3. APPROACH

This document details the approach to the technical and practical constraints required for successful implementation. Three distinct subsystems comprise the design: a glove garment, a wrist-mounted microcontroller, and a coaching software application. The glove subsystem will include temperature, stretch, pressure sensors, and inertial measurement units (IMU). This coagulation of inputs will be physically connected to a battery-powered microcontroller that will sample the sensor inputs with high resolution. After data is sampled, it will be broadcast to an application, stored in a persistent database, analyzed, and finally displayed to provide golf swing feedback to the user.

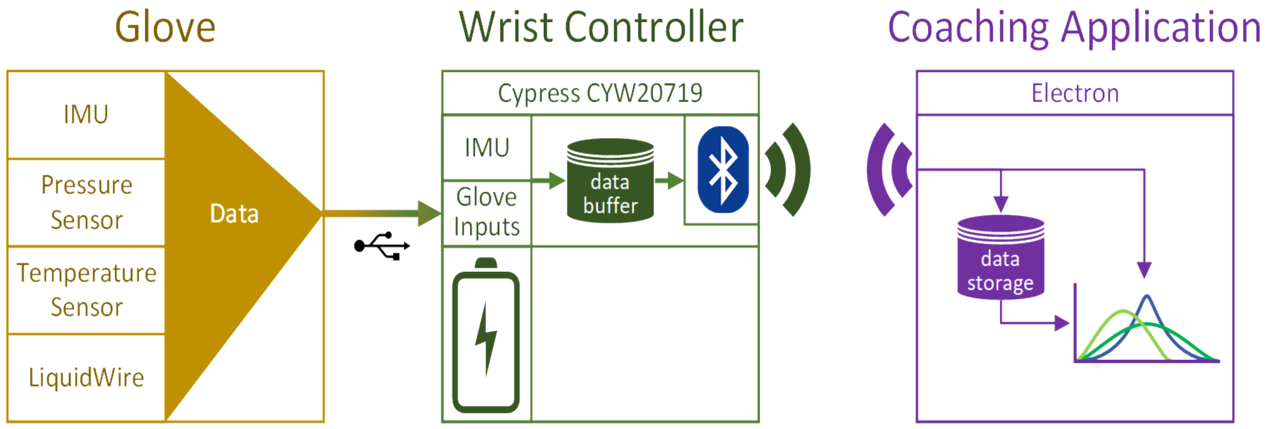

Figure 3.1. System Overview

3.1. Hardware

The Golf Glove is composed of the subsystems outlined in the Design Constraints section of this document. In this section, critical hardware components are compared and optimal candidates are selected.

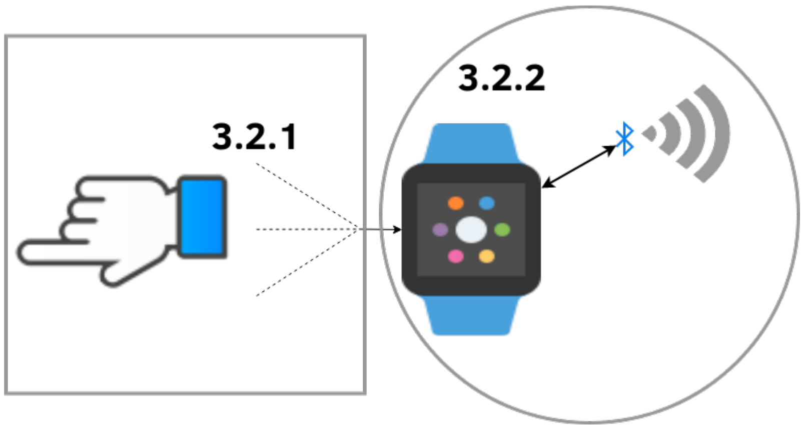

Figure 3.2. Hardware Subsystems

3.1.1. Glove

The glove subsystem consists of the garment itself, linear stretch sensors, flexible pressure sensors, an IMU, and electrical connector components. The garment will provide mounting for the sensors. The sensors will receive power and transfer data to the controller through the electrical connector mounted on the glove.

