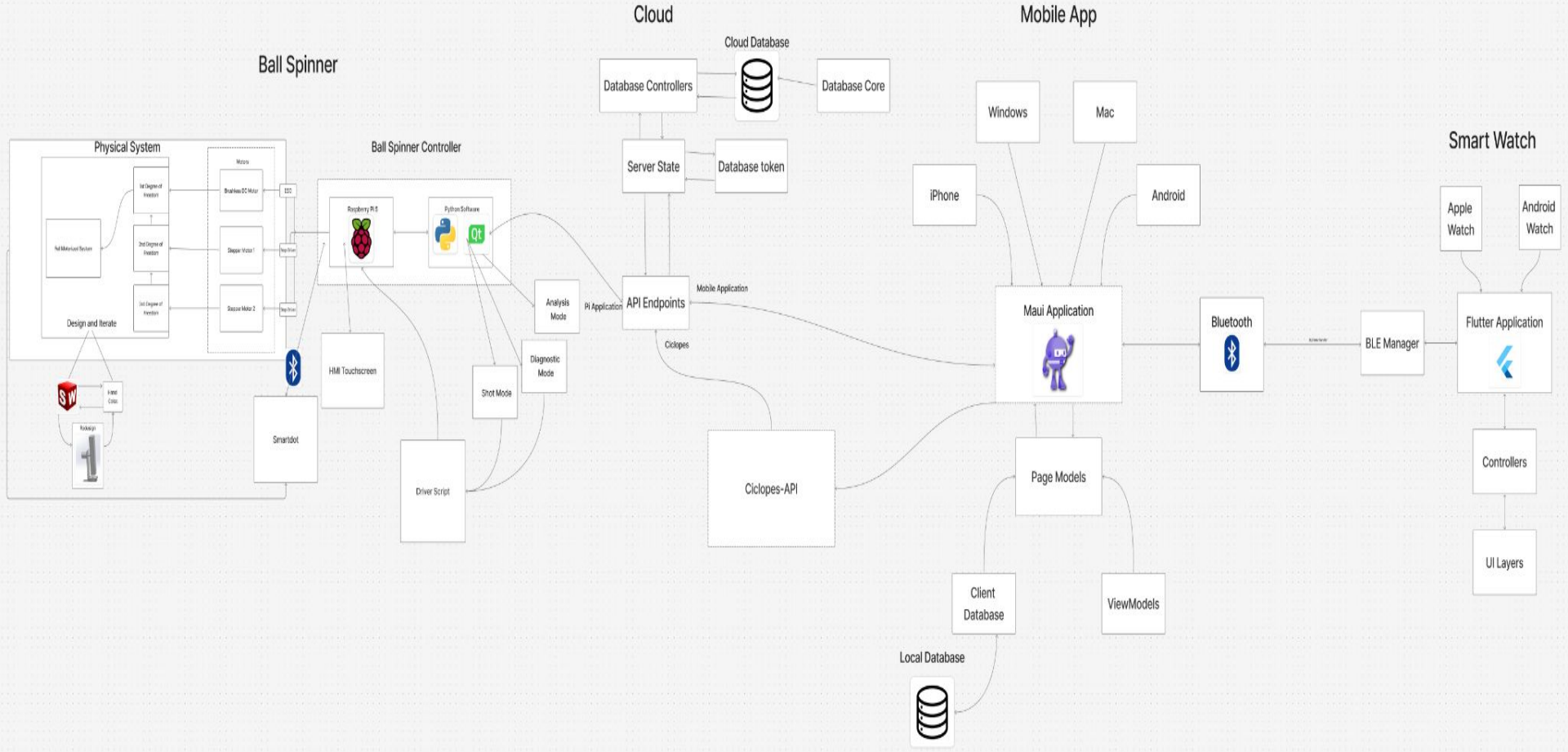
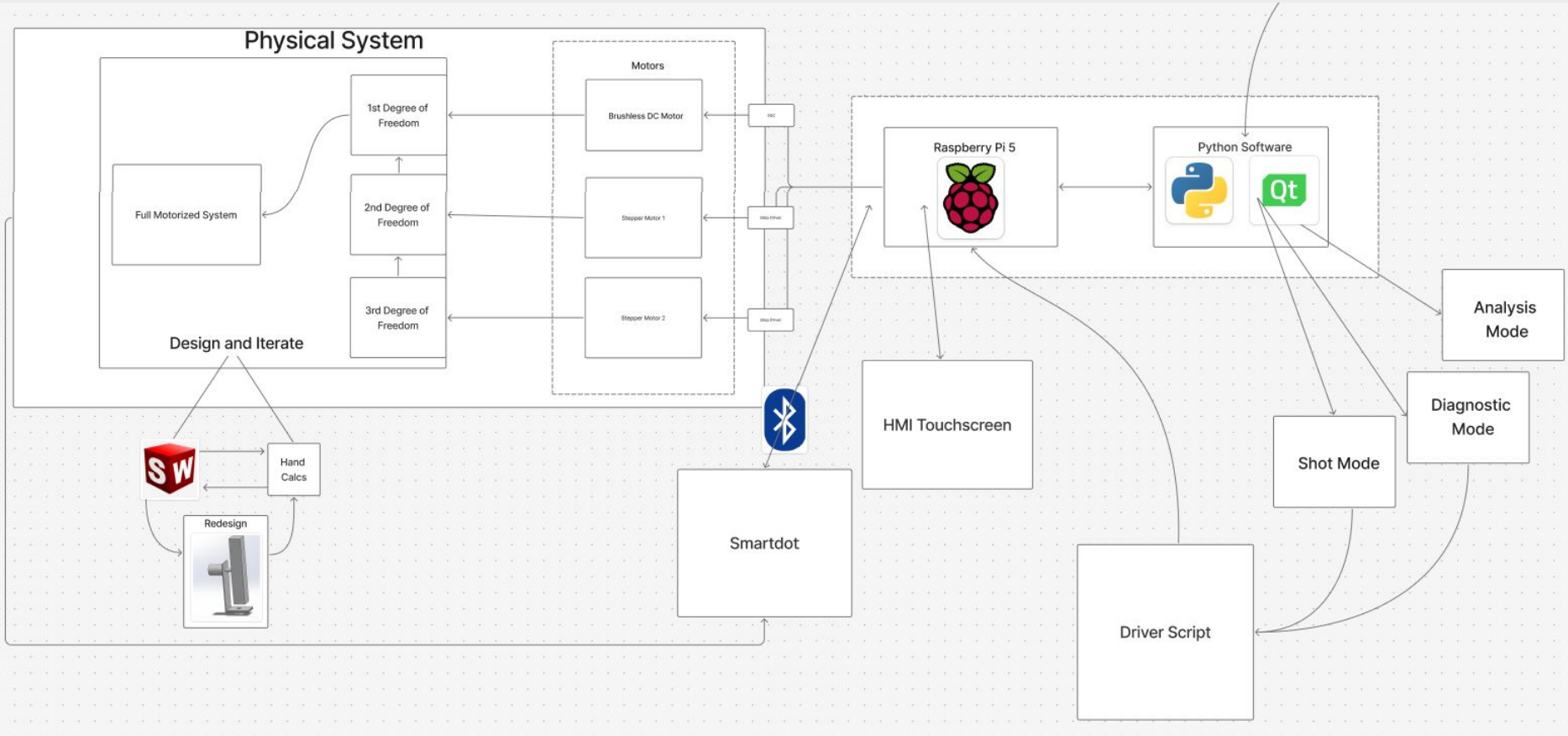
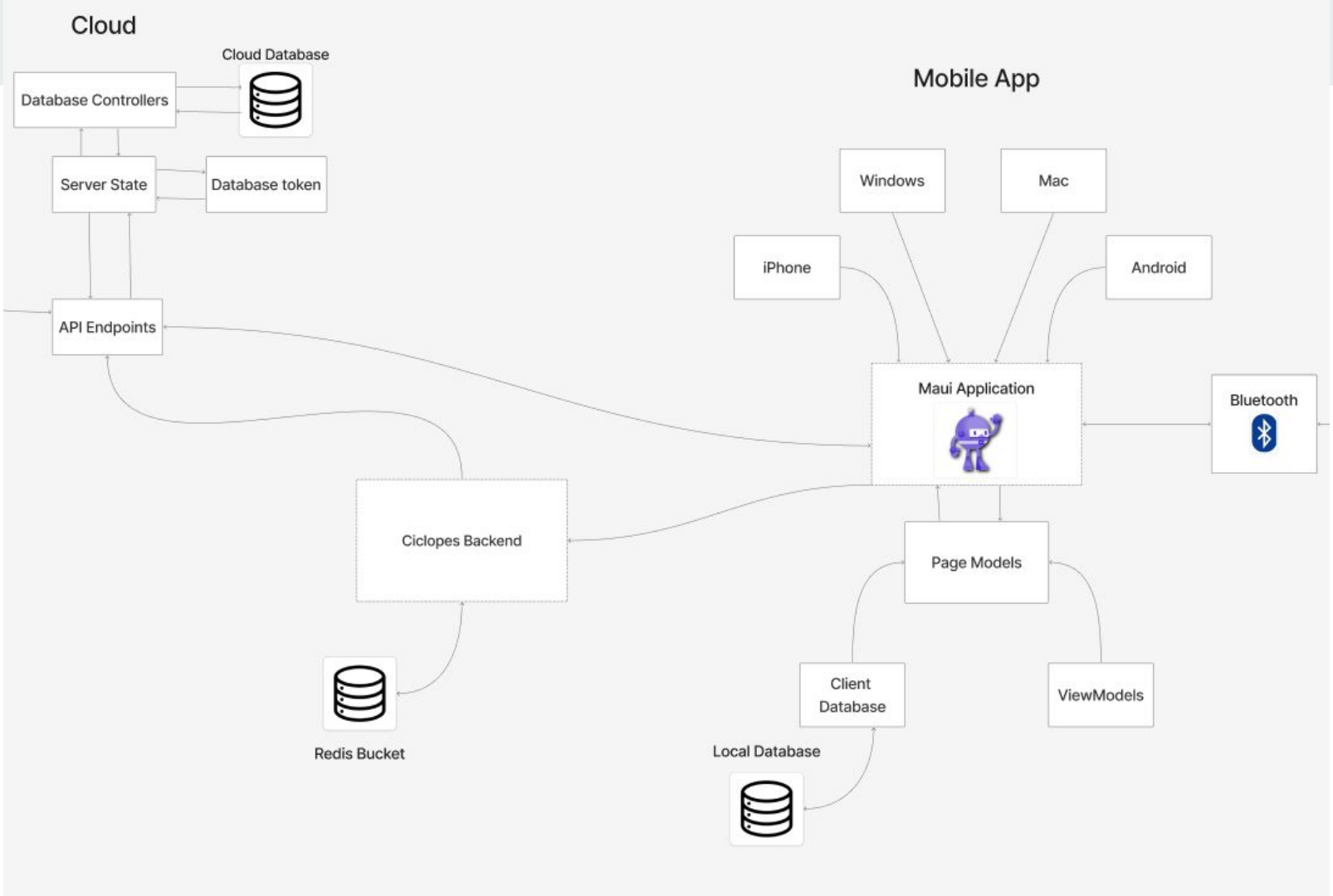
Milestone 2

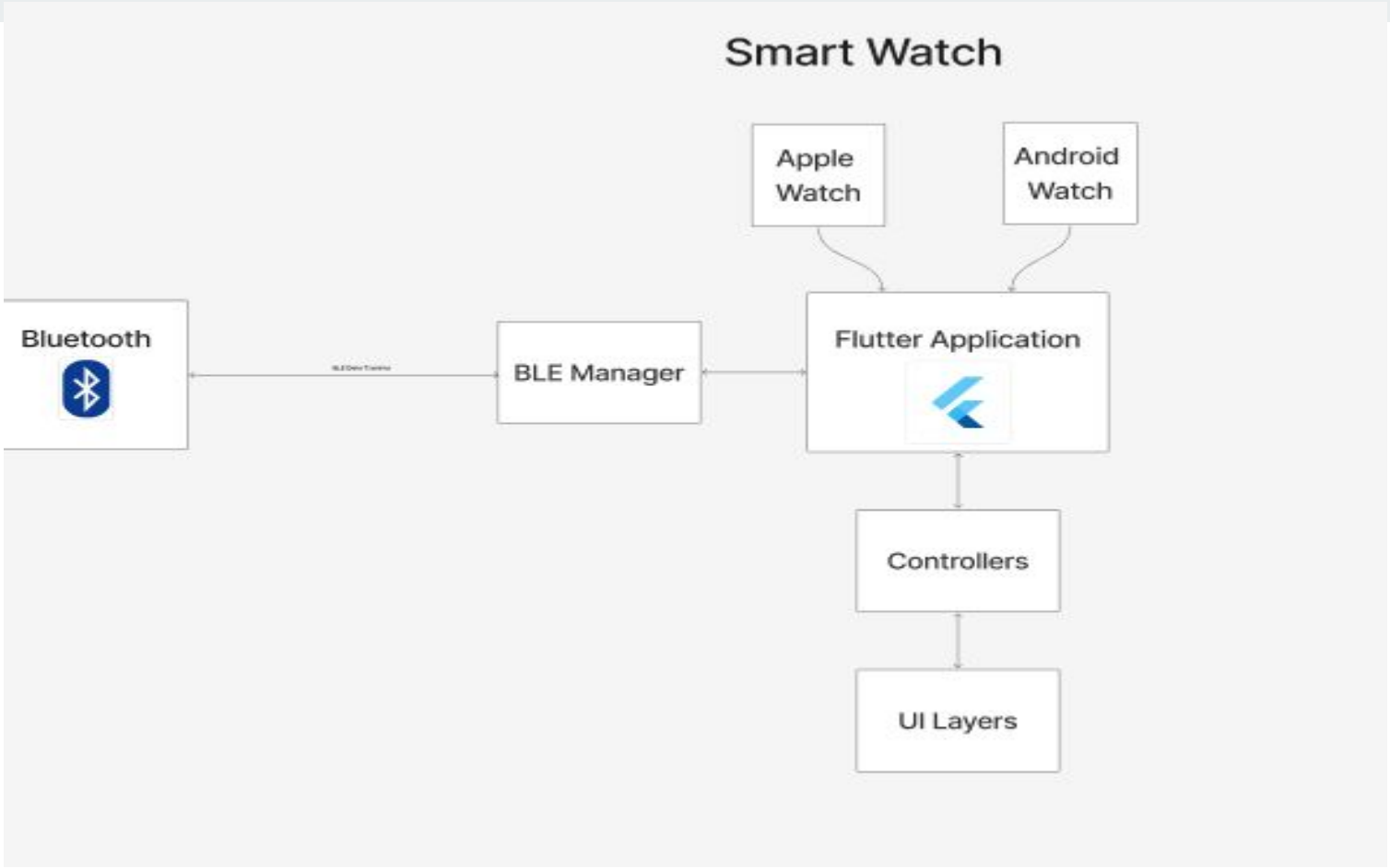
Matt Brown, Josh Byers, Charles Carroll, Zach Cox, Joseph Downey, Gabe Manero, Jakeb Nielsen, Andrew Olvera, Gavin Wentz, Hunter Wolfe

HIGH LEVEL OVERVIEW



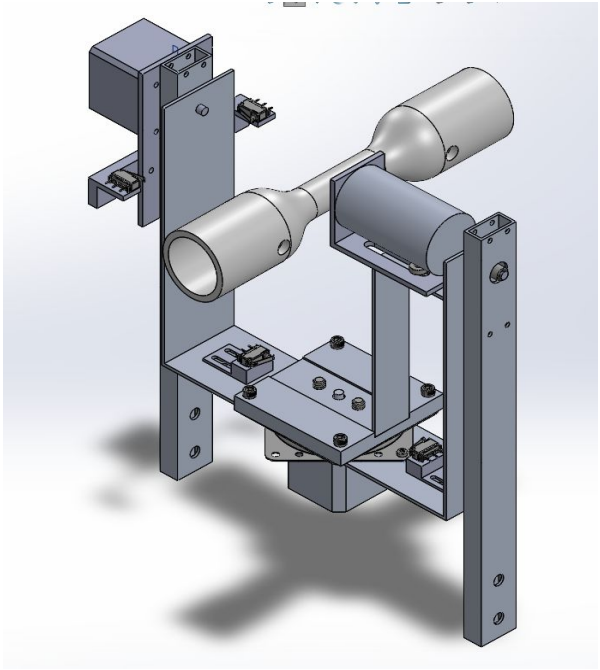

Physical System
Motors
1st Degree of Freedom
2nd Degree of Freedom
3rd Degree of Freedom
Full Motorized System
Design and Iterate
Brushless DC Motor
Stepper Motor 1
Stepper Motor 2
Raspberry Pi 5
Python Software
Qt
Analysis Mode
Diagnostic Mode
Shot Mode
HMI Touchscreen
Smartdot
Driver Script
Hand Calcs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

Physical Design Overview

DXF file of U Bracket and L-Bracket for waterjet cutting



Final Design for Ball Spinner

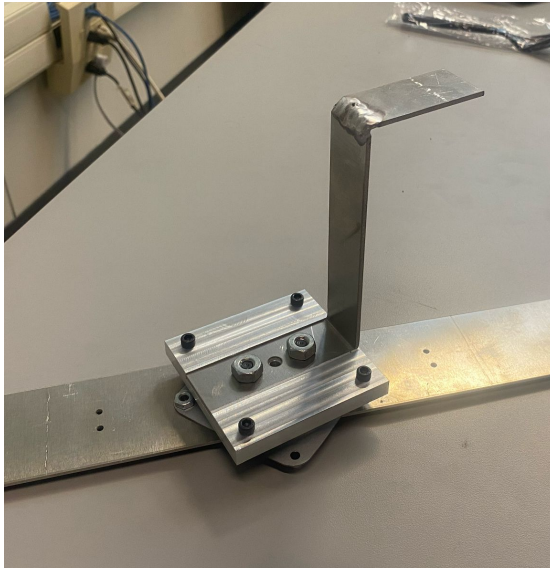
Physical Design Team (Goals and Achievements for MS2)

