Try password
root
PUL+ → 3.3V
PUL- → GPIO-18?
DIR+ → 3.3V
DIR- → GPIO-12?
ENA+ → opt
ENA- → floating
GND → GND on PS and PI
+V → 24VDC
PI
PS
Motor
A+ → Blue
A- → Green
B+ → Red
B- → Black

Team Pi (Goals for MS2)

Hardware

- Implement safety
- Design and evaluate power distribution and source options (battery vs plug in power configs)
- Test motor operations under various conditions
- Integrate sensors and upgrades
- Begin Stepper Motor connections with appropriate drive module

Team Pi (Achievements for MS2)

Hardware

- New BLDC + ESC setup
  - Built custom wiring harnesses for BLDC phase leads, Hall sensor and temp sensor
  - Navigated VESC Tool setup for motor detection, firmware and calibrations
- ESC + Pi communication setup
  - Attempted UART communication resorted to USB comms
  - Initialized libraries and tested Python scripts
- Hall + temp sensor integration
  - Confirmed real time RPM and duty cycle feedback
  - Validated motor readings
- Nema 17 stepper motor + DM542 driver working
- Physical layout and wiring
- Power system transition to 24 V industrial supply (P1-150-24) for motors
  - 24 → 5 V buck converter for stable Rpi 5 power

Team Pi (Future: MS3)

Hardware

- ESTOP, fuses and more electrical protections
  - Includes indicator lights
- Integrate hall sensor and temp sensor data into UI
- Get 3rd degree tilt stepper motor running
- Combined motor system testing
  - Develop synchronized movement routines to simulate bowling ball shot
- Work with MEs to integrate motors and connecting the two degrees
  - Cable routing and stress relief

Team Pi (MS2 Goals)

Software

- Organize repository
- E-stop button
- Unit Test
- Investigate Frontend Testing utilities
- Add further diagnostic page functionality
- Add logging to all of our important functions
- SmartDot Simulator
- Shot script and database implementation
- Add stored shot/session page

Team Pi (MS2 Achievements)

Software

- Develop “shot script” for the primary BLDC through a bowling shot sequence (2 Motors!!)
- Organize repository
- E-stop button
- Investigate Frontend Testing utilities
- Increase robustness of the SmartDot Connection
    - SmartDotConnectionManager
- Built an extensive ReadMe for Raspberry Pi Setup
- Add logging to all of our important functions
- SmartDot Simulator

Team Pi (MS2 Achievements Continued)