3.1.1.1. Glove Garment

The glove garment is the mounting point for the linear stretch sensors, an IMU, and pressure sensors. A connector will also be mounted on the glove to connect to the controller, making the glove and controller exchangeable. For the initial prototype, two sizes of gloves will be created to fit different golfer's hand sizes comfortably. The glove will be constructed out of leather, the standard material for golf gloves.

3.1.1.2. Linear Stretch Sensors

The linear stretch sensors are crucial to the Golf Glove design to measure wrist angles. These sensors vary linearly in either resistance or capacitance as the sensor stretches and typically consists of an elastic material fused with a flexible resistive material or capacitive shape.

Two brands of linear stretch sensors were considered: LiquidWire and StretchSense. LiquidWire was ultimately chosen because of its resistive variability and low cost. Additionally, detecting a change in resistance with a microcontroller requires a simpler circuit than detecting a change in capacitance.

Because LiquidWire sensors are available in custom lengths, the design will use a length of 11.5 cm for development. Upon examining the natural bend of the hand, the maximum required stretch length of a sensor is approximately 2.5 cm. LiquidWire rates their sensors as stretchable up to 125% of their resting length without damaging the sensor; thus, a resting length of 11.5 cm was selected [1]. The resting resistance of each sensor will be approximately 50 Ω; however, due to the manufacturing process of LiquidWire, the resistive value will vary slightly. As a result, a calibration procedure will be required for each sensor. A test of the sensors shows a typical resistive increase of 15 Ω at full stretch.

3.1.1.3. Flexible Pressure Sensors

The Golf Glove must record the force of the user's hand on the golf club accurately in an unobtrusive manner. Therefore, the design will use two flexible pressure sensors to detect the force in two locations on the palm of the hand. Having two measurement locations allows the device to analyze the user's distribution of force on the golf club. Per the design constraints, these measurements will be used to discourage improper swing form that increases injuries such as tennis elbow.

A study found that maximum human grip strength reaches a value of around 55 kgf. A reasonable golf club grip will certainly not encroach into this maximum range as it peaks at a high-end value of 10 kgf [2, 3]. Thus, a sensor with a maximum pressure sensing ability of 25 kgf is appropriate. Table 3.1 shows possible pressure sensors that satisfy this 25 kgf requirement.

Table 3.1. Flexible Pressure Sensor Comparison

Name	**Diameter (mm)**	**Price (USD)**	**Resistance Range (Ω)**	**Interface/Communication**
Interlink 400 Series [4]	<15	7.00	Infinite-250	Analog
Velostat Conductive Sheet [5]	Custom Sheet	3.95	Variable	Analog

The Interlink 400 series of resistive pressure sensors was chosen due to the simplicity of their integration into the garment. These flexible sensors can be reasonably bent without damage and are manufactured with tinned probes that allow for a direct connection to the glove connector. While the Velostat conductive sheet would need to be manually crafted to specification, the Interlink 400 series sensors are manufactured to a specification; meaning, there is no need to cut or measure for each glove and the resistance value can be coded into the device firmware as a constant value. Additionally, while the price of materials per sensor is higher for Interlink sensors, using them drastically improves the manufacturability of the Golf Glove because the sensors to not need to be manually crafted.

3.1.1.4. Electrical Connector

The electrical connection between the glove's sensors and the wrist-mounted controller should be robust and capable of handling multiple data inputs and outputs. Counting the amount of sensor channels and voltage inputs/outputs, it was determined that a minimum of eight conductors would be needed. These conductors include the outputs of the stretch sensors, pressure sensors, and IMU. Rather than create a new proprietary connector, thereby compromising the manufacturability design constraint, the design will use a standardized connector. In Table 3.2 shows a comparison of potential connector standards.

Table 3.2. Standard Connection Comparison

Standard	Number of Connections	Reversible	Max Power (A)
USB-C	12	Yes	5
USB-Micro	5	No	5
Cat-5	8	No	0.63

The design will utilize a standard USB-C connection to connect the glove sensors and the wrist-mounted controller. USB-C provides more than enough power throughput to the sensors and an adequate number of pins to transfer data to the wrist-mounted microcontroller. The reversibility is an added bonus; removing the possible confusion and frustration of irreversible connector standards certainly enhances the user's overall experience.