Goals

- Continue work on enclosure (Bottom panel and acrylic box mounting)
- Completed 1st and 2nd degree assembly
- Ensure assembly is ready for motor mounting

Achievements

- L-bracket, U-bracket, and Rotational Plate fabricated
- 1st and 2nd Degree assembled
- Assembly ready for motors (2nd degree)



Fully aluminum first and second degree with holes ready for limit switch mounts

Physical Design Team (Future: MS3)

Goals

- Fabricate and assemble 3rd Degree of Freedom
- Finish testing to ensure motors meet required acceleration
- Limit switches implemented

Team Pi Demo

Team Pi (Goals for MS2)

- Add encoders to the 2 stepper motors
- Finalize the wavelet implementation within the analysis mode page
- Build standalone app
- Adding ENA connection for both stepper drivers to drop current
- Switching to UART comms for VESC
- PCB routing and order
- Integrate E Stop LED indicator
- Investigate power up sequencing of current system

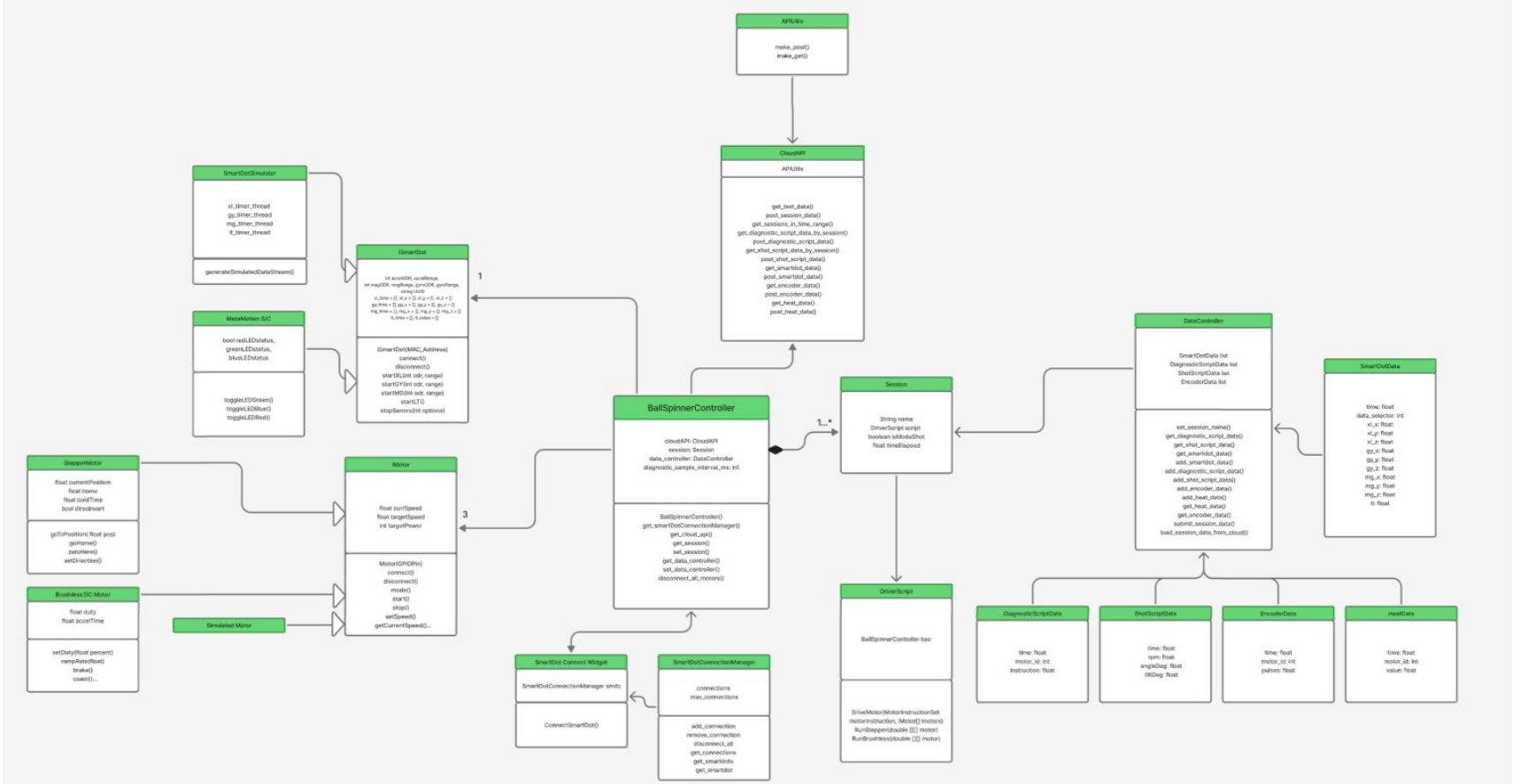
Team Pi (MS2 Achievements)

- Added shot editing capability to the data view page
- Implemented a PID system to drive primary motor
- Added a motor characterization test to the BSC
- PCB routing done. DMF analysis done. Waiting for order confirmation
- Enable pin integrated for the stepper motors
- Integrated E Stop LED indicator
- Found power up sequencing of 2nd and 3rd degrees

Team Pi (Future: MS3)

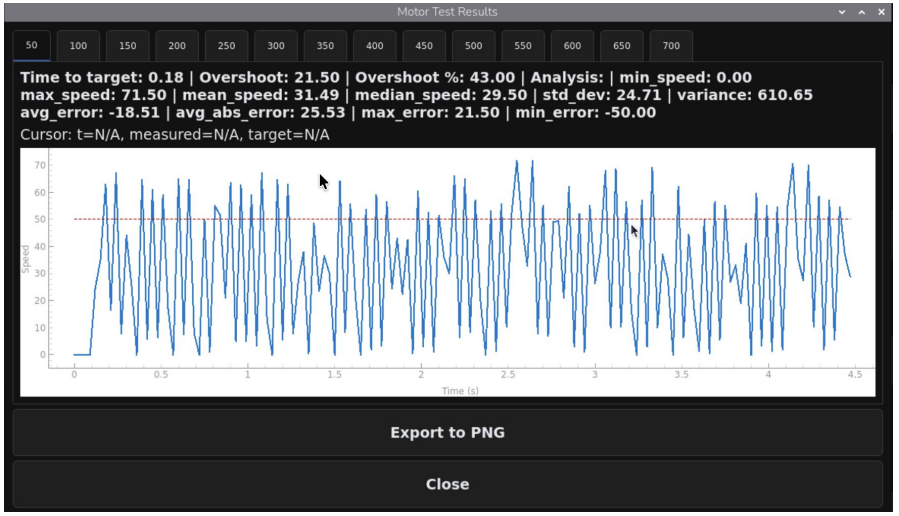
- Add encoders to the secondary and tertiary motors
- Get a releasable standalone version of the BSC application
- Integrate PCB with system
- Rewire the BSC
- Place BSC in enclosure
- Integrate new panel mount LEDs
- Organization of system circuitry
- Investigate power up sequencing after PCB implementation

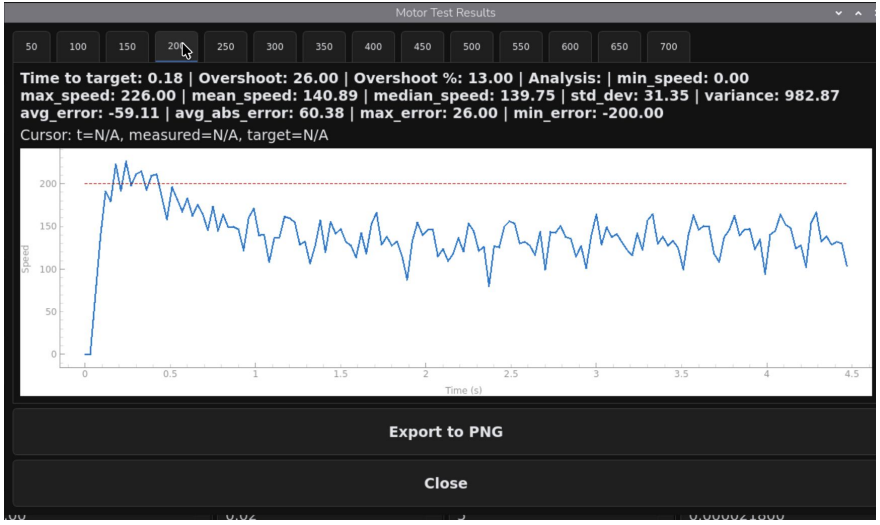
Team Pi UML

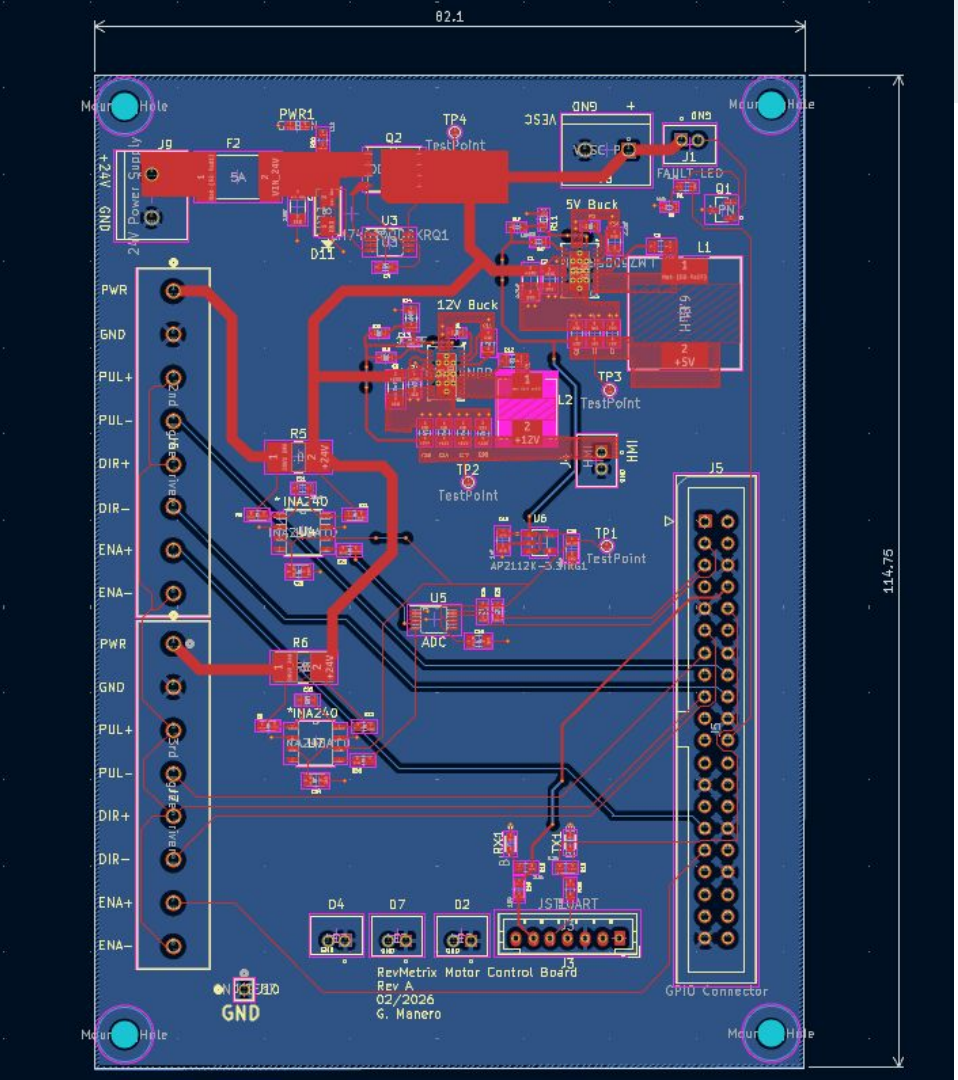

APIUtils
make_post()
make_get()
CloudAPI
APIUtils
get_test_data()
post_session_data()
get_sessions_in_time_range()
get_diagnostic_script_data_by_session()
post_diagnostic_script_data()
get_shot_script_data_by_session()
post_shot_script_data()
get_smartdot_data()
post_smartdot_data()
get_encoder_data()
post_encoder_data()
get_heat_data()
post_heat_data()
SmartDotSimulator
xl_timer_thread
gy_timer_thread
mg_timer_thread
ft_timer_thread
generateSimulatedDataStream()
MetaMotion S/C
bool redLEDstatus,
greenLEDstatus,
blueLEDstatus
toggleLEDGreen()
toggleLEDBlue()
toggleLEDRed()
iSmartDot
iSmartDot(MAC_Address)
connect()
disconnect()
startXL(int odr, range)
startGY(int odr, range)
startMG(int odr, range)
startLT()
stopSensors(int options)
1
BallSpinnerController
cloudAPI: CloudAPI
session: Session
data_controller: DataController
diagnostic_sample_interval_ms: int
BallSpinnerController()
get_smartDotConnectionManager()
get_cloud_api()
get_session()
set_session()
get_data_controller()
set_data_controller()
disconnect_all_motors()
1...*
Session
String name
DriverScript script
boolean isModeShot
float timeElapsed
DataController
SmartDotData list
DiagnosticScriptData list
ShotScriptData list
EncoderData list
set_session_name()
get_diagnostic_script_data()
get_shot_script_data()
get_smartdot_data()
add_smartdot_data()
add_diagnostic_script_data()
add_shot_script_data()
add_encoder_data()
add_heat_data()
get_heat_data()
get_encoder_data()
submit_session_data()
load_session_data_from_cloud()
SmartDotData
time: float
data_selector: int
xl_x: float
xl_y: float
xl_z: float
gy_x: float
gy_y: float
gy_z: float
mg_x: float
mg_y: float
mg_z: float
ft: float
SteppperMotor
float currentPosition
float rtheme
float toldTime
bool directwart
getToPosition(float pos)
getHome()
zeroHere()
setDirection()
Brushless DC Motor
float duty
float accelTime
setDuty(float percent)
rampRate(float)
brake()
coast()...
Simulated Motor
IMotor
float currSpeed
float targetSpeed
int targetPower
Motor(GPIOPin)
connect()
disconnect()
mode()
start()
stop()
setSpeed()
getCurrentSpeed()...
3
SmartDot Connect Widget
SmartDotConnectionManager smdc
ConnectSmartDot()
SmartDotConnectionManager
connections
max_connections
add_connection
remove_connection
disconnect_all
get_connections
get_smartdots
get_smartdot
DriverScript
BallSpinnerController bsc
DriveMotor(MotorInstructionSet motorInstruction, IMotor[] motors)
RunStepper(double [][] motor)
RunBrushless(double [][] motor)
DiagnosticScriptData
time: float
motor_id: int
instruction: float
ShotScriptData
time: float
rpm: float
angleDeg: float
tiltDeg: float
EncoderData
time: float
motor_id: int
pulses: float
HeatData
time: float
motor_id: int
value: float

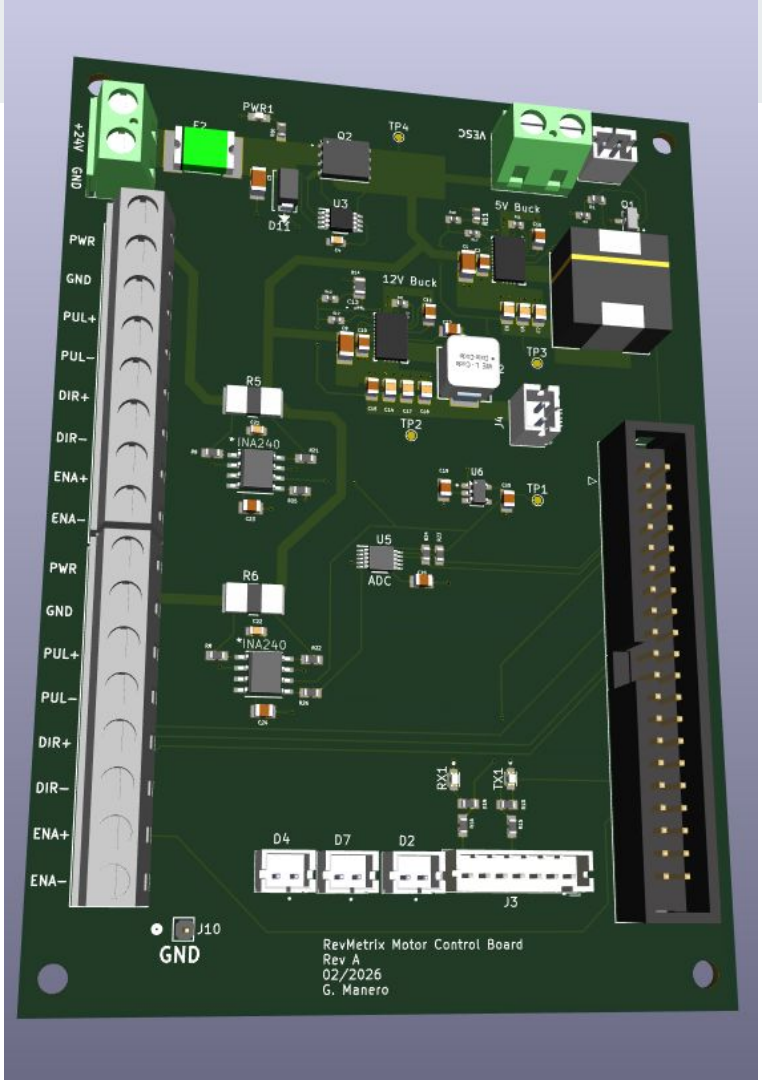
PID Motor Control

```python
if self.currSpeed < min(self.targetSpeed**2/900, self.targetSpeed) and self.targetSpeed != 0:
    # Motor is stopped give big kick to get it going, then let PID take over
    command = 1/self.duty_cycle_scale  # Garbage for logging purposes since we're not really usi
    duty = min(self.targetSpeed/6000, 1.0)  # run at 100% duty until we get a speed reading, the
else:
    command = (
        self.targetSpeed
        + (error * self.Kp)
        + (self._integral * self.Ki)
        + (d_error * self.Kd)
    )
    duty = self.clamp(command * self.duty_cycle_scale, 0.0, 1.0)
```