Software

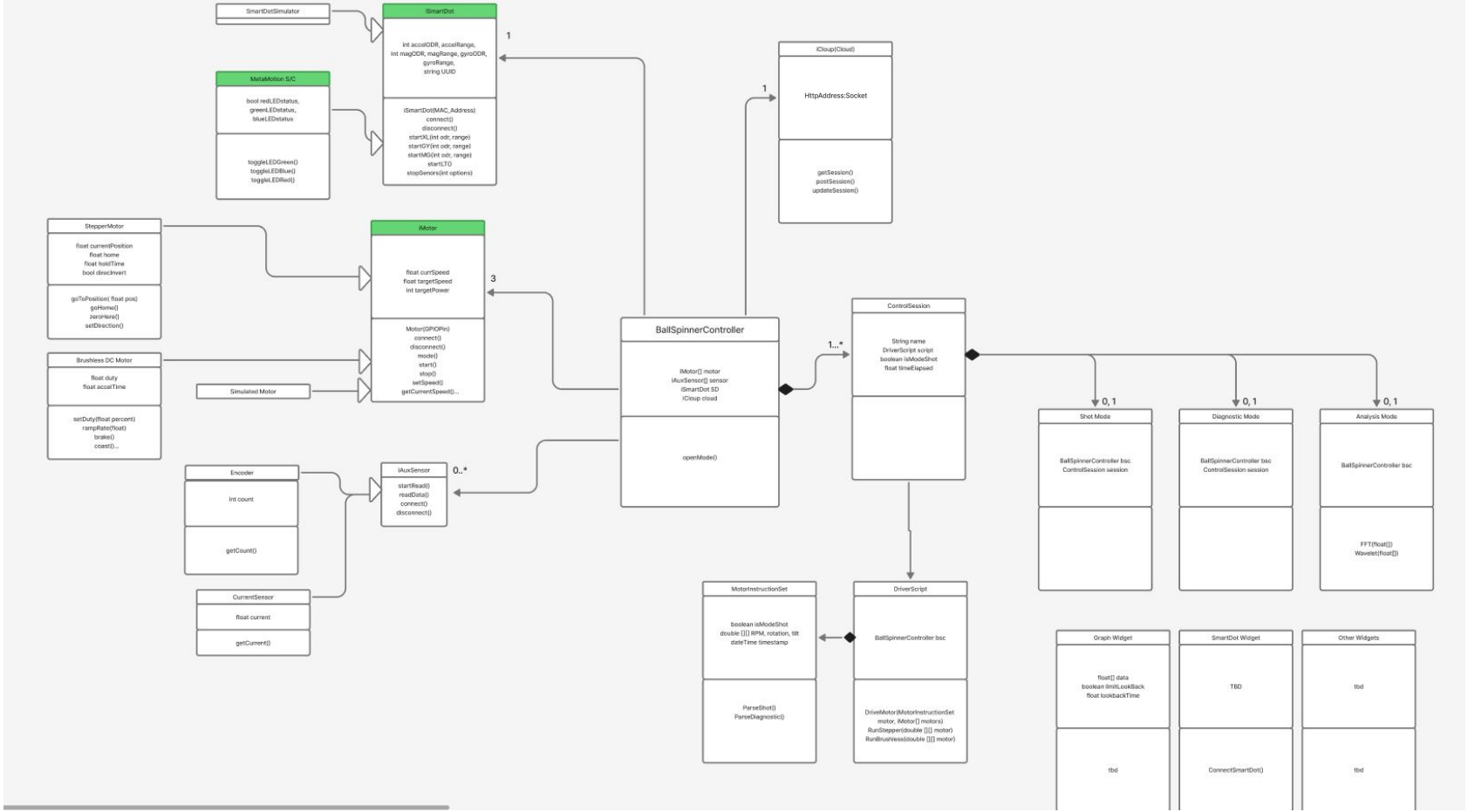
- Added Motor functionality to Diagnostic page
- Shot script and database implementation
- Add stored shot/session page
- Data Models
- Data Controller
- Full Cloud Integration
- Full Navigation system
- Encoder on primary motor log integration

Team Pi (Future: MS3)