3.1.2. Wrist-Mounted Controller

The wrist-mounted controller contains an IMU, the microcontroller, the radio, and the battery. Other than the IMU, these components are responsible for collecting and transmitting data to the coaching application. This section compares the design options considered for these components.

3.1.2.1. Inertial Measurement Unit

The Golf Glove requires two IMUs—one in the wrist-mounted controller and one on the back of the hand—to accurately measure movement according to design constraint of wrist acceleration, hand orientation, and hand acceleration within a 5% margin of error. These IMUs will measure acceleration of the hand and the rotation of the wrist. A typical golf swing involves linear acceleration of up to 1.5 g and rotational speeds of up to 1000 degrees per second at the wrist [6, 7]. Table 3.3 provides a comparison of the considered IMUs.

Table 3.3. Inertial Measurement Unit Comparison

IMU	Cost (USD)	Power Draw (mA)	Accelerometer (g-force)	Gyroscope (degrees per sec)	Output Resolution/Rate
STM LSM9DS1 [8]	\$ 6.14	1.9–3.6	±2/±4/±8/±16	±245/±500/±2000	16-bit @ 1 kHz
Bosch BMX055 [9]	\$ 7.02	2.4–3.6	±2/±4/±8/±16	±125 to ±2000	16-bit @ 1 kHz
mCube MC6470 [10]	\$ 3.01	0.07–1.0	±2/±4/±8/±16	500 (Interpolated)	14-bit @ 0.1 kHz

The design constraints specify a minimum sampling rate for sensor data of 120 Hz; this immediately eliminates the mCube IMU device since it has maximum sampling rate is 100 Hz. The Bosch and

STMicro devices have comparable inertial measurement capabilities with selectable 2 g, 4 g, 8 g, or 16 g acceleration scales and similarly selectable ranges of rotational speed scales. The selectable modes define the minimum and maximum range of values expected; a lower acceleration or rotational speed setting allows the device to measure with more precision. The least precise mode supports measurement accuracy to 0.0006 g and 0.004 degrees per second, well within the design constraint of 1.2 m/s. Because both the devices offer nearly identical measurement and output modes, the STM LSM9DS1 was selected due to its lower price and smaller power draw during sleep-mode. Additionally, the LSM9DS1 device is widely used in many hobbyist and professional projects and comes with a well-tested set of open source libraries for interfacing.

3.1.2.2. Microcontroller

The design requires a low-power microcontroller because of its constraints for wireless communication, portability, and sustainability. Table 3.4 compares the microcontrollers considered for the Golf Glove design.

Table 3.4. Microcontroller Comparison

Microcontroller	**Cost (USD)**	**Wireless Connectivity**	**Power Draw (mA)**	**Flash Memory (kB)**	**ADC**
Cypress CYW20719 [11]	$ 9.63	BR/EDR/BLE Bluetooth 5.0 @ 2 Mbps	10 mA	1024	1 x28-channel, 10-bit, 3.6 ksps
Microchip IS1871 [12]	$ 2.98	BLE Bluetooth 5.0	8 mA	256	1 x6-channel

The Cypress CYW20719 was selected because it contains a BLE-capable radio and an adequate number of built-in ADCs. The Microchip controller is not available in North America during the time of this project and does not have a readily available development environment. Cypress provides WICED: a featured, free, and integrated development environment for use with the microcontroller. The CYW20719 comes with a complete Bluetooth stack on-board and Cypress has released several development and Bluetooth debugging tools specifically for the CYW line of products. Based off these factors, the Cypress chip and development environment best fit the design.

3.1.2.3. Radio

The Golf Glove will use the CYW20719's built-in Bluetooth 5.0 radio for wireless communication. By using the built-in radio system, the controller will not need to interface with an external communication system. This decision is also driven by the concern for the overall size of the system. By utilizing a built-in radio, the device size and cost are minimized.

The design specifications require an internal radio system with a minimal power draw with moderate to high data throughput. The integrated BLE 5.0 stack offers a host of benefits over other systems such as infrared transmission or WiFi. While infrared systems consume a small amount of power, the connection itself is intermittent. WiFi faces the opposite problem; while transmission distance and reliability are excellent, power draw is comparatively large. The chosen protocols, BLE paired with the Generic Attribute (GATT) application layer, satisfies both the power and data throughput constraints.