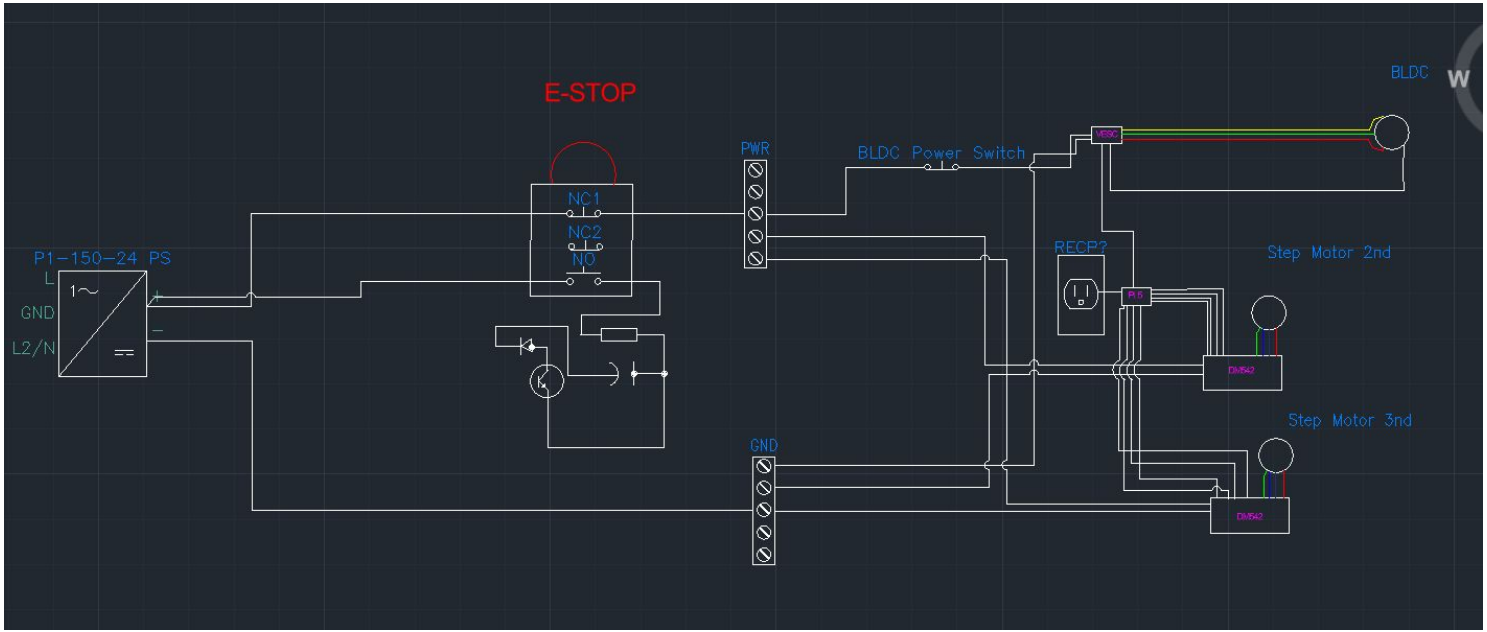





82.1
114.75
Mounting Hole
PWR1
TP4
TestPoint
Q2
J9
F2
5A
24V Power Supply
+24V
GND
VESC
J1
FAULT LED
Q1
5V Buck
U3
D11
L1
12V Buck
TP3
L2
TP2
HMI
R5
INA240
J5
U6
AP2112K-3.3TRG1
TP1
U5
ADC
R6
PWR
PUL+
PUL-
DIR+
DIR-
ENA+
ENA-
RX1
TX1
D4
D7
D2
JST UART
J3
J10
RevMetrix Motor Control Board
Rev A
02/2026
G. Manero
GPIO Connector

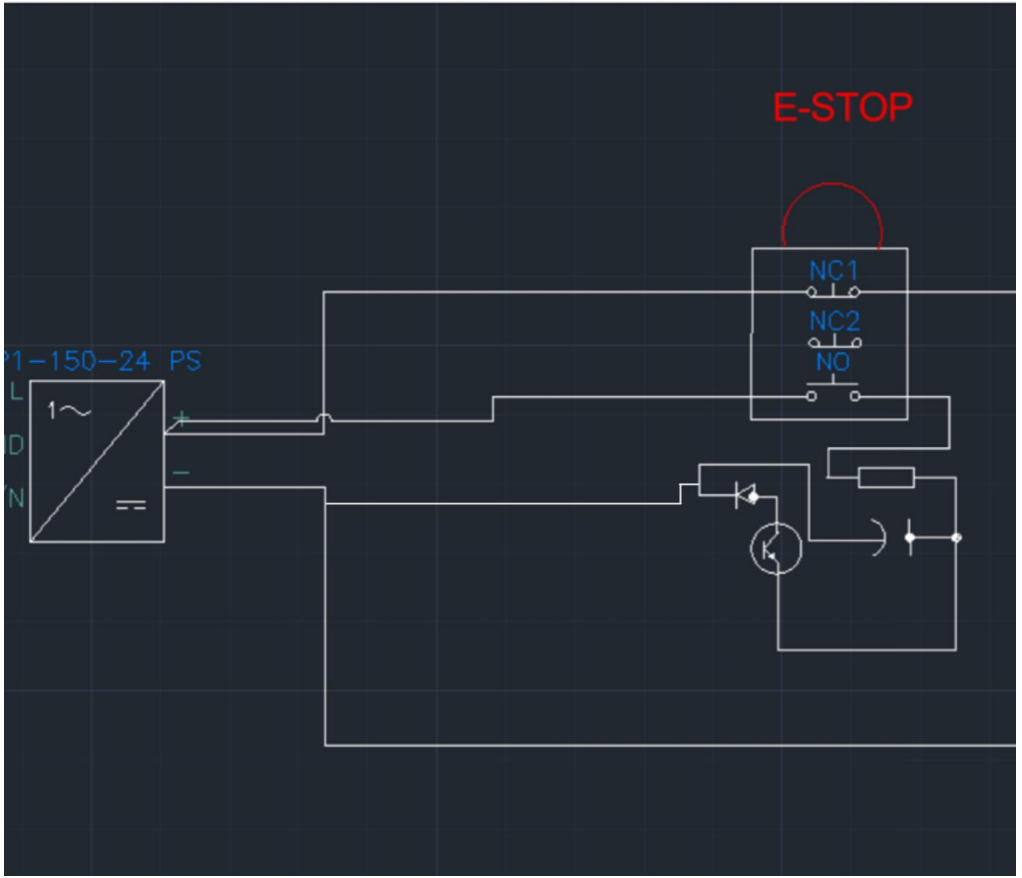

PWR1
TP4
Q2
VESC
+24V
GND
D11
U3
5V Buck
Q1
12V Buck
TP3
TP2
R5
INA240
U6
TP1
U5
ADC
R6
PWR
PUL+
PUL-
DIR+
DIR-
ENA+
ENA-
RX1
TX1
D4
D7
D2
J3
J10
GND
RevMetrix Motor Control Board
Rev A
02/2026
G. Manero

Team Pi Current Electrical System

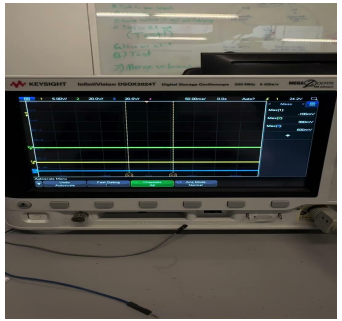


E-STOP Updated

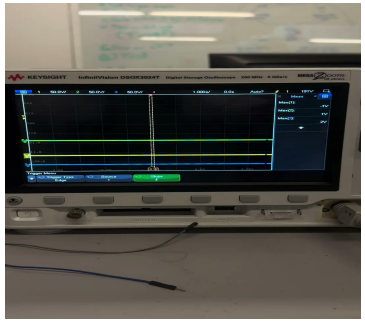
https://youtube.com/shorts/z9sWY7U_5AA?si=pulk5aZ2aARLP9XW