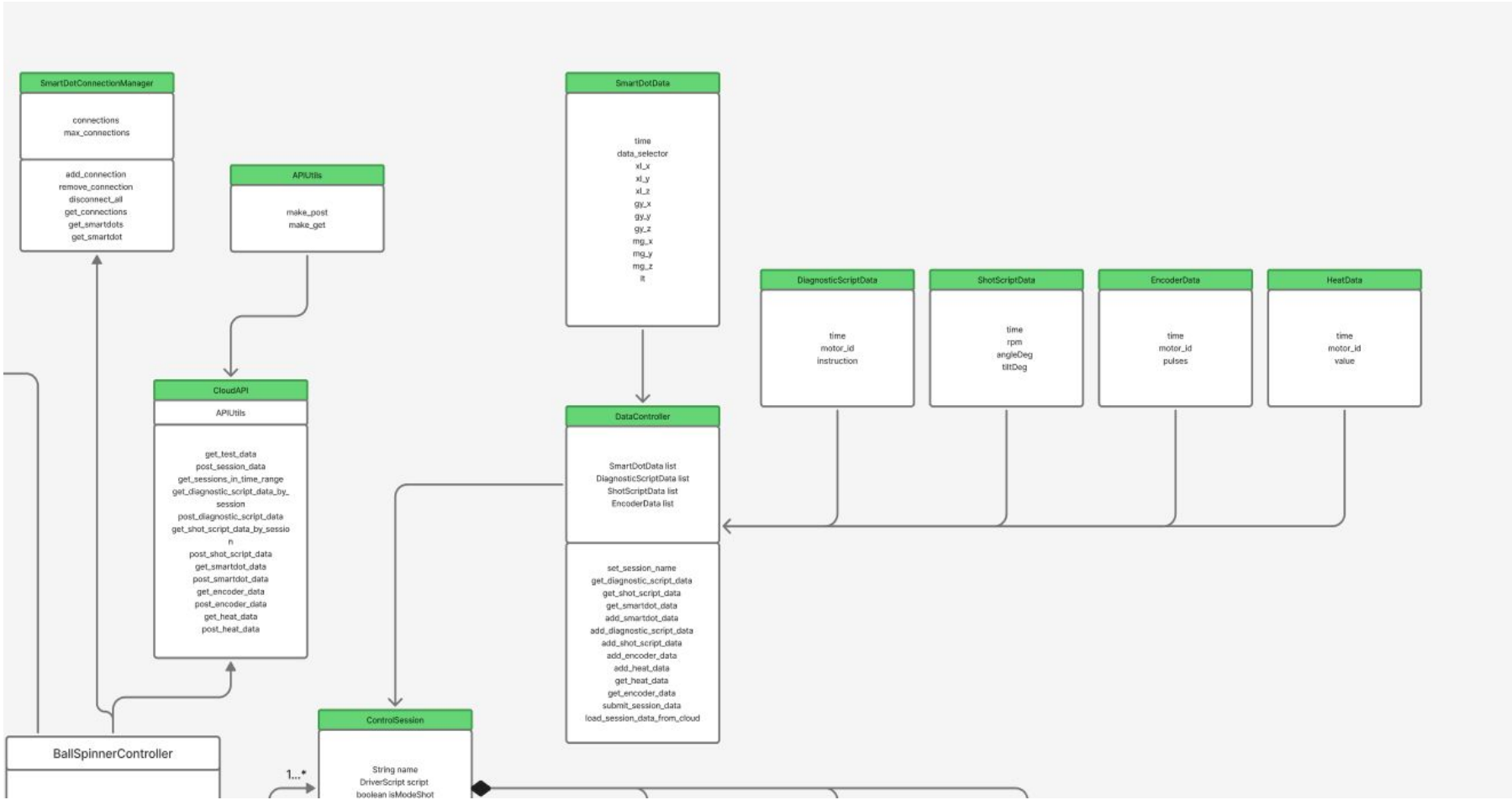
Software

- Unit Test
- Analysis page functionality
- Refining ability to replay shots
- Adding full save and replay of diagnostic sessions
- Add Loading modal for API requests
- Fix bugs with timing and stepper motor
- Fully update UML
- Display Heat and Encoder Data
- Remember MMS

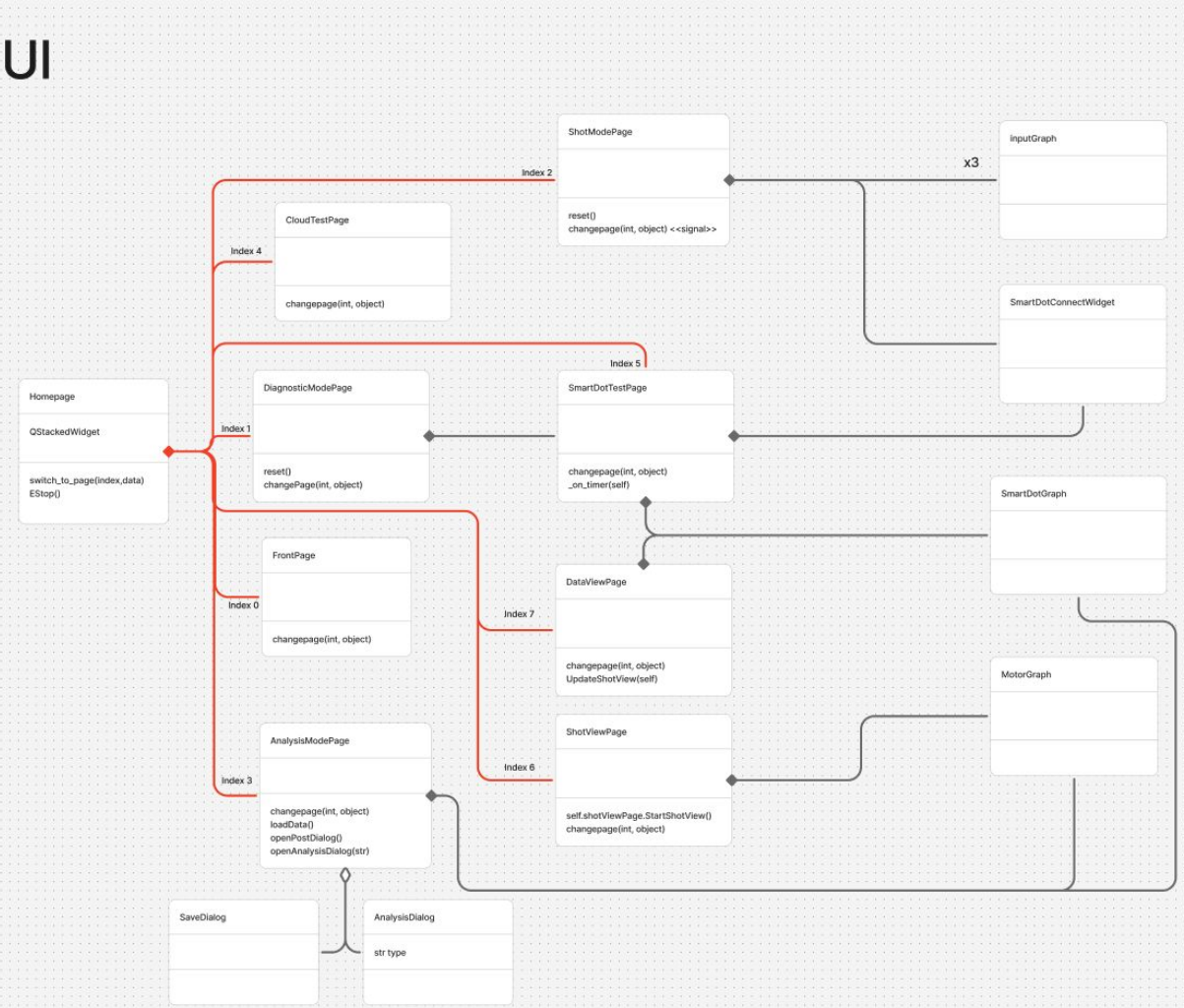
Team Pi UML (OLD)