3.1.2.4. Power Requirements

The Golf Glove requires a battery powered system that must remain operational for a minimum of five hours according to the design constraints. Calculations below include max power usages of each high power internal/external device to describe an estimated minimum capacity required. The power consuming components the design considered are the I/O pins, the internal Bluetooth module, and the dual IMUs. The battery capacity depends on these devices' max power draw. The maximum power usage calculations over a five hour period are listed in Table 3.5. By summing the power usage of each device and adding another ~150 mAh as a buffer, it was determined that a 500 mAh battery would be sufficient for the design. Additional considerations for the battery selection are rechargeability and form factor.

Table 3.5. Power Usage Calculations

Device/Subsystem	**Max Power (mAh)**	**Quantity Needed**	**Hours of Use**	**Power Usage (mAh)**
I/O Pins	16	4	5	320
Radio RX/TX	11.5	1	5	57.5
IMU	4.6	2	5	46

Total Power Usage: (I/O pins) + (Radio) + (IMUs**)** = 320 mAh + 57.5 mAh + 46 mAh = **423.5 mAh**

Note, total power usage does not consider the sleep time. The calculation is a worst-case scenario to determine the minimum battery specifications. Given the above calculation, the battery should contain a minimum capacity of 500 mAh. Table 3.6 compares the battery options for the design.

Table 3.6. Battery Choices

Component (Battery)	**Price (USD)**	**Capacity (mAh)**	**Output Voltage (V)**	**Form Factor**	**Chemical Composition**	**Rechargeable**
LP503035 [13]	1.00	500	3.7	Pouch Cell	Li-Ion Polymer	Yes
UxCell AAA [14]	1.64	500	1.2	AAA Cylinder	Ni-MH	Yes
Duracell AAA [15]	0.55	541	1.5	AAA Cylinder	Alkaline	No
Panasonic CR3032 [16]	2.45	500	3.0	Coin Cell	Lithium	No

The LP503035 was selected as the power source for the Golf Glove. The battery incorporates recharging capabilities and contains a capacity sufficient to run the Golf Glove for 5 hours. This battery is sufficient to drive the internal circuitry with an average output voltage of 3.7 V at a cost-effective price. The LP503035 contains an on-board over voltage protection circuit that will prevent the battery from discharging past 3.0 V, all contained inside of a thin form factor allowing for sleek containment inside the main wrist housing.

3.2. Software

The design will require two pieces of software: one for the wrist-mounted controller and one for the coaching application. The wrist-mounted controller will collect data from the glove's various sensors and will then package and transmit that data via Bluetooth to the coaching application, which will process, store, and display the received data. Because the coaching application will fulfill both backend and frontend roles, the backend and frontend approaches will be discussed in separate sections.

3.2.1. Microcontroller Firmware

Microcontroller firmware drives the sensor logic for the design, handling swing detection as well as sensor data acquisition. The main components of the firmware are described below. The firmware will poll over I2C for the IMUs and on-board ADCs for the linear stretch sensors and pressure sensors. As for measurements, the two IMUs will each produce three gyrometer, accelerometer, and magnetometer measurements. The two pairs of stretch sensors will measure two wrist angles and the microcontroller will measure a single temperature reading. One sensor will go slack in each direction while the opposite sensor is reading correct stretch resistance. Sensor readings will have a resolution of 16 bits for each value.

The firmware will have two main modes of recording. The first mode detecting a swing using a set threshold for sensor values. The data will be recorded at a rate of 120 Hz to produce a high-resolution reading of swing movement. A swing is completed once the IMU stops reading acceleration changes and the hand pressure releases. While recording, data will be stored in the microcontroller flash memory. Once completed, a pointer to the swing memory will be added to a FIFO queue and then pushed to the coaching application on the next event loop.

The second mode is real data transmission mode where sensor data is sent to the application as it is measured in real time. This mode will be useful for the user to see sensor data while practicing swing positioning, without the detection interfering with readings. The coaching software will receive a stream of sensor data as it is measured and will be rate limited by having the microcontroller sleep for the duration required to achieve a target rate of 120 Hz.