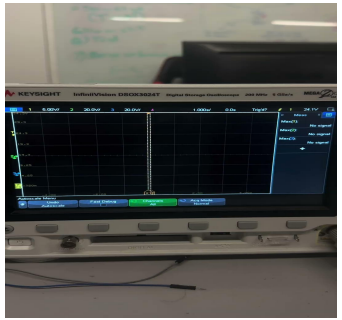
Power sequencing



Video 1



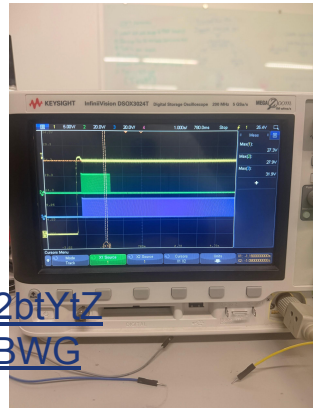
Video 2

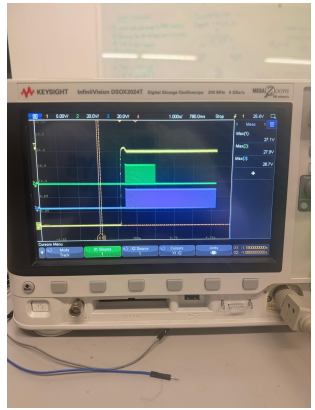


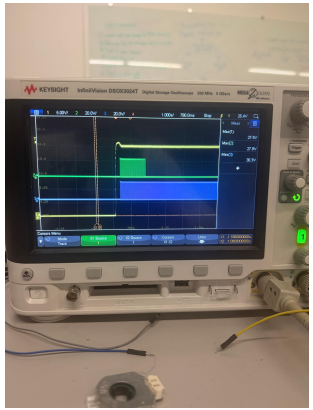
Video 3

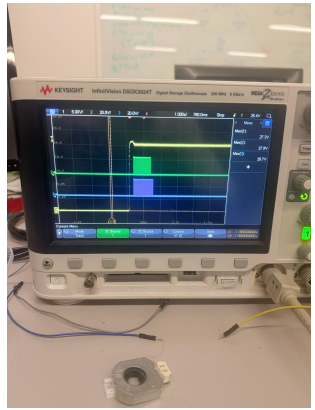
Video 4

https://youtube.com/shorts/c2btYtZEn3g?si=H5MURL9pOc5wyBWG

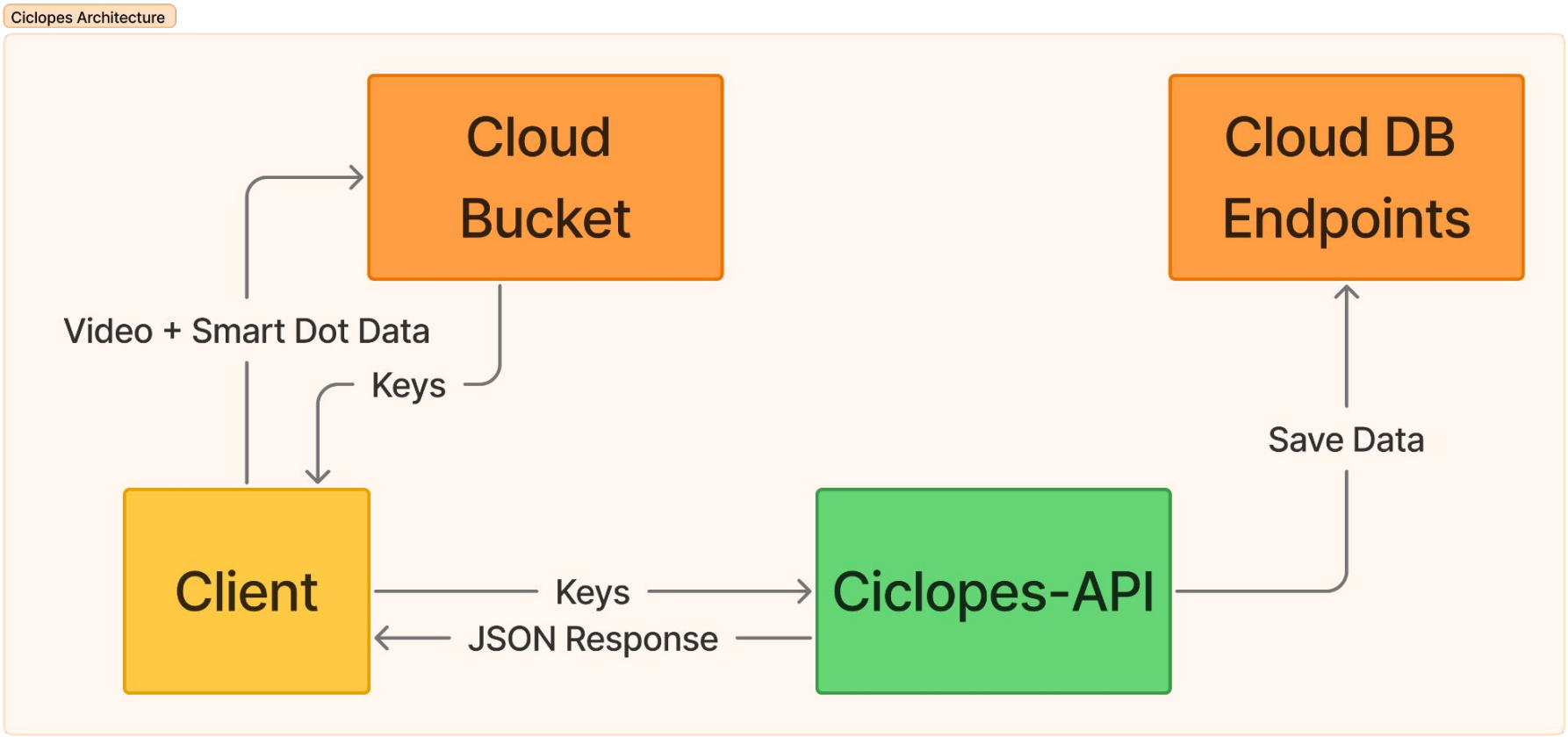




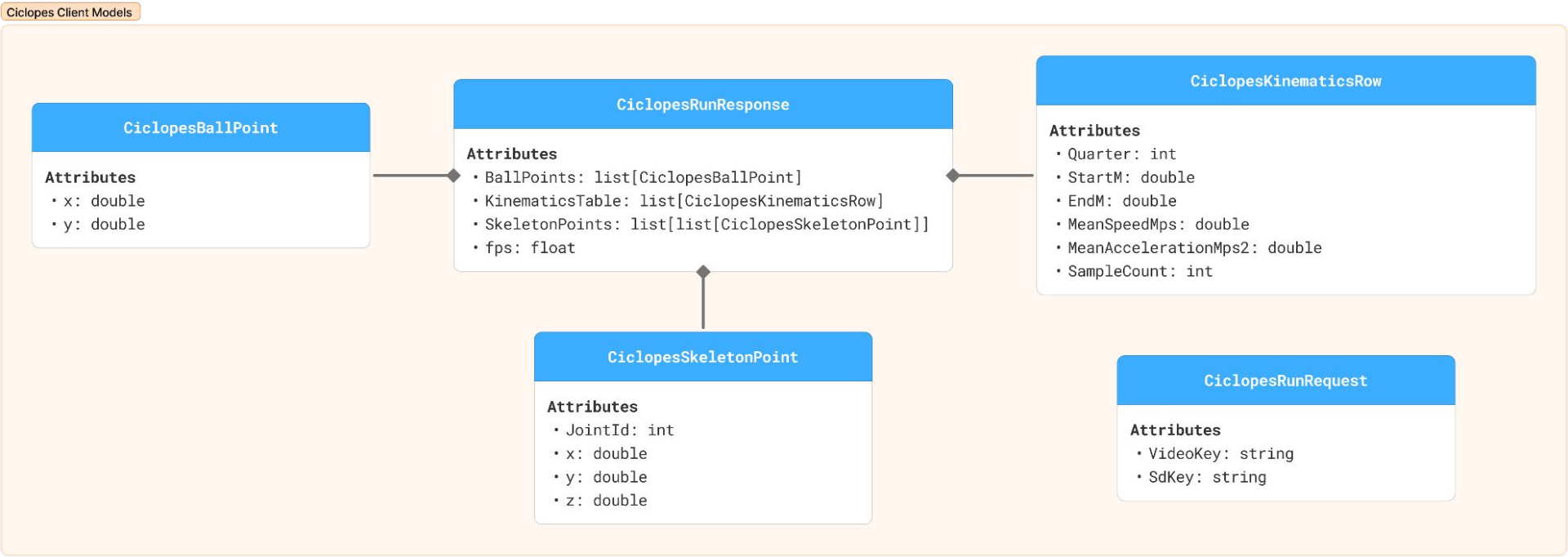




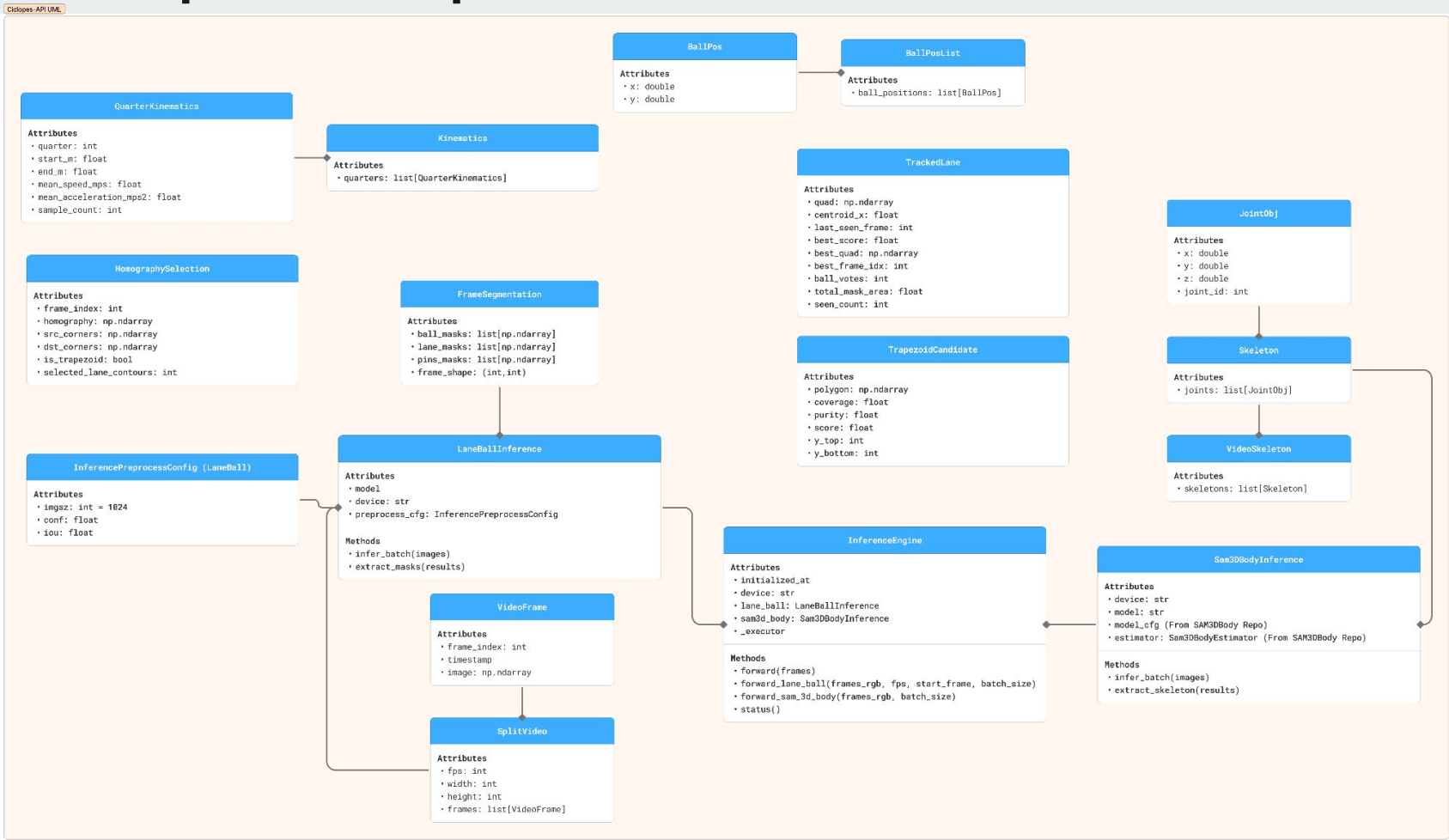
Ciclopes - Overview



Ciclopes - Client Side Models



Ciclopes - Ciclopes-API UML / Architecture



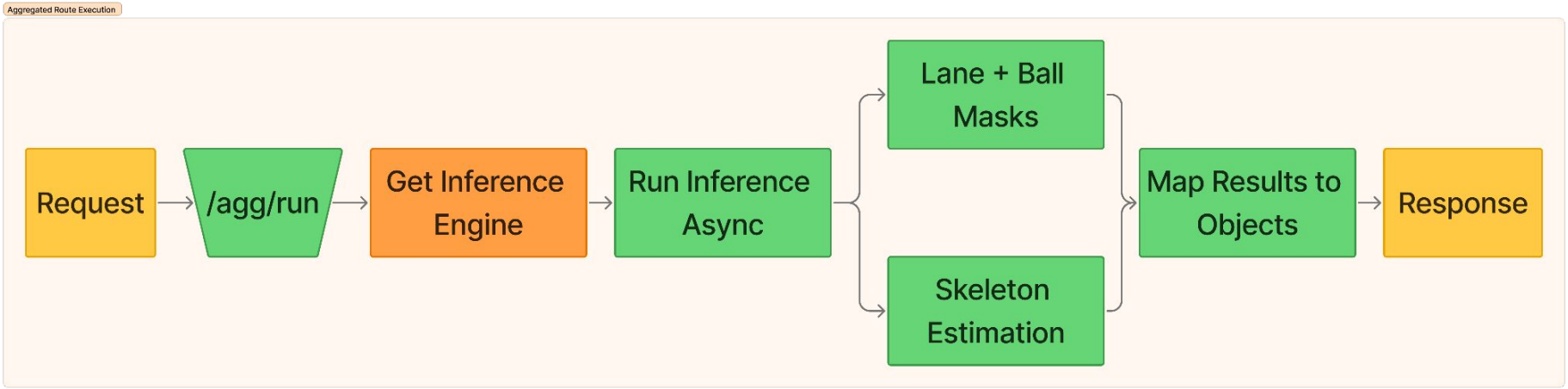
Ciclopes - Goals for MS2

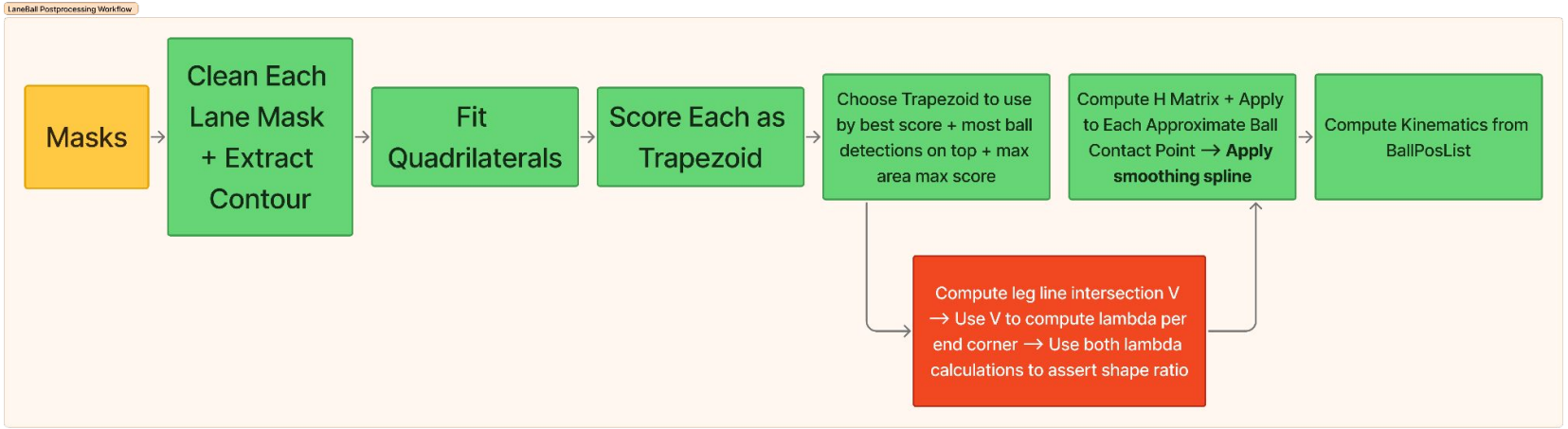
- Integrate fully into client application
  - Improve upon UI for better user experience
  - Improved data visualization for interpretability
- Utilize smart dot data for release and ground point detection
  - PoseEstimation from average approach time before release and a few frames after
  - LaneBall from ground contact after
- Non blocking inference of SAM3D Body
  - Separate API calls for LaneBall and SAM3D Body – LaneBall will finish much faster
- Implement edge case handling
  - Ball leaving lane
  - Many frames without ball detection
- Real world test
- Stretch Goal: Prototype on device inference for one architecture
  - Experiment with camera setup workflow for immediate user feedback

Ciclopes - Accomplishments for MS2

- Integrate fully into client application
  - Improve upon UI for better user experience
  - Improved data visualization for interpretability
- Fixed ball position render and added smoothing
  - Interpolation using a smoothing spline
- Implemented estimated pose rendering with EMA smoothing
- Non blocking inference of SAM3D Body
- Implement edge case handling
  - Extrapolation using expected next step distance from running velocity
  - Ball leaving lane, and many frames without ball detection
- Designed DB schema to save ball positions, kinematics table, and rendering vectors

Ciclopes - Ciclopes-API Execution





Ciclopes - DB Schema

Table: skeleton_joints

skeleton_id	joint_id	x	y	z

Table: skeleton_frames

skeleton_id	shot_id	frame_idx	fps

Table: ball_frames

ball_point_id	shot_id	frame_idx	fps

Table: ball_points

ball_point_id	x	y

Ciclopes - Original Video

Ciclopes - Demo

Ciclopes - Goals for MS3

- Implement DB integration
- Add query Ciclopes from shot statistics views
    - Overlay multiple shots and data analysis view (connect with user collected scores / outcomes) - Stretch: AI Insights (LLM Powered but not AI slop)
- Implement SmartDot data integration
    - Record real world use demo (Setup service from Capstone room machine)
- Integrate UI styling + component library (ShadCn.Blazor)
    - UI components + styling build system
- Finalize UI/UX design with team and overhaul UX
- Complete integration into application

Wiki

MS2 Goals

- **Continue updating current state of the project**

MS2 Achievements

- **Updated technologies used for last semester**
- **Added a watch application section**

Future

- **Continue updating mobile progress**
- **Get help updating other parts of the project**

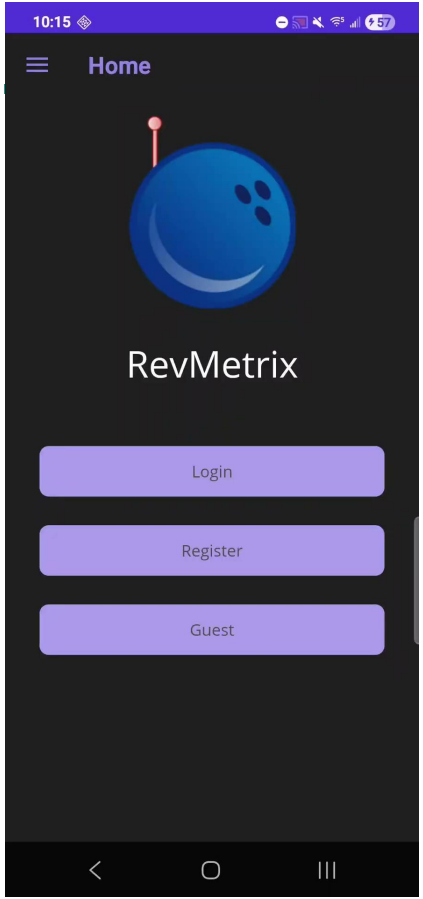
SMARTWATCH APPLICATION

Smartwatch Application

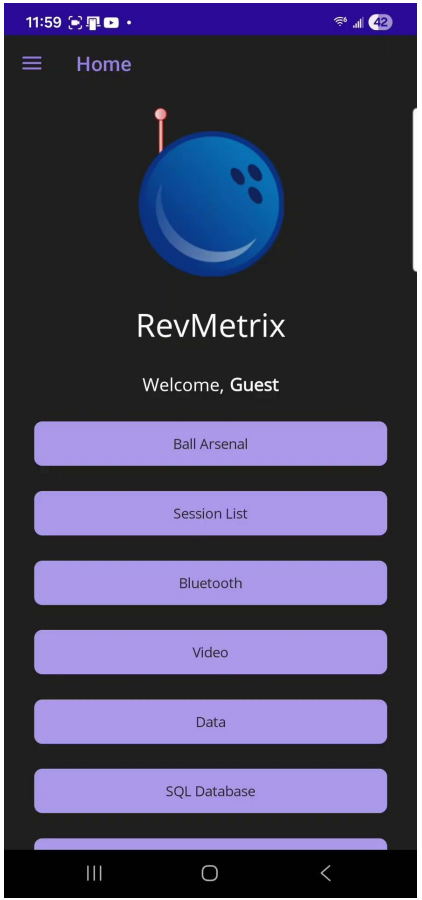
The Smartwatch application has been created as a companion to the mobile application. It is meant to be used with Ciclopes in the mobile application. Since users will be using their phone to record the lane, the smartwatch application will allow them to input their shots simultaneously.

- BLE Manager
  BLE Manager
- Local Cache
  Local Cache
- Session Controller
  Session Controller
- Watch Models
  Watch Models
- Dev Settings
  Dev Settings Page
- Shot Page
  Shot input
- Frame Page
  Frame Page
- Game Page
  Game page

Mobile App - Demo!



Video page



Mobile App (MS2 Goals)

- Improve Stats Page
- Add requested features to Ball Arsenal and Event/Establishment creation
- Create Cloud Sync Feature
- Improve Ciclopes
- Improve Bluetooth to MMS and MMC

Mobile App (MS2 Achievements)

- **Login/Registration Unmask Passwords**
- **Video Page**
  - Updated app to .NET 10
  - Recreated video page from scratch
- **Ball Arsenal**
  - Added missing fields to Ball Class
  - Implemented edit feature
- **Shot Page**
  - Updated edit shots
  - Cleaned up page reloading logic
- **Stats Page**
  - Implemented logic for calculating Splits and Washouts
- **CellularCore/Unit Testing**
  - CalculateShotType
  - IsSplit

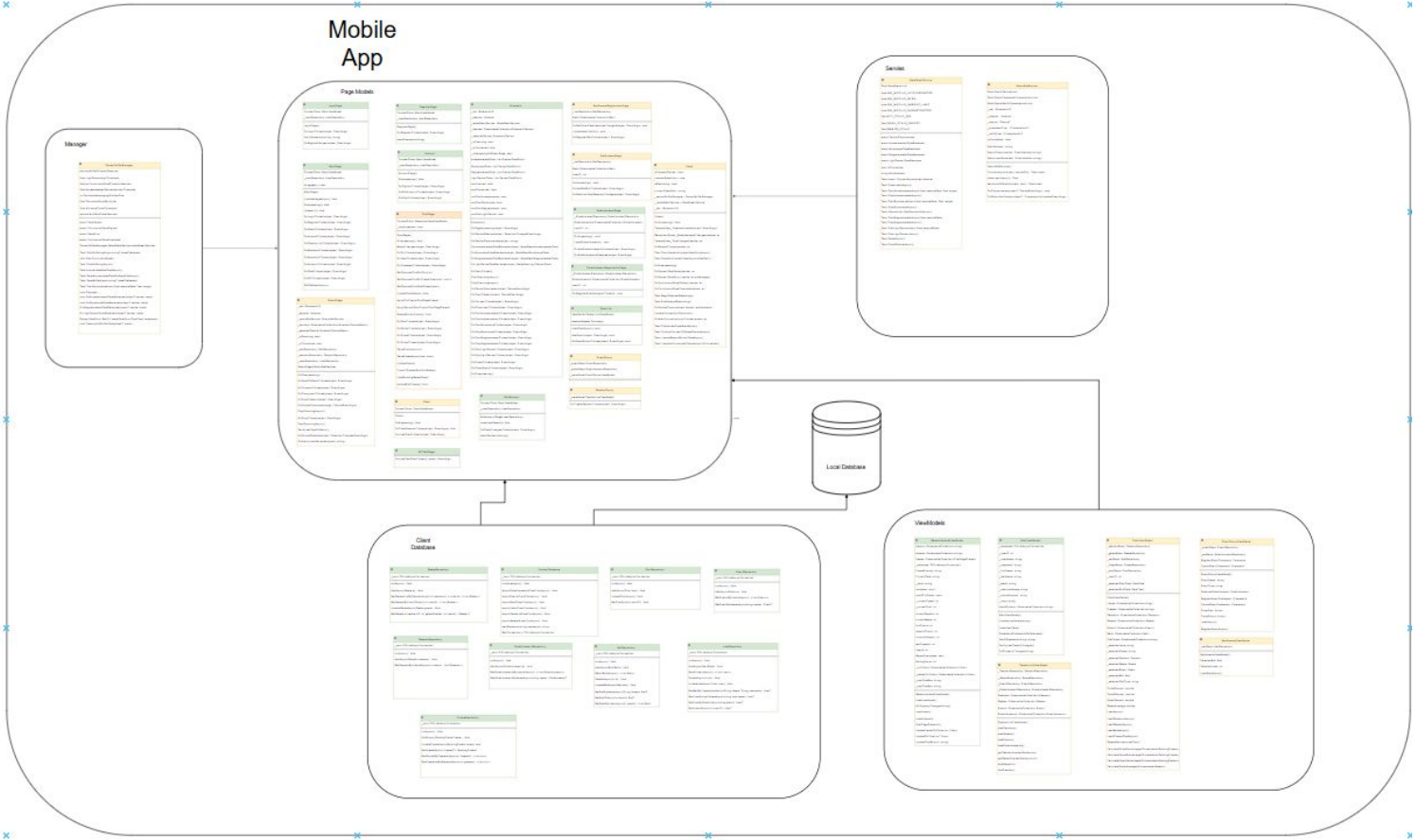
Cloud Syncing Feature

- **Button on Account page to Sync**
- **Allows user to pick either Cloud or Local to keep**
- **Added Support for Gets, Puts, and Deletes**
- **Updated All SQL Models in the App with CloudID**