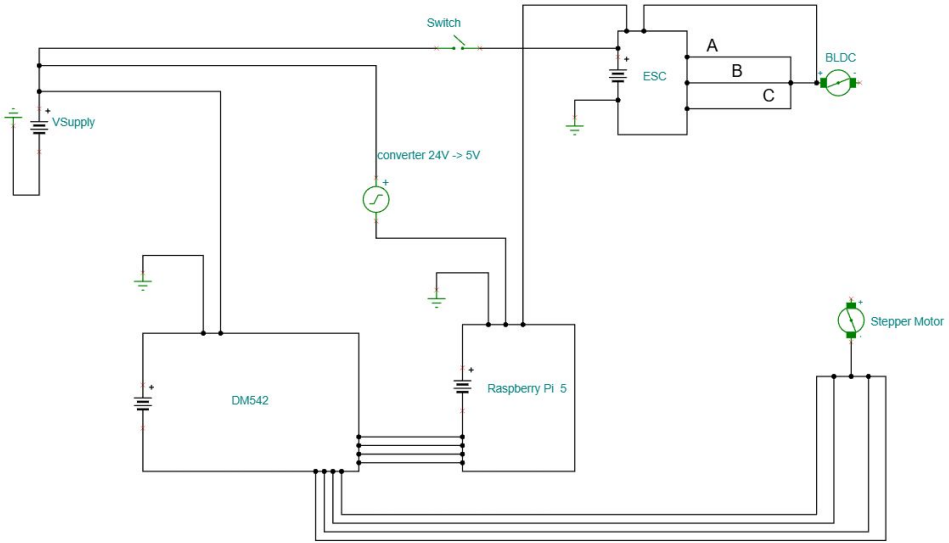
Team Pi UML (New)