3.2.1.1. Implementation

Microcontroller firmware will be developed in C. The software will consist of a main event loop that will sleep while not polling the sensors. At a rate determined by testing, the microcontroller will interrupt sleep and read the sensor values to check if swing conditions are met. The rate will be determined by testing for power usage and swing detection accuracy. During mode one, swings will be identified by a sudden change in pressure as well as accompanying change in accelerometer velocity reading. Once a swing is detected, the microcontroller begins reading the sensor data into the device memory. While recording, the microcontroller is also checking for the swing end condition. Once the stop condition is met, the swing data will be sent to the coaching software.

In mode two, the swing detection logic will be disabled and the data will be communicated to the coaching software immediately while the controller is in the "Detect a Swing" state. The controller software will poll for received Bluetooth data and check for a flag from the controller software. The flag will either disable or enable real-time reading, which will change the state logic for the “Detect a Swing” state.

3.2.1.2. Software Flow Diagram

The state flow diagram in Figure 3.3 models the three states of the microcontroller.

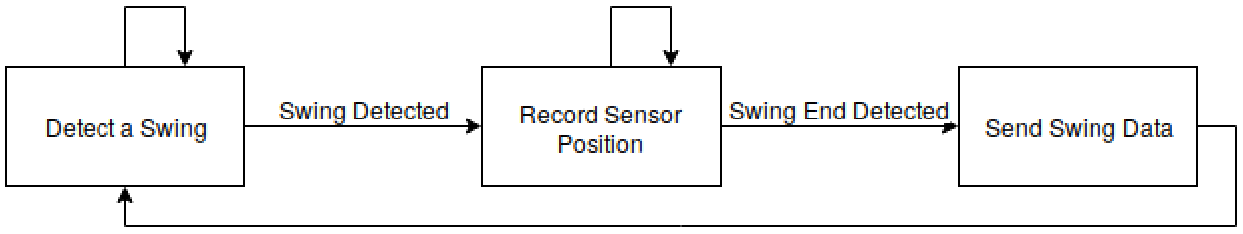

Figure 3.3. Firmware Stateflow Diagram

3.2.1.3. Interfacing

The controller will interface with the coaching software over Bluetooth. Once the controller receives power, it will begin broadcasting for a Bluetooth connection. Next the coaching software will initiate a Bluetooth connection to the controller and pair the two so that data transfer can begin. The connection status will be displayed on the coaching software to the end-user.

3.2.2. Coaching Software Backend

To facilitate the flow of swing data for analysis, the coaching application will receive the data from the wrist-mounted controller and subsequently process, store, and display it. The application’s logic will be split between a backend, which is responsible for handling the Bluetooth connection as well as data processing and storage (Bluetooth receiver), and a frontend graphical user interface (GUI) that renders the visual components. This section exclusively details the software approach for the backend of the coaching application and Section 3.2.3 explains the frontend approach.

As outlined in the Firmware section, the device will have two operating modes: real-time and full-swing. The coaching application will have a user option to choose the mode of operation and will have different visuals accordingly. Figure 3.4 provides an overview of this subsystem with the chosen technologies.