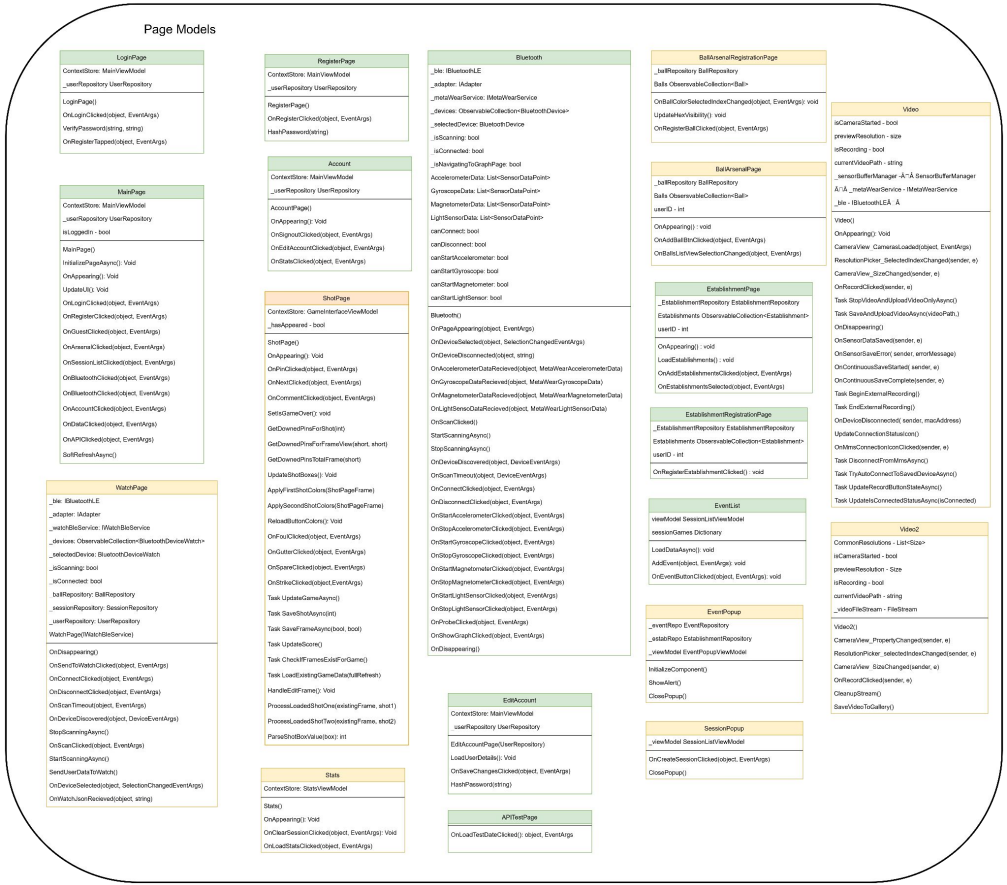
Mobile App (Future: MS3)

- Improve Stats Page/shot input
- Edit/ update events and establishments
- Change app color scheme
- Save shots from watch
- Update MMS page

Mobile UML Overview



Mobile UML - Page Models

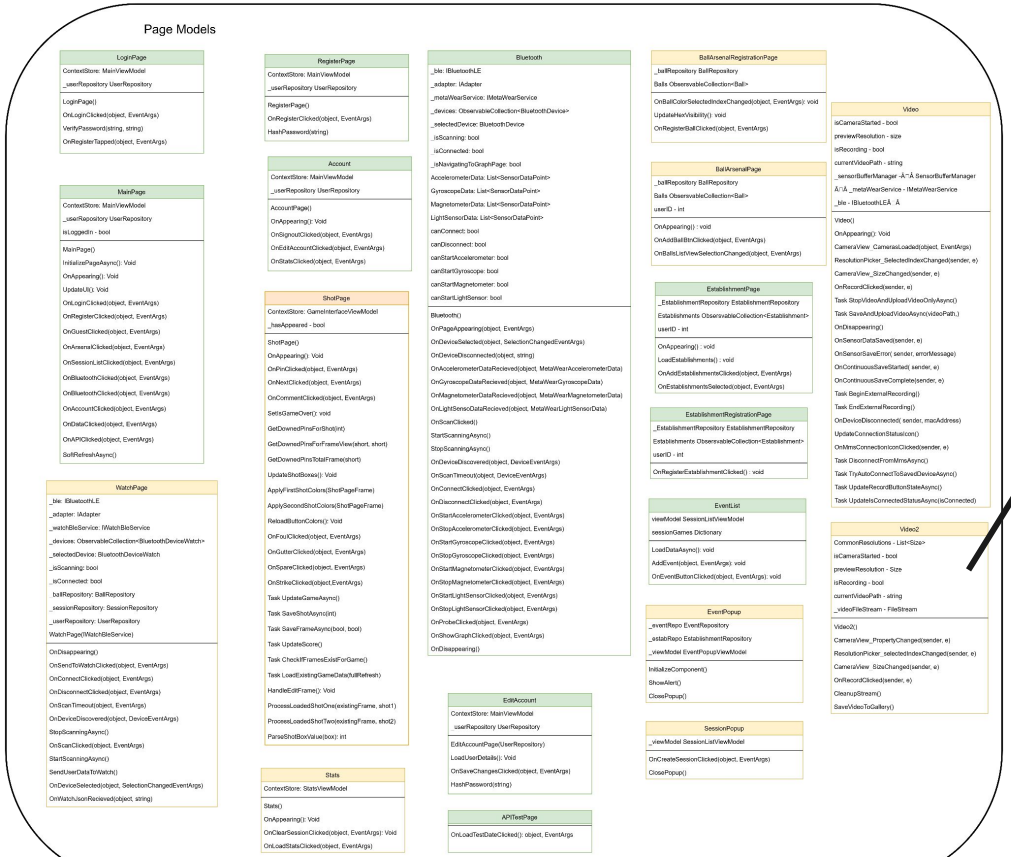


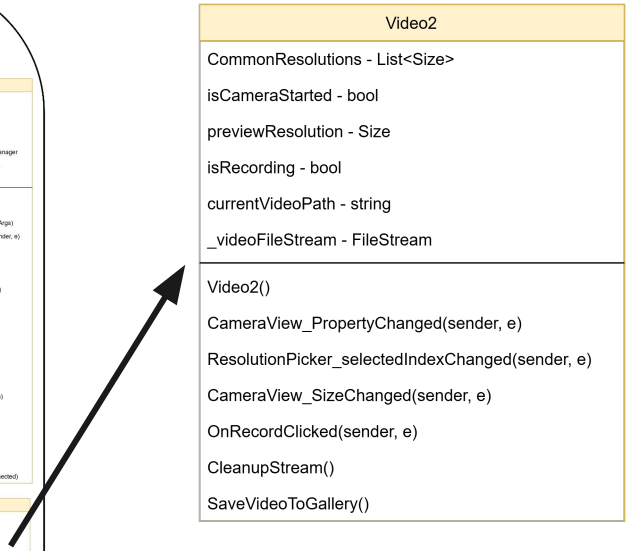
Implemented

Partially Implemented

Not Yet Implemented

Mobile UML - New Video Page



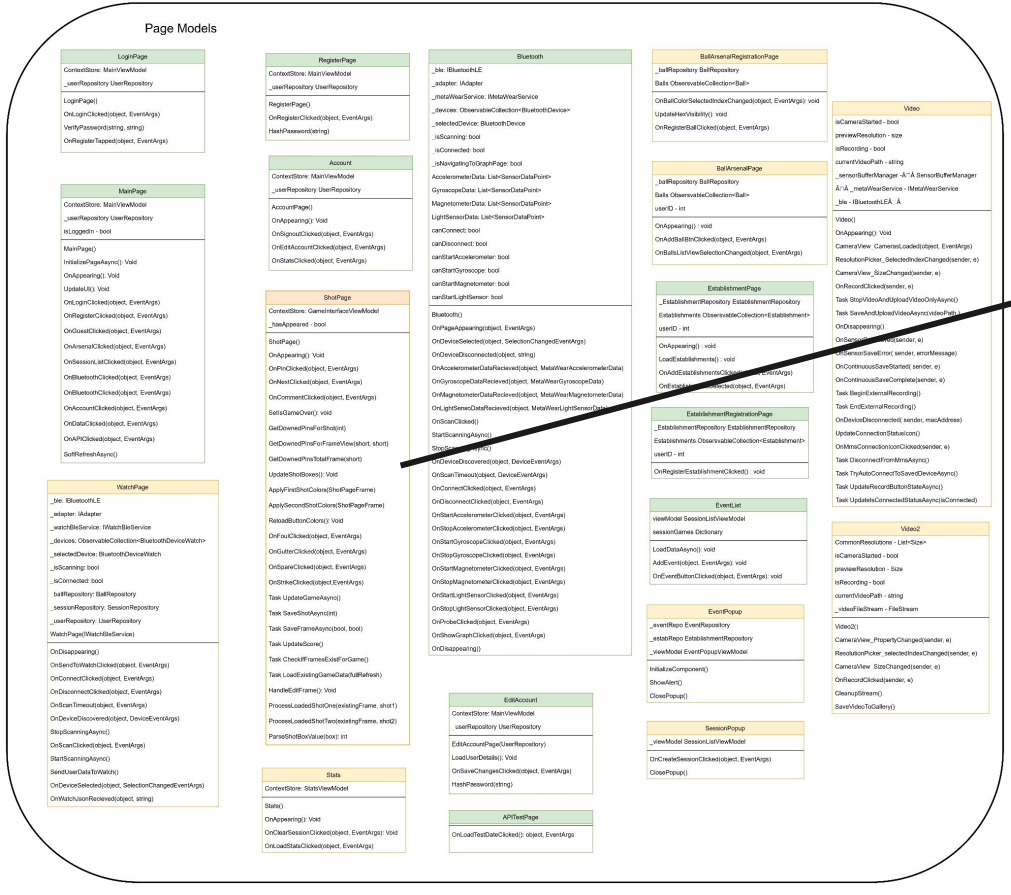


Implemented

Partially Implemented

Not Yet Implemented

Mobile UML - Page Models



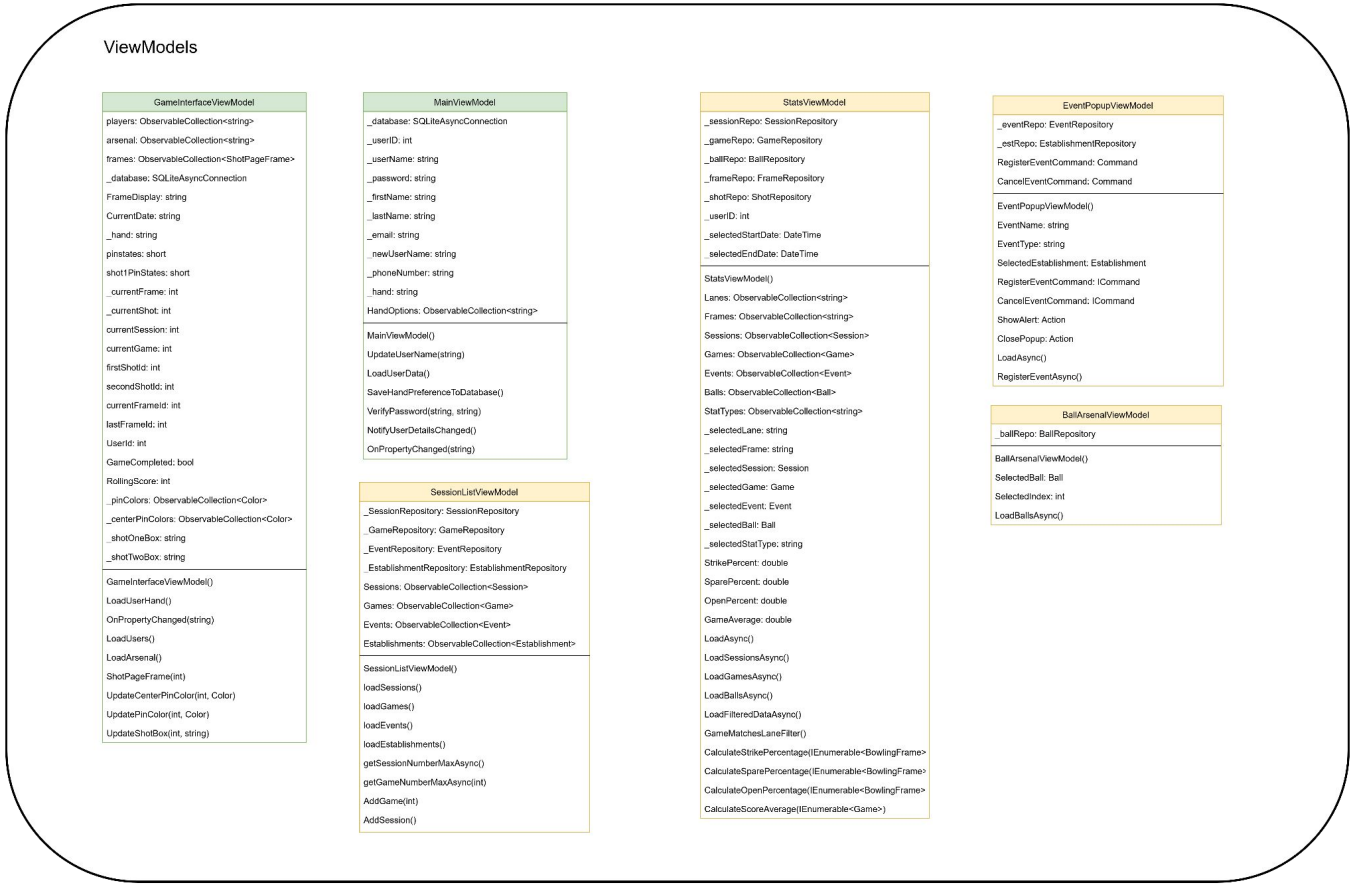
ShotPage
ContextStore: GameInterfaceViewModel
_hasAppeared - bool
ShotPage()
OnAppearing(): Void
OnPinClicked(object, EventArgs)
OnNextClicked(object, EventArgs)
OnCommentClicked(object, EventArgs)
SetIsGameOver(): void
GetDownedPinsForShot(int)
GetDownedPinsForFrameView(short, short)
GetDownedPinsTotalFrame(short)
UpdateShotBoxes(): Void
ApplyFirstShotColors(ShotPageFrame)
ApplySecondShotColors(ShotPageFrame)
ReloadButtonColors(): Void
OnFoulClicked(object, EventArgs)
OnGutterClicked(object, EventArgs)
OnSpareClicked(object, EventArgs)
OnStrikeClicked(object,EventArgs)
Task UpdateGameAsync()
Task SaveShotAsync(int)
Task SaveFrameAsync(bool, bool)
Task UpdateScore()
Task CheckIfFramesExistForGame()
Task LoadExistingGameData(fullRefresh)
HandleEditFrame(): Void
ProcessLoadedShotOne(existingFrame, shot1)
ProcessLoadedShotTwo(existingFrame, shot2)
ParseShotBoxValue(box): int