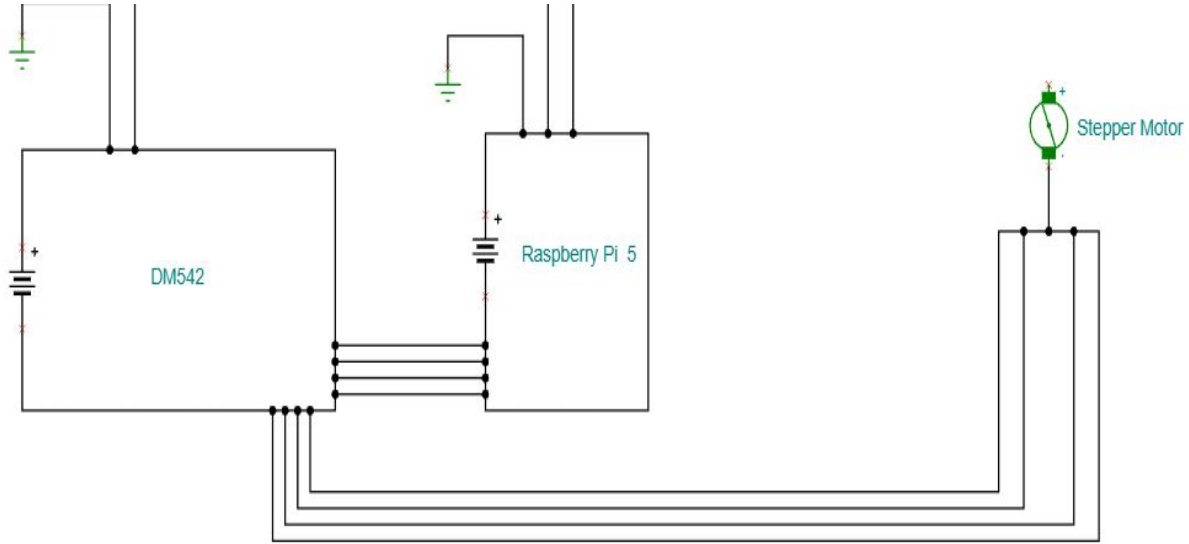
UI UML


UI
ShotModePage
reset()
changepage(int, object) <<signal>>
Index 2
x3
inputGraph
CloudTestPage
changepage(int, object)
Index 4
SmartDotConnectWidget
Index 5
Homepage
QStackedWidget
switch_to_page(index,data)
EStop()
DiagnosticModePage
reset()
changePage(int, object)
Index 1
SmartDotTestPage
changepage(int, object)
_on_timer(self)
SmartDotGraph
FrontPage
changepage(int, object)
Index 0
DataViewPage
changepage(int, object)
UpdateShotView(self)
Index 7
MotorGraph
ShotViewPage
self.shotViewPage.StartShotView()
changepage(int, object)
Index 6
AnalysisModePage
changepage(int, object)
loadData()
openPostDialog()
openAnalysisDialog(str)
Index 3
SaveDialog
AnalysisDialog
str type

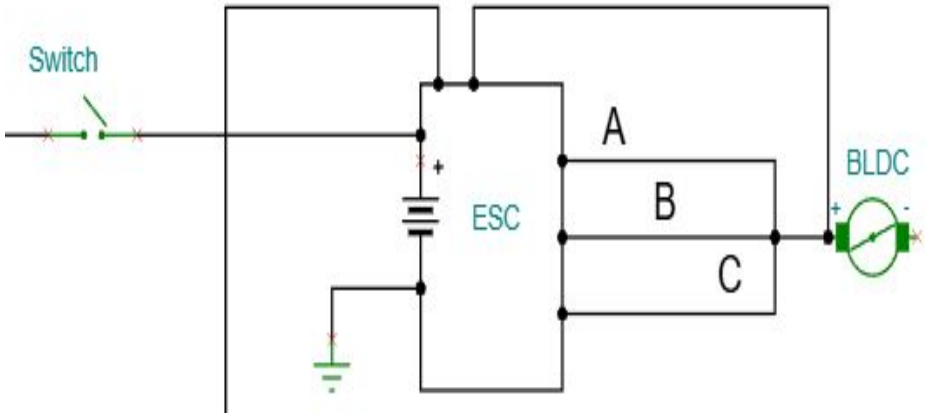
Team Pi Milestone 2 Hardware System Overview



2nd Degree Motor



Updated BLDC 1st Degree Motor



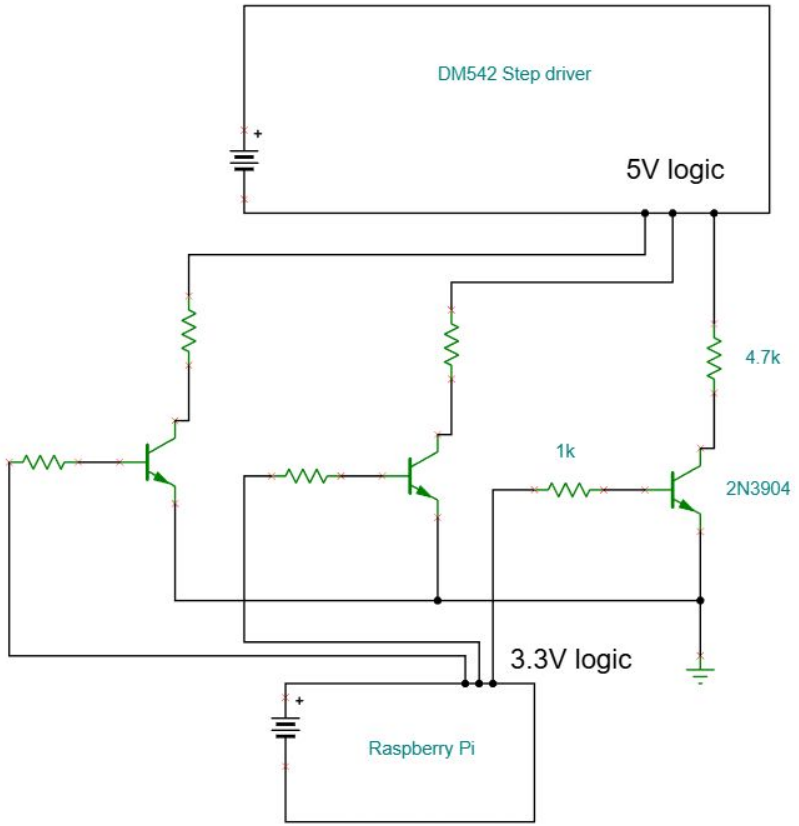
Logic Level Shifter

• Raspberry Pi GPIO outputs 3.3V, but the DM542 stepper driver requires 5V logic for PUL, DIR, and ENA signals.

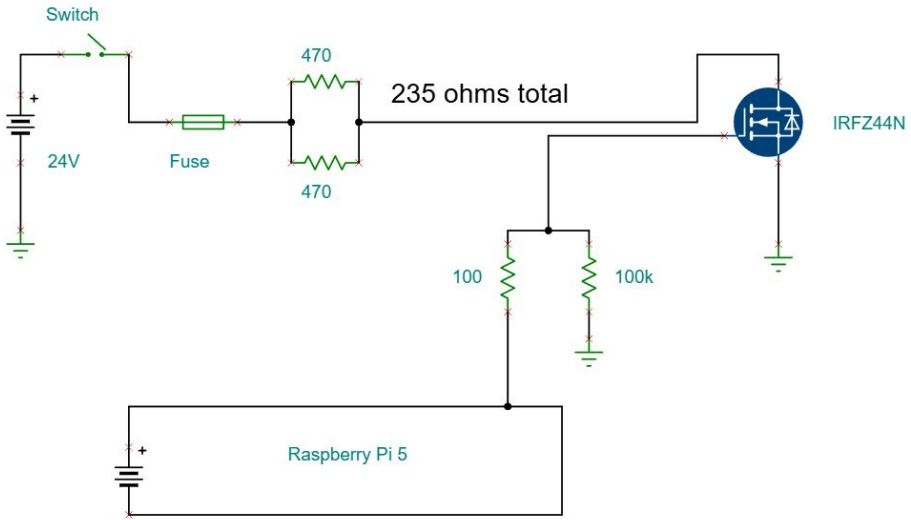
• A 3.3V signal is not guaranteed to register as HIGH for the DM542 → risk of missed steps or no motor movement.

• Safely converts 3.3V GPIO signals → 5V control signals for the driver.

• Ensures reliable, clean, noise-immune pulses for step and direction control.



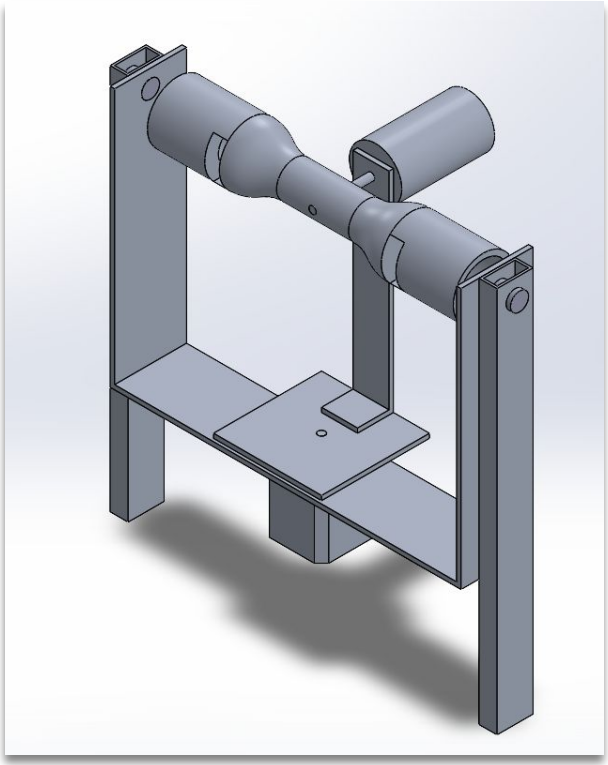
Resistive Brake



Physical Design Key Notes

- **Key Goal:** Motorize 1st degree of freedom and begin motion testing

- **Key Technologies:**
  - SOLIDWORKS - used to 3D model the prototype as well as future iterations
  - 3D Print Lab