Implemented

Partially Implemented

Not Yet Implemented

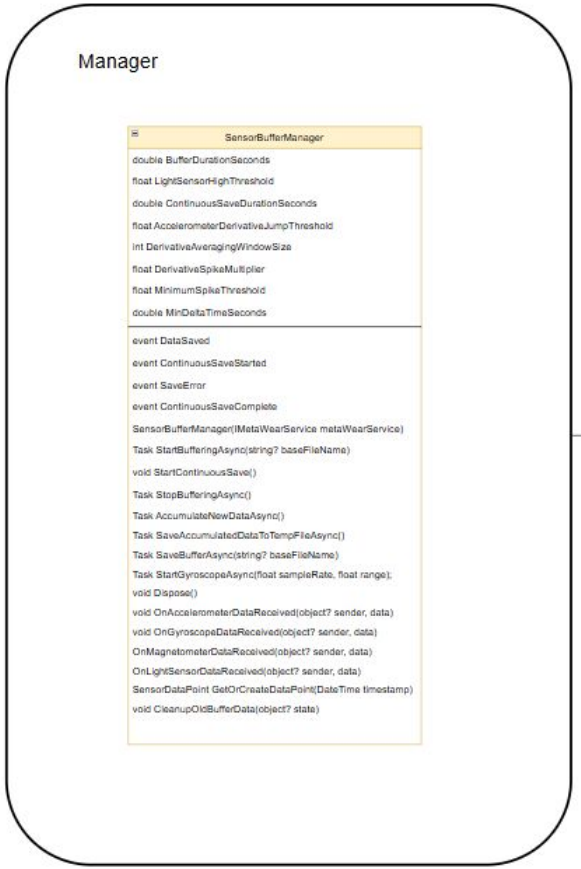
Mobile UML - View Models



Mobile UML - Database

Client
Database

Mobile UML - Managers

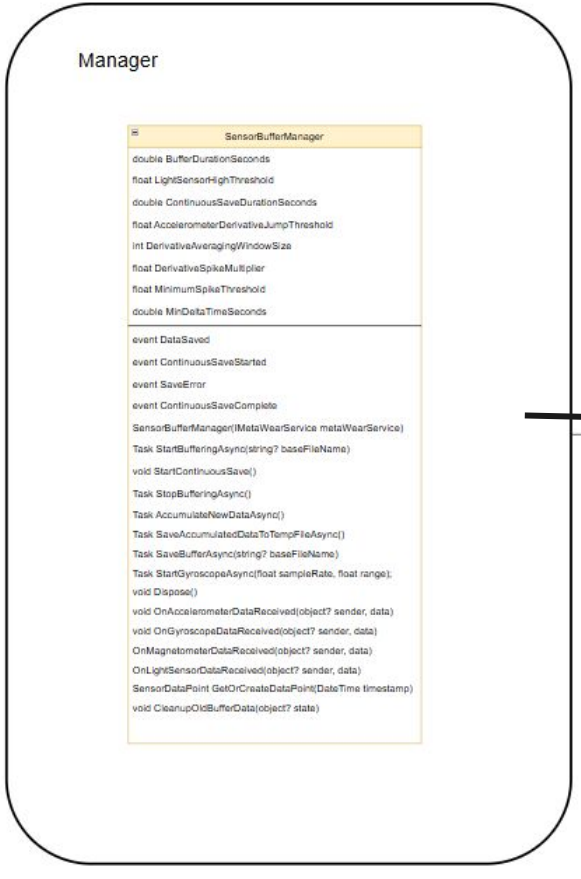


Implemented

Partially Implemented

Not Yet Implemented

Mobile UML - SensorBufferManager

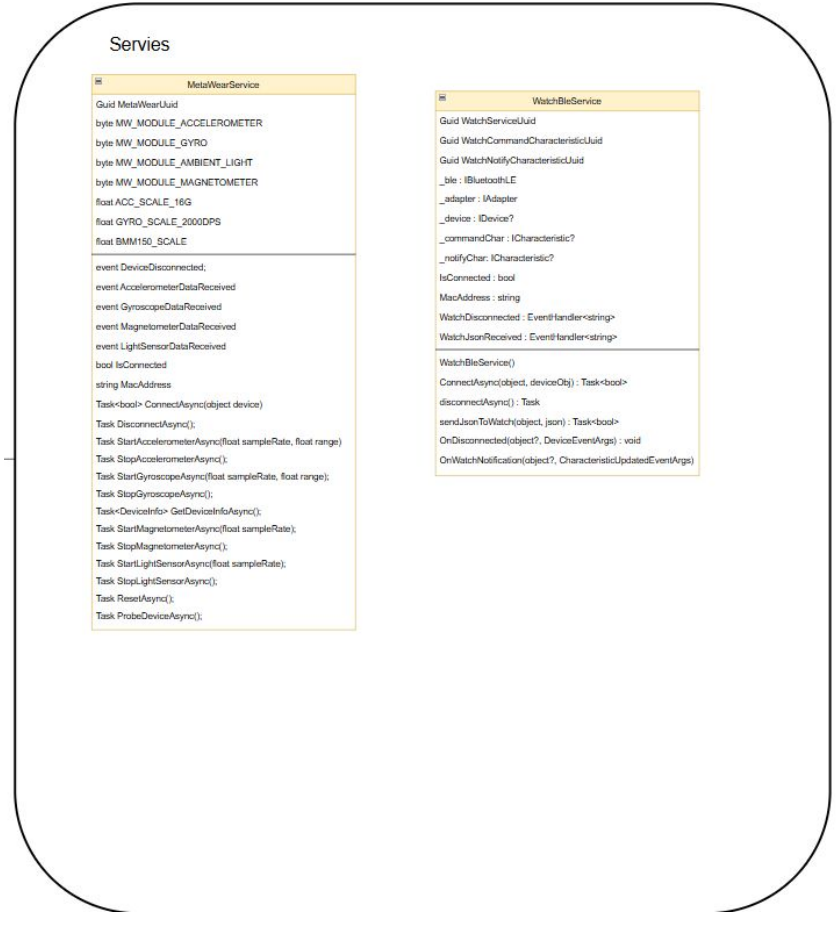


Implemented

Partially Implemented

Not Yet Implemented

Mobile UML - Services



Implemented

Partially Implemented

Not Yet Implemented

Mobile UML - MetaWearService

Servies

MetaWearService

Guid MetaWearUuid
byte MW_MODULE_ACCELEROMETER
byte MW_MODULE_GYRO
byte MW_MODULE_AMBIENT_LIGHT
byte MW_MODULE_MAGNETOMETER
float ACC_SCALE_16G
float GYRO_SCALE_2000DPS
float BMM150_SCALE

event DeviceDisconnected;
event AccelerometerDataReceived
event GyroscopeDataReceived
event MagnetometerDataReceived
event LightSensorDataReceived
bool IsConnected
string MacAddress
Task<bool> ConnectAsync(object device)
Task DisconnectAsync();
Task StartAccelerometerAsync(float sampleRate, float range)
Task StopAccelerometerAsync();
Task StartGyroscopeAsync(float sampleRate, float range);
Task StopGyroscopeAsync();
Task<DeviceInfo> GetDeviceInfoAsync();
Task StartMagnetometerAsync(float sampleRate);
Task StopMagnetometerAsync();
Task StartLightSensorAsync(float sampleRate);
Task StopLightSensorAsync();
Task ResetAsync();
Task ProbeDeviceAsync();

WatchBleService

Guid WatchServiceUuid
Guid WatchCommandCharacteristicUuid
Guid WatchNotifyCharacteristicUuid
_ble : IBluetoothLE
_adapter : IAdapter
_device : IDevice?
_commandChar : ICharacteristic?
_notifyChar: ICharacteristic?
IsConnected : bool
MacAddress : string
WatchDisconnected : EventHandler<string>
WatchJsonReceived : EventHandler<string>

WatchBleService()
ConnectAsync(object, deviceObj) : Task<bool>
disconnectAsync() : Task
sendJsonToWatch(object, json) : Task<bool>
OnDisconnected(object?, DeviceEventArgs) : void
OnWatchNotification(object?, CharacteristicUpdatedEventArgs)

MetaWearService

Guid MetaWearUuid
byte MW_MODULE_ACCELEROMETER
byte MW_MODULE_GYRO
byte MW_MODULE_AMBIENT_LIGHT
byte MW_MODULE_MAGNETOMETER
float ACC_SCALE_16G
float GYRO_SCALE_2000DPS
float BMM150_SCALE

event DeviceDisconnected;
event AccelerometerDataReceived
event GyroscopeDataReceived
event MagnetometerDataReceived
event LightSensorDataReceived
bool IsConnected
string MacAddress
Task<bool> ConnectAsync(object device)
Task DisconnectAsync();
Task StartAccelerometerAsync(float sampleRate, float range)
Task StopAccelerometerAsync();
Task StartGyroscopeAsync(float sampleRate, float range);
Task StopGyroscopeAsync();
Task<DeviceInfo> GetDeviceInfoAsync();
Task StartMagnetometerAsync(float sampleRate);
Task StopMagnetometerAsync();
Task StartLightSensorAsync(float sampleRate);
Task StopLightSensorAsync();
Task ResetAsync();
Task ProbeDeviceAsync();

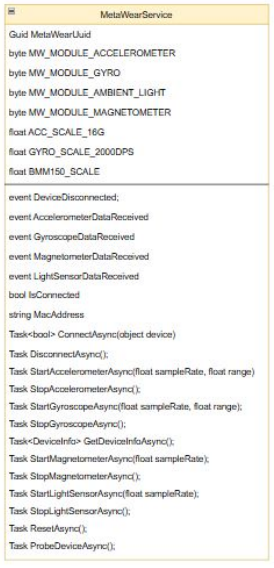
Implemented

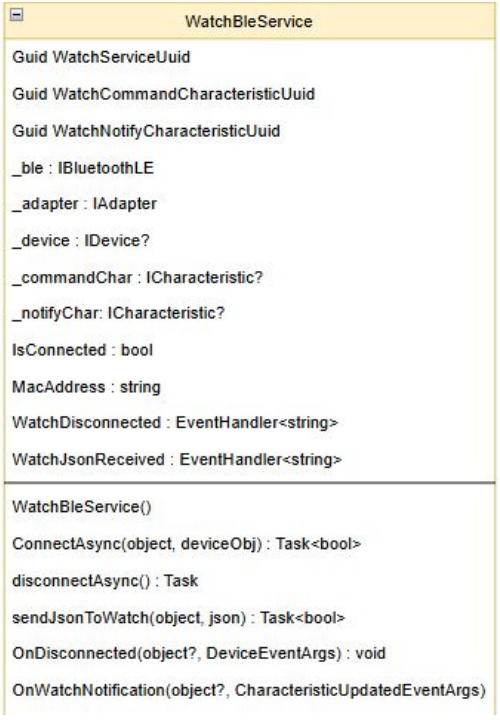
Partially Implemented

Not Yet Implemented

Mobile UML - WatchBleService

Servies



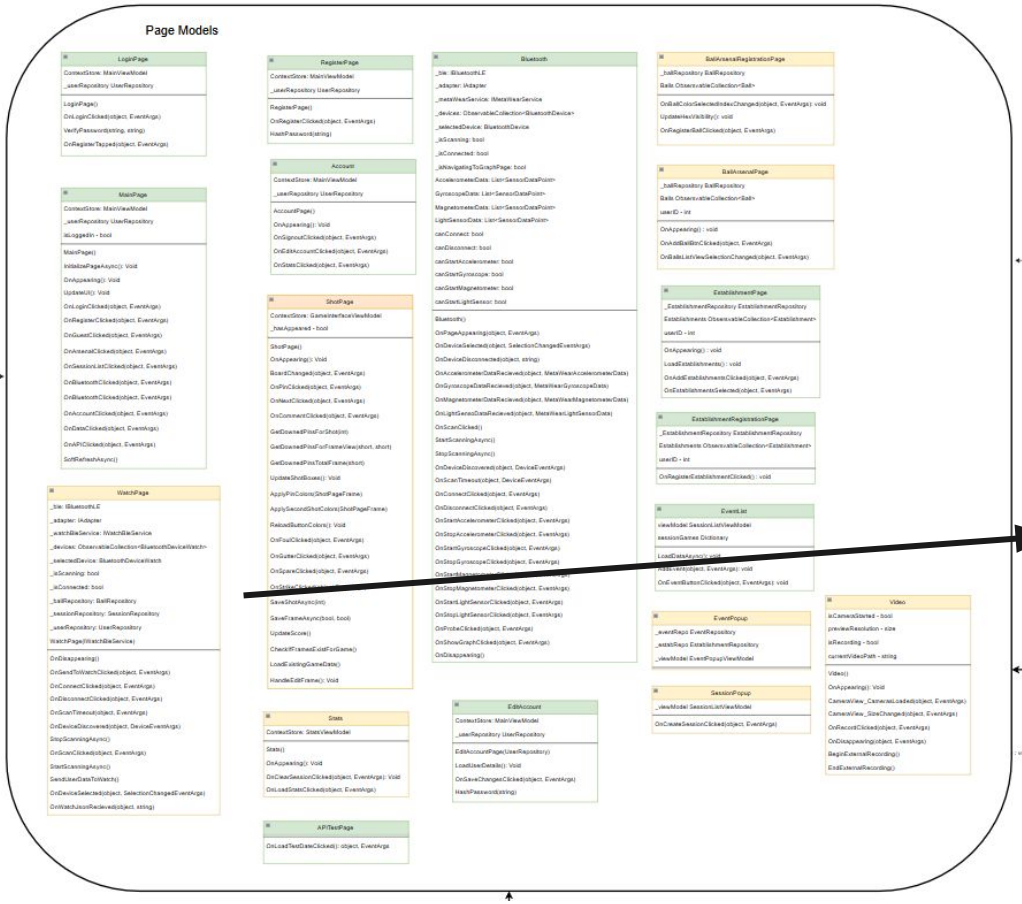


Implemented

Partially Implemented

Not Yet Implemented

Mobile UML - WatchPage



WatchPage

_ble: IBluetoothLE

_adapter: IAdapter

_watchBleService: IWatchBleService

_devices: ObservableCollection<BluetoothDeviceWatch>

_selectedDevice: BluetoothDeviceWatch

_isScanning: bool

_isConnected: bool

_ballRepository: BallRepository

_sessionRepository: SessionRepository

_userRepository: UserRepository

WatchPage(IWatchBleService)

OnDisappearing()

OnSendToWatchClicked(object, EventArgs)

OnConnectClicked(object, EventArgs)

OnDisconnectClicked(object, EventArgs)

OnScanTimeout(object, EventArgs)

OnDeviceDiscovered(object, DeviceEventArgs)

StopScanningAsync()

OnScanClicked(object, EventArgs)

StartScanningAsync()

SendUserDataToWatch()

OnDeviceSelected(object, SelectionChangedEventArgs)

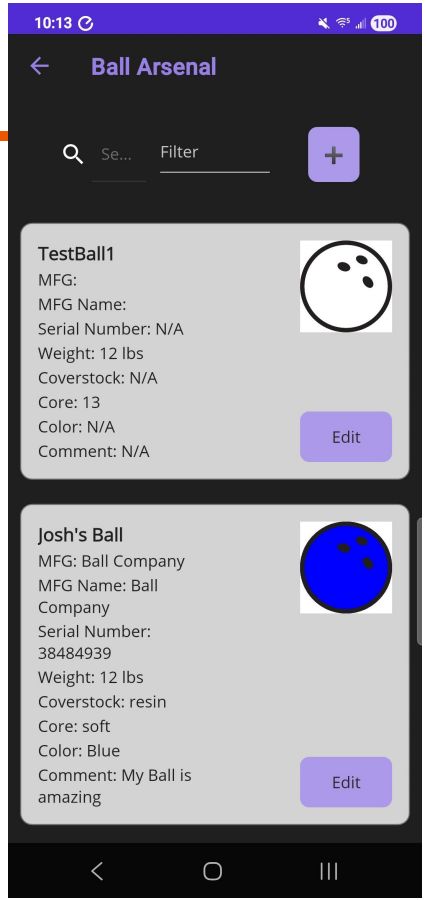
OnWatchJsonRecieved(object, string)

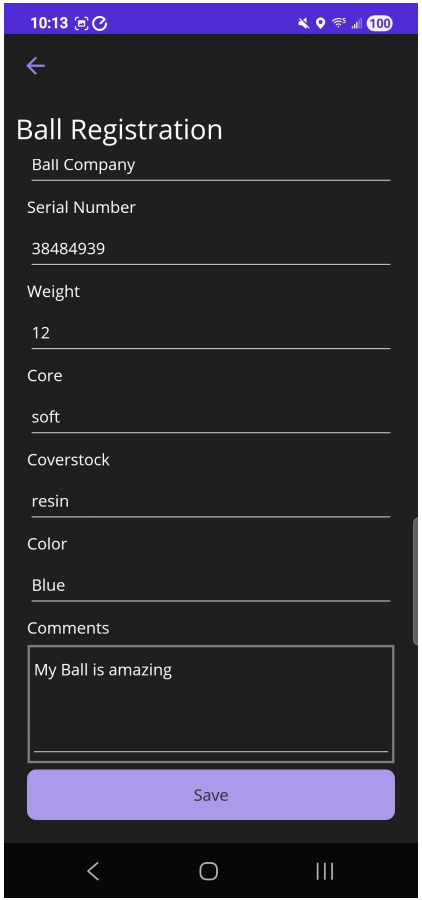
Implemented

Partially Implemented

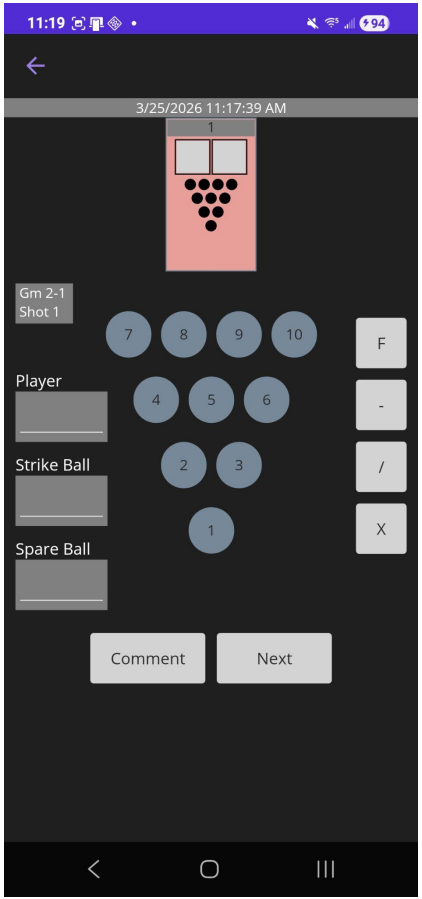
Not Yet Implemented

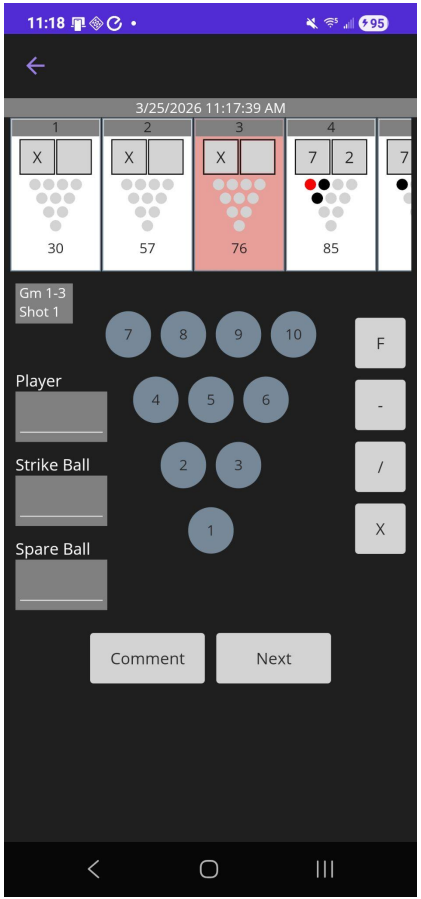
Mobile App - Ball Arsenal Update



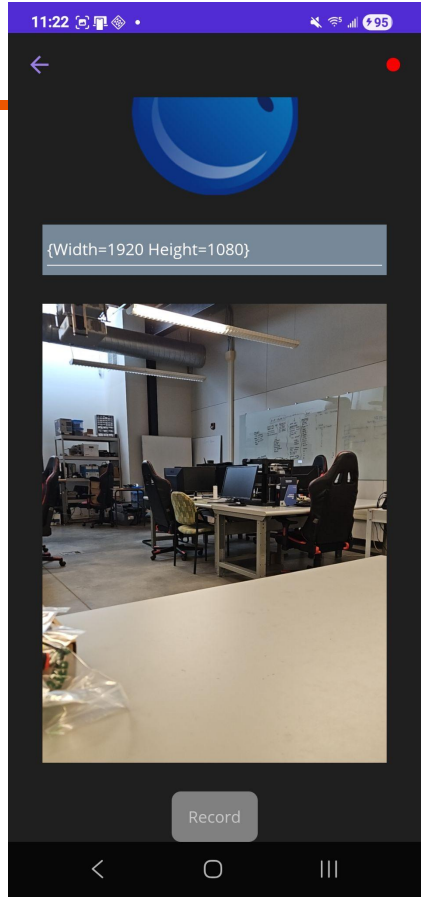


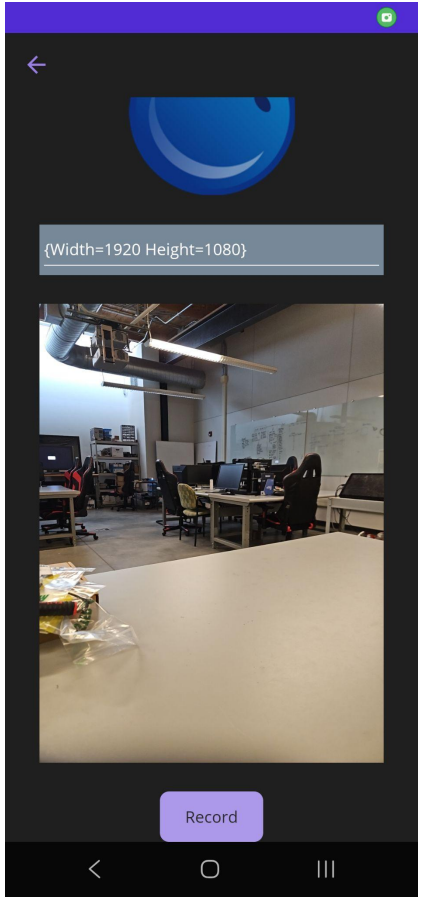
Mobile App - Shot Page



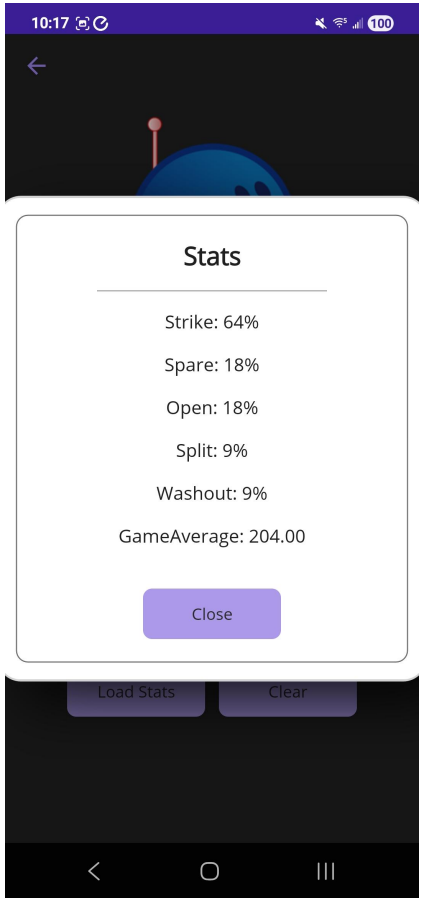


Mobile App - Video Pages

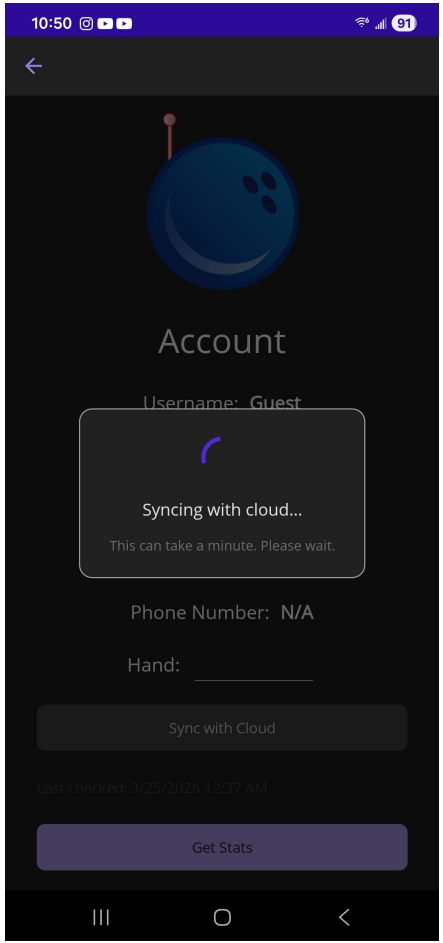


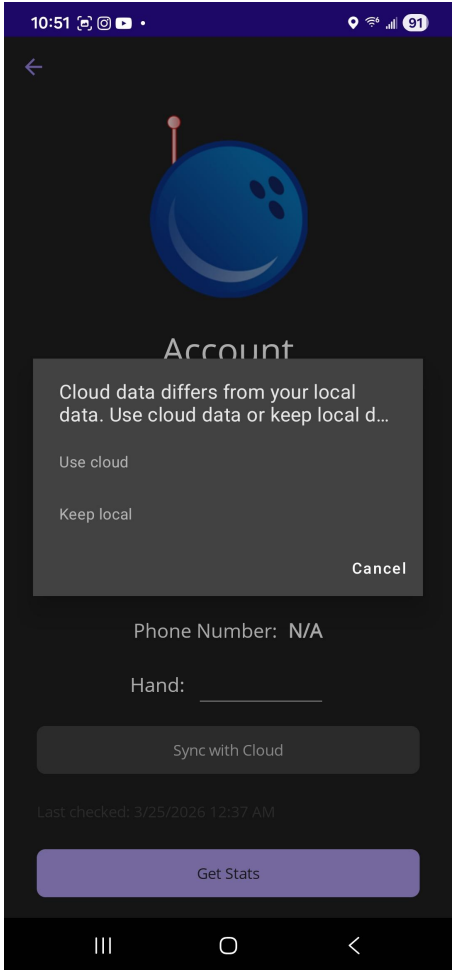


Mobile App - Stats Update

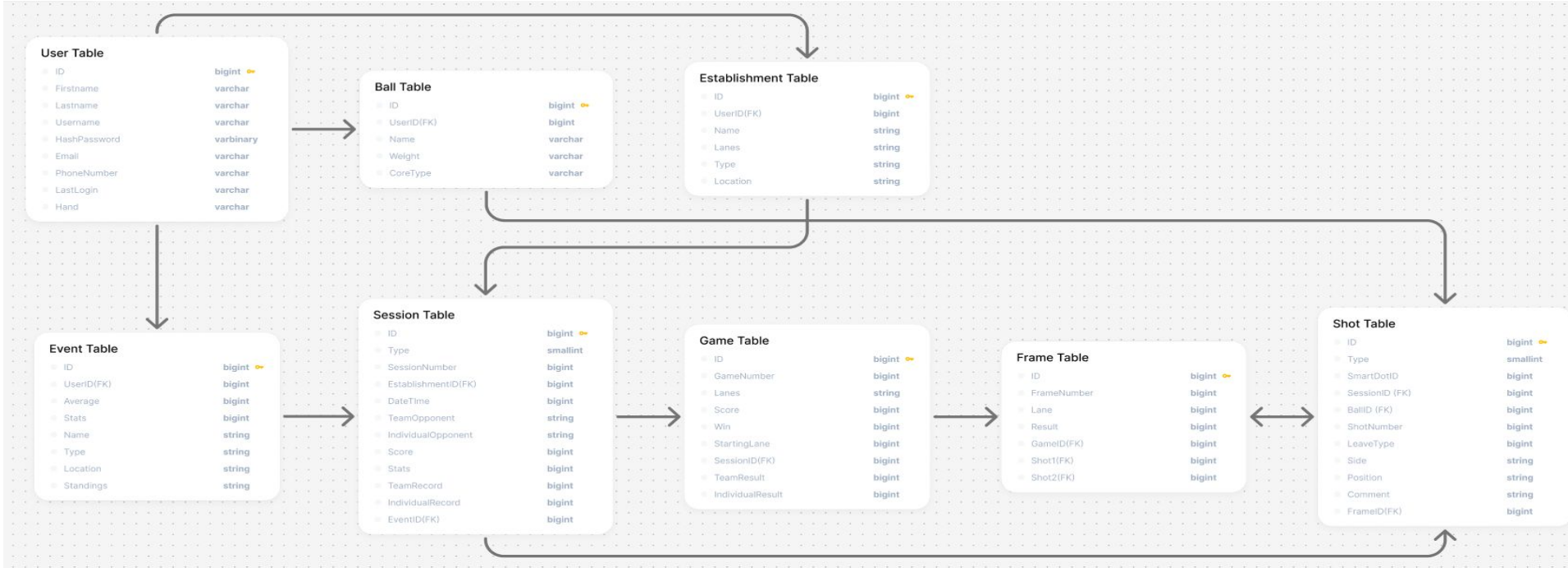


Mobile App - Account Page





Database Schema



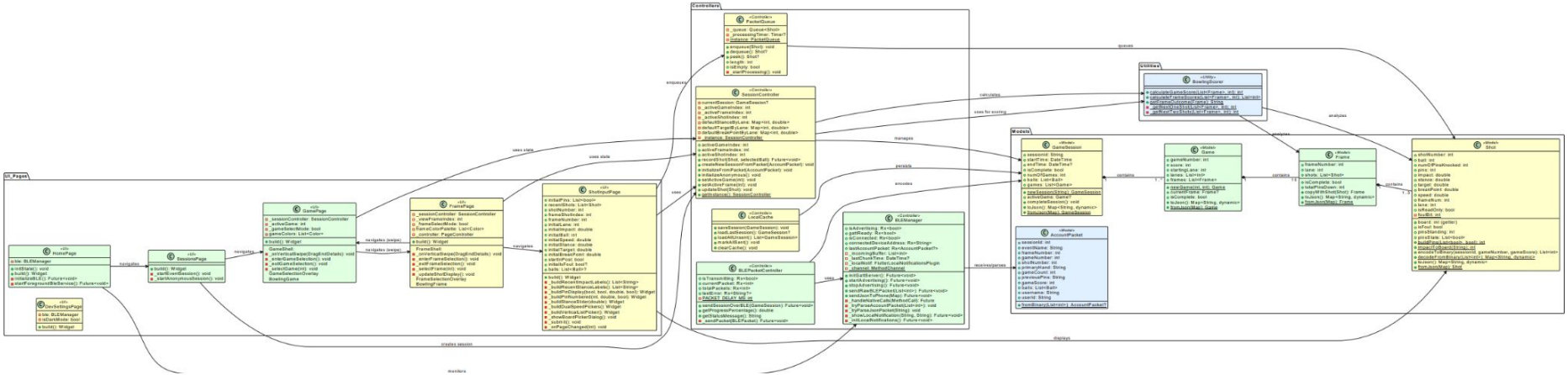
Cloud (MS1 Goals and Accomplishments)

- Get Mobile Sync Feature Working
  - Update Mobile User to have auth Token
  - Add Deletes
  - Add POCOs so Cloud is Uniform
  - Add MobileID in cloud Tables
- Implement changes for Pi team

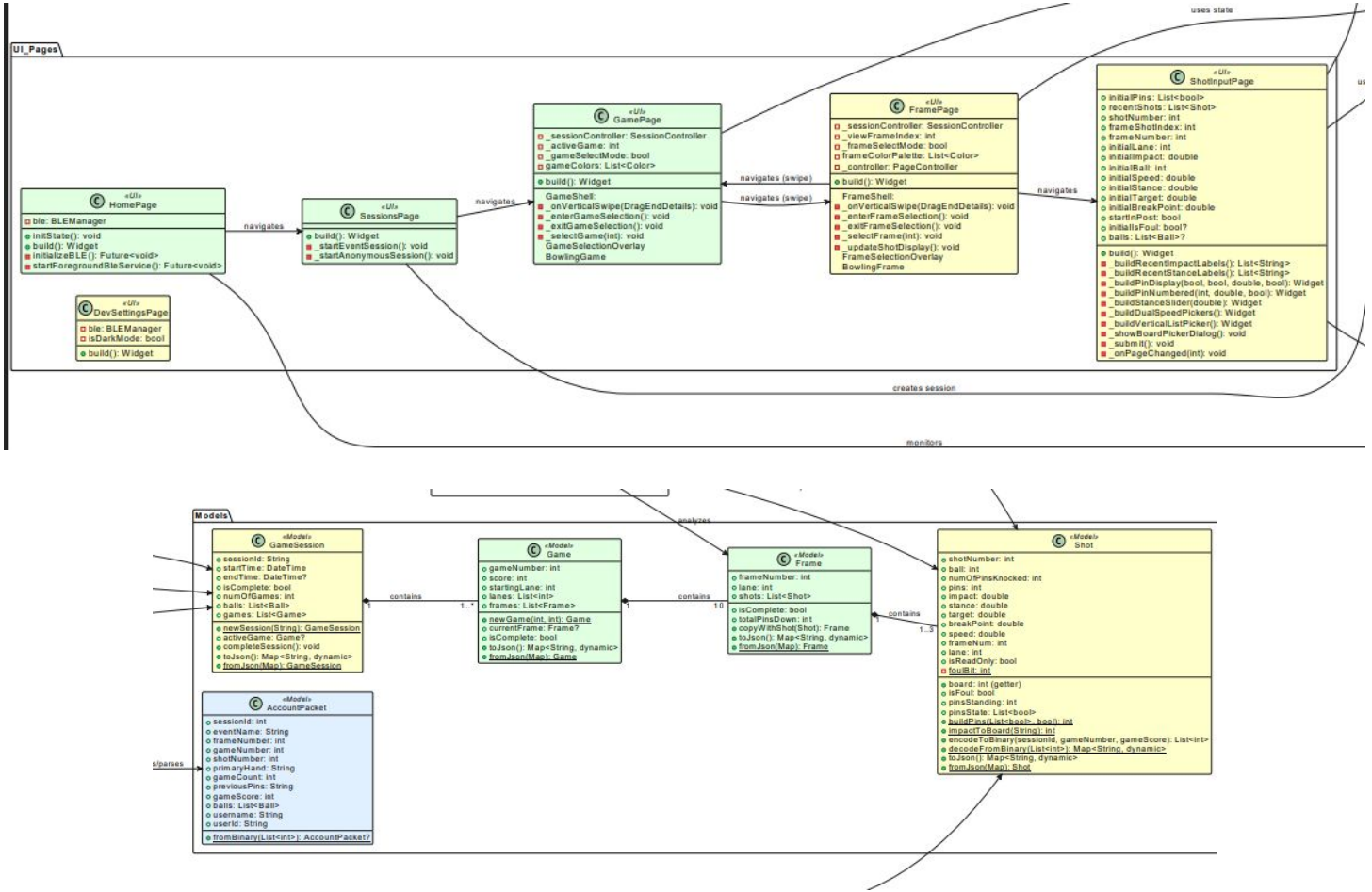
Cloud (Future: MS3)

- Update all the Tables to be Correct with the Mobile App
- Update Deletes to just mark some data as “Deleted”
- Fix any other issue that arises

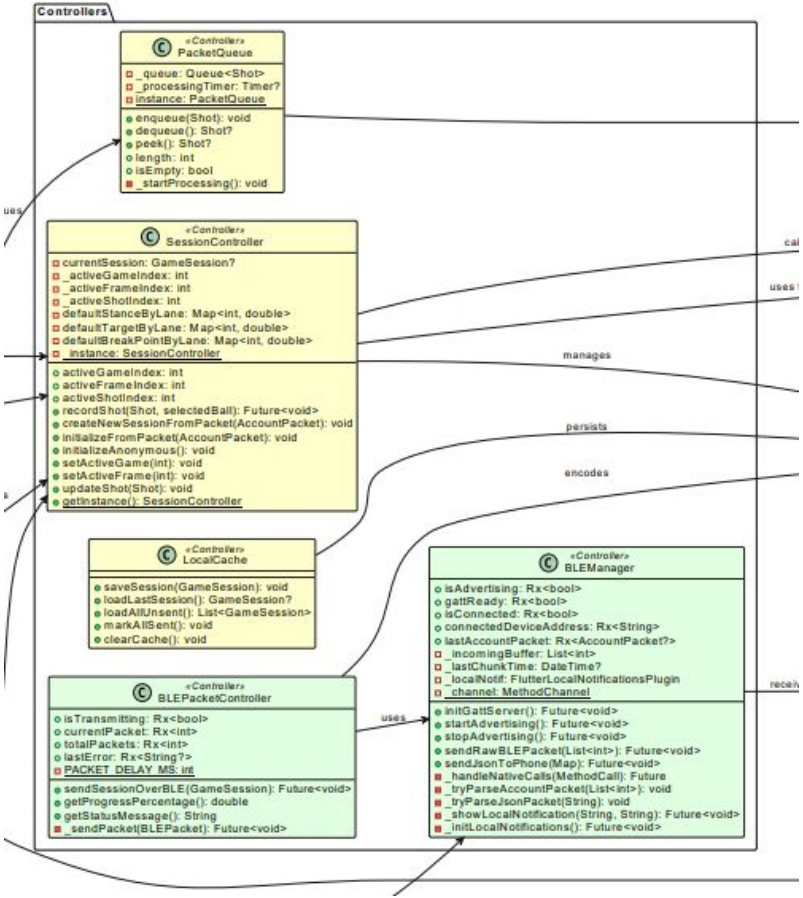
Smart Watch UML

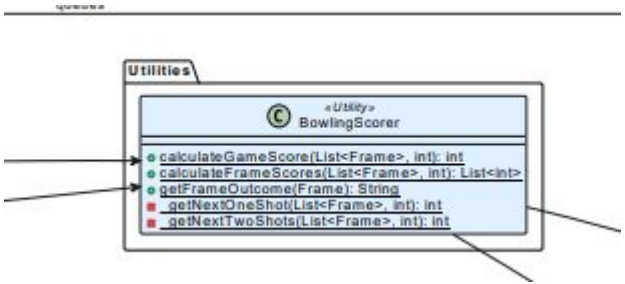


Smart Watch UML (Expanded)