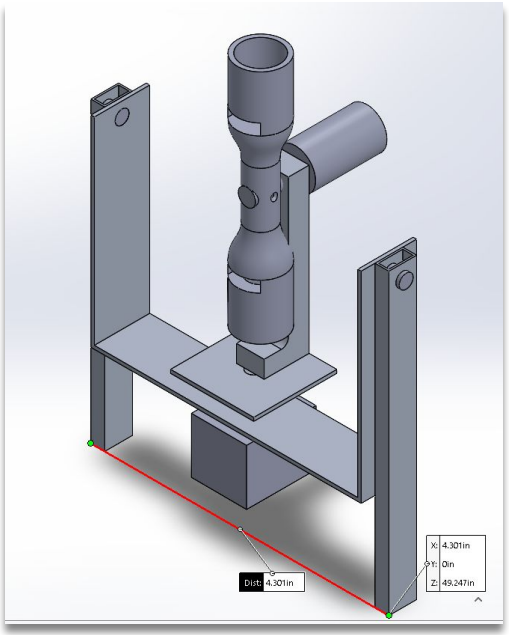


Updated model for aluminum prototype

Physical Design High Level Overview

- **High Level Overview of Model:**
  - Breakdown revised model
  - Key Changes (Wooden to Aluminum):
    - SmartDot Stick
    - Material
    - L-Bracket
    - U-Bracket
    - Size & Weight (3.5x in weight, 2.2x in overall length)
    - Hollow Supports

- **Key Technologies:**
  - SOLIDWORKS - 3D Modeling
  - 3D Print Lab



Scaled Down aluminum prototype (40% of full scale model)

SmartDot Support Overview

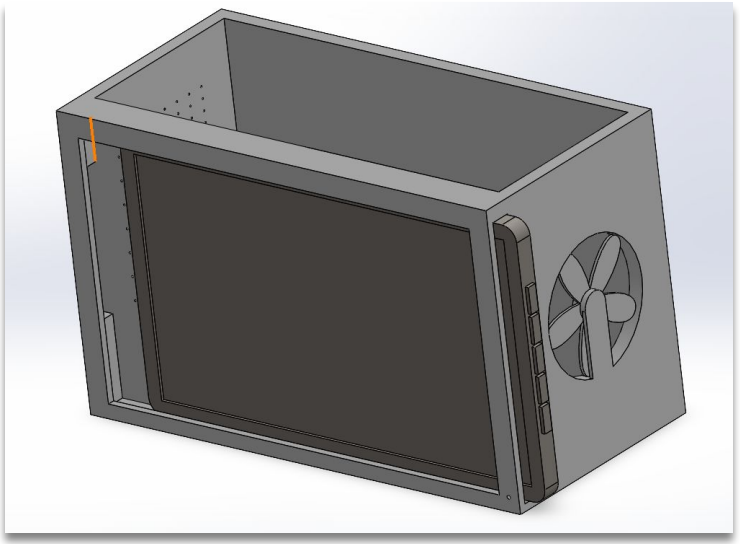
- **High Level Overview of Model:**
  - Key Changes
    - Cylindrical Body
    - Friction fit for SmartDot
    - Set screw
    - Dimensioned to bowling ball
    - Low infill to minimize weight

- **Key Technologies:**
  - SOLIDWORKS - 3D Modeling
  - 3D Print Lab

Updated SmartDot Support

Physical Design

- **Parts that are implemented:**
  - Updated system model
  - SmartDot Housing Stick
  - BSC housing model
  - Acrylic Safety Casing
- **To be implemented:**
  - Vibration Control
  - 1st and 2nd DOF motor integration with Team Pi
- **For implementing future functionality:**
  - Work with Team Pi to ensure motor compatibility for MS3



Current BSC Housing Design
Featuring Fan and Ventilation Ports