Smart Watch UML (Expanded)





Tools and Technologies

- **Flutter** - framework
- **Dart** - programming language
- **Kotlin** - Android BLE foreground services
- **Visual Studio Code** - IDE
- **Bluetooth Low Energy** (BLE)
- **Android/IOS Launcher** - Emulator

Smart Watch (MS2 Goals)

- Fully functioning initial handshake
- Establish packet composition
- Build and unpack packets
  - Shot packet - sending per shot
  - Account packet - coming from phone after connection established
- Further refine new UI changes
  - New board inputs
  - Refactor variables
  - Clean up
- Game page functionality
  - Scoring

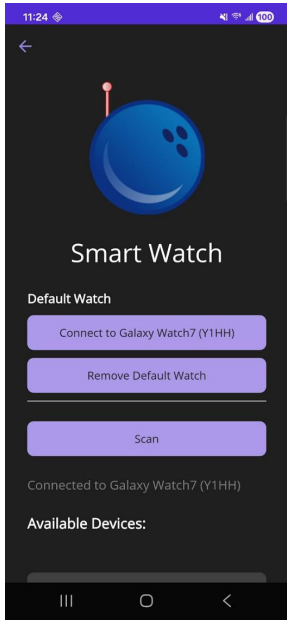
Smart Watch (MS2 Achievements pt. 1)

Phone Side:

- Mocked and built initial account packet
  - Sends user data from database
  - Send on connection
- .NET10 updates
  - Fixed video page command
- Watch page changes
  - Default watch for account
  - Name shown when scanning
- Phone parsing of incoming shot packet
  - Parses and puts in log

Watch Side:

- Receive and parse account packet
  - Merged with existing model and workflow
- Deprecated test code/hardcoded values
- Mocked and built shot packet
  - Shot objects
  - Sent on submission
- UI updates for session page
- Game page logic working
  - Connected to session being sent
  - Game scoring on watch



Packets

SHOT

- Packet Type, **1 byte**
- Version
  - V1, 1-127, upper bit (128) signifies a second byte (0/1, T/F) **1 byte**
  - V2, if needed for more space, **1 byte**
- Packet Length, size of packet, **1 byte**
- Shot
  - Session ID, 4 billion max num/ **4 bytes**
  - Game, 0-31, 5 bits
  - Frame, 0-11, 4 bits
  - Shot #, 1 or 2, 1 bit
  - Ball ID, 63 num max, 6 bits, (Game/Frame/Shot/Ball ID = **2 bytes**)
  - Pins, 10 bits, foul as bit 11, **2 bytes**
  - Stance, 100 max, **1 byte**
  - Target, 100 max, **1 byte**
  - Break Point, 100 max, **1 byte**
  - Impact/board, 100 max, **1 byte**
  - Ball Speed, max 250 (25mph), **2 byte**
    - Ball speed is x10, so 15.5 -> 155
  - Lane #, 160 ish, **1 byte**
  - Game score after calculating - **3 bytes**
  - **22 byte payload + 1 byte padding**

ACCOUNT

- Packet type, **1 byte**
- Version
  - V1, 1-127, upper bit (128) signifies a second byte (0/1, T/F) **1 byte**
  - V2, if needed for more space, **1 byte**
- Packet Length, size of packet, **1 byte**
- Session ID - **4 bytes**
- Event Name - **variable bytes**
- Primary Hand - **1 byte** (0 = left, 1 = right)
- Frame Number - **4 bytes**
- Game Number - **2 bytes**
- Shot Number - **2 bytes**
- Game Count - **2 bytes** (number of games in session)
- Previous Shot Pins - **2 bytes**
- Current Game Score - **4 bytes**
- Ball Count - **1 byte**
- For Each Ball:
  - Ball ID - **4 bytes**
  - Ball Name - **variable bytes**
- Username - **variable bytes**
- User ID - **4 bytes**

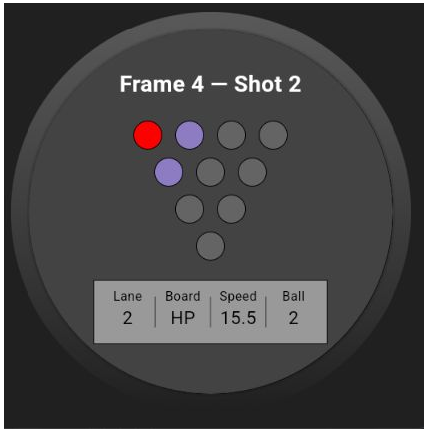
```
I/flutter (20826): WATCH SESSION: Sending shot packet (23 bytes):
I/flutter (20826): WATCH SESSION Packet hex: 0x03 0x01 0x00 0x17 0x00 0x00 0x00 0x04 0x09 0x47 0x00 0x40 0
x28 0x2a 0x2c 0x22 0x00 0xa5 0x01 0x11 0x00 0x00 0x00
```

```
03-25 11:21:58.511 30777 30823 D app_process64: PHONE BLE RECEIVED → 03 01 00 17 00 00 00 04 09 47 00 40 28 2A 2C 22 00 A5 01 11 00 00 00
03-25 11:21:58.511 30777 30823 D app_process64: ParseShotPacket: PacketType=0x03, Version=0x01 0x00
03-25 11:21:58.512 30777 30823 D app_process64: ParseShotData: Shot packet parsed successfully:
03-25 11:21:58.512 30777 30823 D app_process64:   Session ID: 4
03-25 11:21:58.512 30777 30823 D app_process64:   Game: 1, Frame: 2, Shot: 2, Ball ID: 7
03-25 11:21:58.512 30777 30823 D app_process64:   Pins Standing: 7 (0x040), Foul: 0
03-25 11:21:58.513 30777 30823 D app_process64:   Stance: 20.0, Target: 21.0, Break Point: 22.0, Impact: 17.0
03-25 11:21:58.513 30777 30823 D app_process64:   Ball Speed: 16.5 mph
03-25 11:21:58.513 30777 30823 D app_process64:   Lane: 1
03-25 11:21:58.514 30777 30823 D app_process64:   Game Score: 17
```

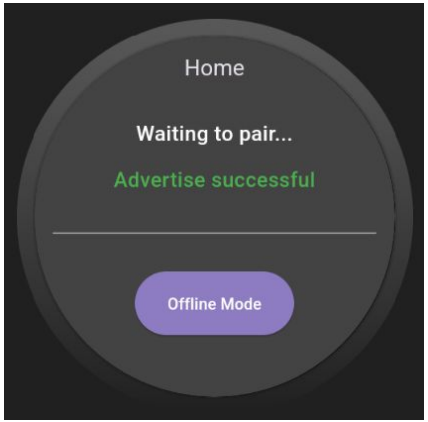
```
I/flutter (20826): WATCH BLE Account packet payload length: 98, total with header: 101, current buffer: 10
1
I/flutter (20826): WATCH BLE *** COMPLETE ACCOUNT PACKET (0x01) RECEIVED ***
I/flutter (20826): WATCH BLE Packet hex: 0x01 0x01 0x62 0x04 0x00 0x00 0x00 0x68 0x75 0x63 0x6b 0x73 0x42
0x64 0x61 0x79 0x00 0x01 0x02 0x00 0x00 0x00 0x01 0x00 0x02 0x00 0x01 0x00 0xc8 0x00 0x08 0x00 0x00 0x00 0
x04 0x05 0x00 0x00 0x00 0x62 0x69 0x67 0x62 0x61 0x6c 0x6c 0x73 0x00 0x06 0x00 0x00 0x00 0x68 0x75 0x63 0x
6b 0x79 0x63 0x68 0x65 0x65 0x73 0x65 0x00 0x07 0x00 0x00 0x00 0x73 0x74 0x72 0x69 0x6b 0x65 0x44 0x61 0x7
4 0x00 0x08 0x00 0x00 0x00 0x73 0x70 0x61 0x72 0x65 0x44 0x61 0x74 0x00 0x63 0x68 0x75 0x63 0x6b 0x00 0x03
 0x00 0x00 0x00
```

New Pages

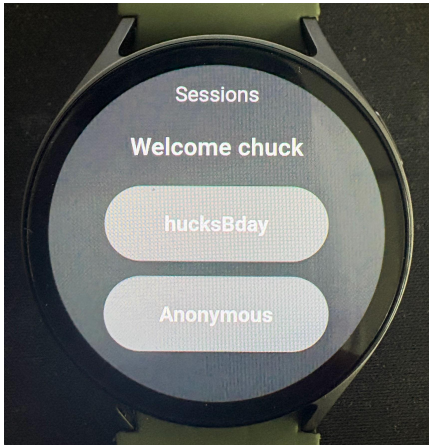
New coloring on second shot



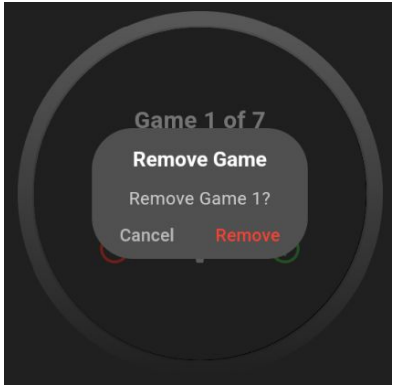
New coloring + Info Bar



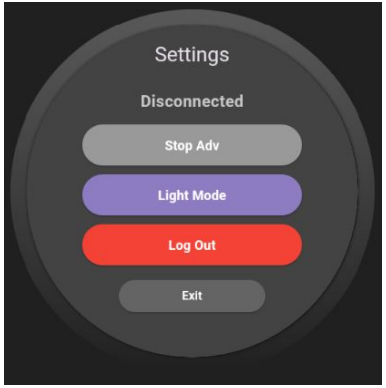
Home page



**Sessions page



Remove confirmation

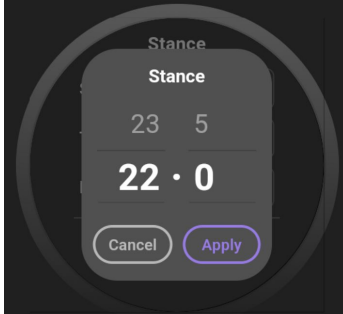


Settings page

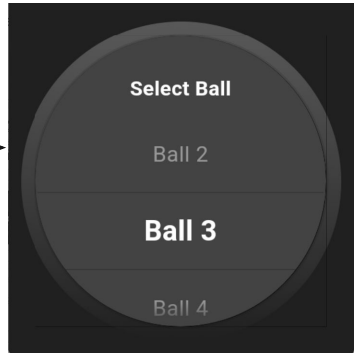
Smart Watch (MS2 Achievements pt. 2)

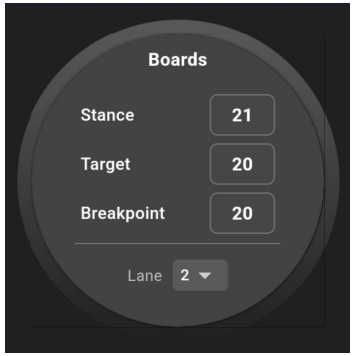
- Refactored shot model
  - Reflects new input
  - Now records Stance/Target/Breakpoint/Impact as board numbers
- Impact name/board mapping
  - Displays the full or abbreviated name, but is recorded as a number
- Updated shot input UI
  - Fixed some colorings/spacing/text sizing
  - Polished popups
  - Made all buttons match theme better
  - Stance and speed persists from previous shot
- Added Recent Results Functionality
  - Pulls last 3 shots, if any, from that given shot number (1/2)

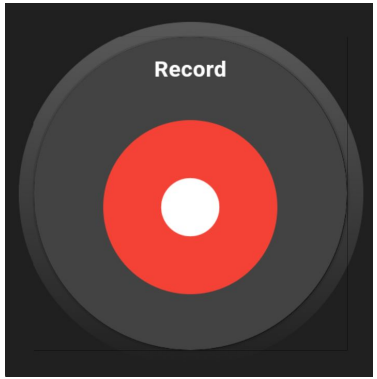
```dart
// models/shot.dart
class Shot {
  final int shotNumber;
  final int ball;
  final int numOfPinsKnocked;
  final int pins;
  final double impact;
  final double stance;
  final double target;
  final double breakPoint;
  final double speed;
  final int frameNum;
  final int lane;
  final bool isReadOnly;
```

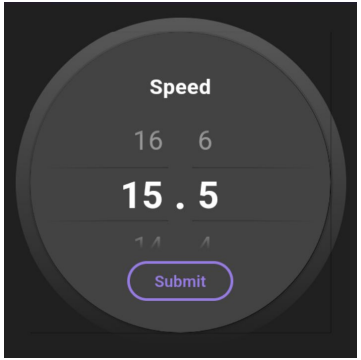


Updated UI









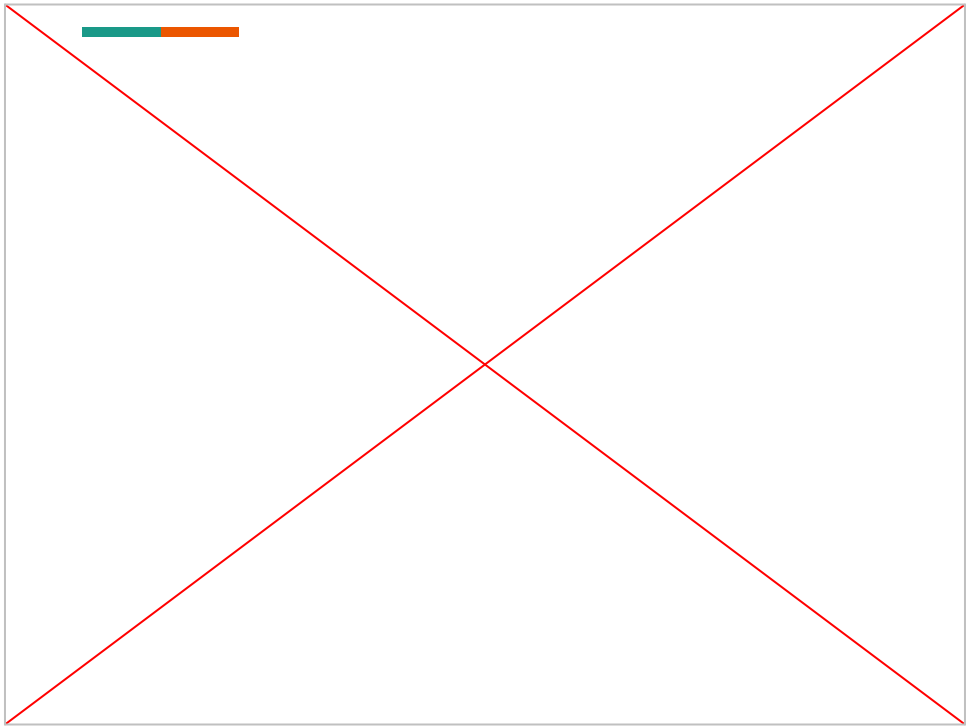
Submission of Shot

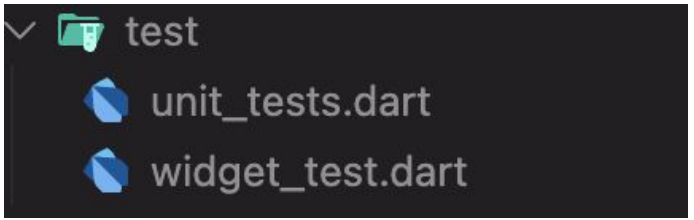
Smart Watch (Future MS2)

- Integrate Bluetooth Packets into the Queue
  - All comms from the watch goes through the queue
  - Logic to keep retrying if a packet fails to send
  - Basic logic for writing packets to local storage
- Fix Recent Results to work fully
  - Edge cases
  - Use with updated workflow
  - Bug fixes
- Info bar
  - Adjust info

- Unpack shot data on phone
  - Put data into database
- Add to database
  - Date and time for events
  - Lane
- Settings page
  - Add functionality for Light/Dark
  - Log out
- Stop from watch to phone
  - End of game
  - Manual
- Update UI on phone side
  - Show shots sent
- WIKI page
- SO MANY TESTS

Smartwatch Demo + Unit Tests






```
charlescarroll@DESKTOP-C7TGMLH flutter_prototype % flutter test test/unit_tests.dart
00:03 +31: All tests passed!
charlescarroll@DESKTOP-C7TGMLH flutter_prototype % flutter test ./test/widget_test.dart
00:01 +3: All tests passed!
```