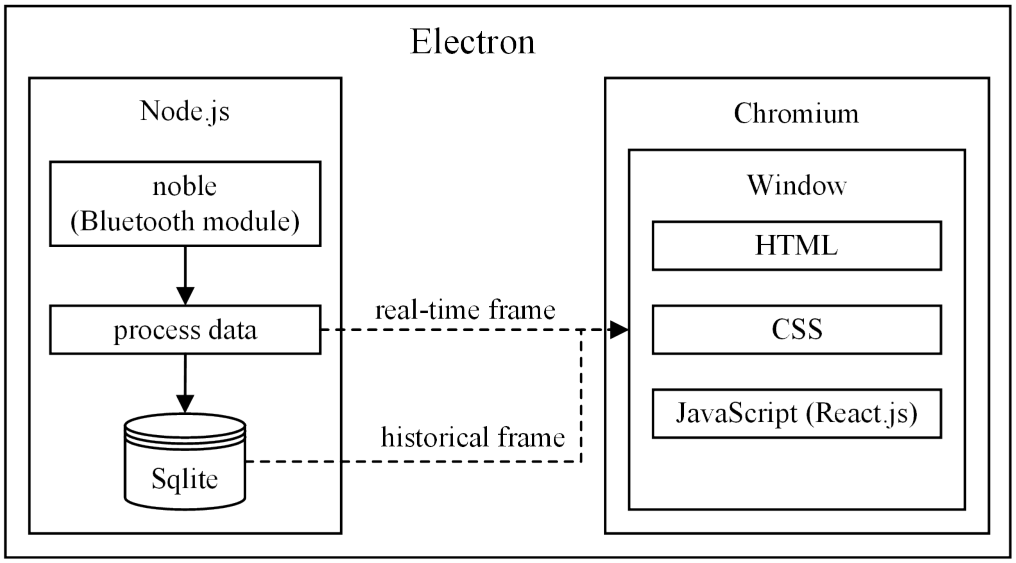

Figure 3.4. Bluetooth and Display Software Overview

3.2.2.1. Implementation

The coaching software will receive data from the microcontroller via Bluetooth Low Energy (BLE) communication. For quick and efficient Bluetooth receiver development, the application will need to use a programming language with a library that has already implemented the Bluetooth protocol stack across the three major operating systems of Windows, OS X, and Linux. Taking into account the team's prior experience, three programming languages—C++, Python, and Javascript—were considered, with each having a few different Bluetooth libraries ranging in maturity and support. Although it is the most efficient of the three considerations, C++ was not chosen due to its slow development cycle and a lack of portability to the major operating systems, which would conflict with the design's manufacturability constraint. Comparatively, Python development is quick and it has frameworks for Windows, OS X, and Linux; however, the frameworks analyzed were neither mature nor supported by a substantial community. Fortunately, Javascript met the search criteria of development speed, had documented Bluetooth platform compatibility, and had a very mature and actively maintained framework.

Node.js, the runtime environment for Javascript, will be used as the basis of the coaching application backend, hosting the backend processes. The Bluetooth interfacing process, which will use the BLE framework *noble* [17]. *Noble* will handle all radio transmission overhead such as message length, error checking, and acknowledgements.

For both real-time and full-swing modes, data will be received as a stream into a buffer. A new timestamp in the buffer will delimit a new frame. Once this new frame is identified, it will be processed for storage and display. In full-swing mode, the frame will be deserialized as an array of a single swing's data points; however, in real-time mode, a frame will be deserialized as a single data point. Deserializing all data members includes parsing the timestamp and the timestamp offset as well as converting the sensor data payload to decimal values. The coaching software will deduce and filter the frames that contain noise by measuring if their payload is outside an expected range. The expected range of values will be tweaked through testing once the prototype is complete.

Following the data frame parsing, the data will be stored in a SQLite database, which will be more than adequate for the application's storage needs and Javascript has SQLite bindings in its standard library. If the device is operating in real-time mode, the application will also pass the parsed frame's data directly to the display.

3.2.2.2. Interfacing

Node.js on the desktop will utilize the *noble* BLE library for interfacing to the microcontroller. It will first pair with the client, then always listen for incoming data. If the device is operating in real-time mode, the backend will receive frames containing single data points; in live-swing operating mode, it will instead receive a frame containing an array of the entire swing's data points. If the user changes the display mode, the backend application will transmit a message to the microcontroller indicating this change. Once acknowledged, the application will begin listening for new frames. Communication will occur using the GATT application layer protocol, which assigns metadata to each attribute that is used to describe the source of each sensor.

3.2.3. Coaching Software Frontend (Display)

According to the constraints, the display must be simple and provide the user with the insight to improve golf swing technique. To accomplish this, each sensor's signal will be overlaid on a graph over time. The user can toggle these graphs on or off to focus wrist movement, hand orientation, or hand velocity. The user will also have the option to zoom in on specific points in the swing. The display must also show real time and full swing graphing modes. For full-swing mode, the user will be able to choose between their most recent swing, or historical swings that are either show instantly or replayed. The Bluetooth pairing process will be exposed to the user through a settings view. The settings will also allow the user to show and manage memory filled with historical swings. Below in Figure 3.5 is a sketch of display.

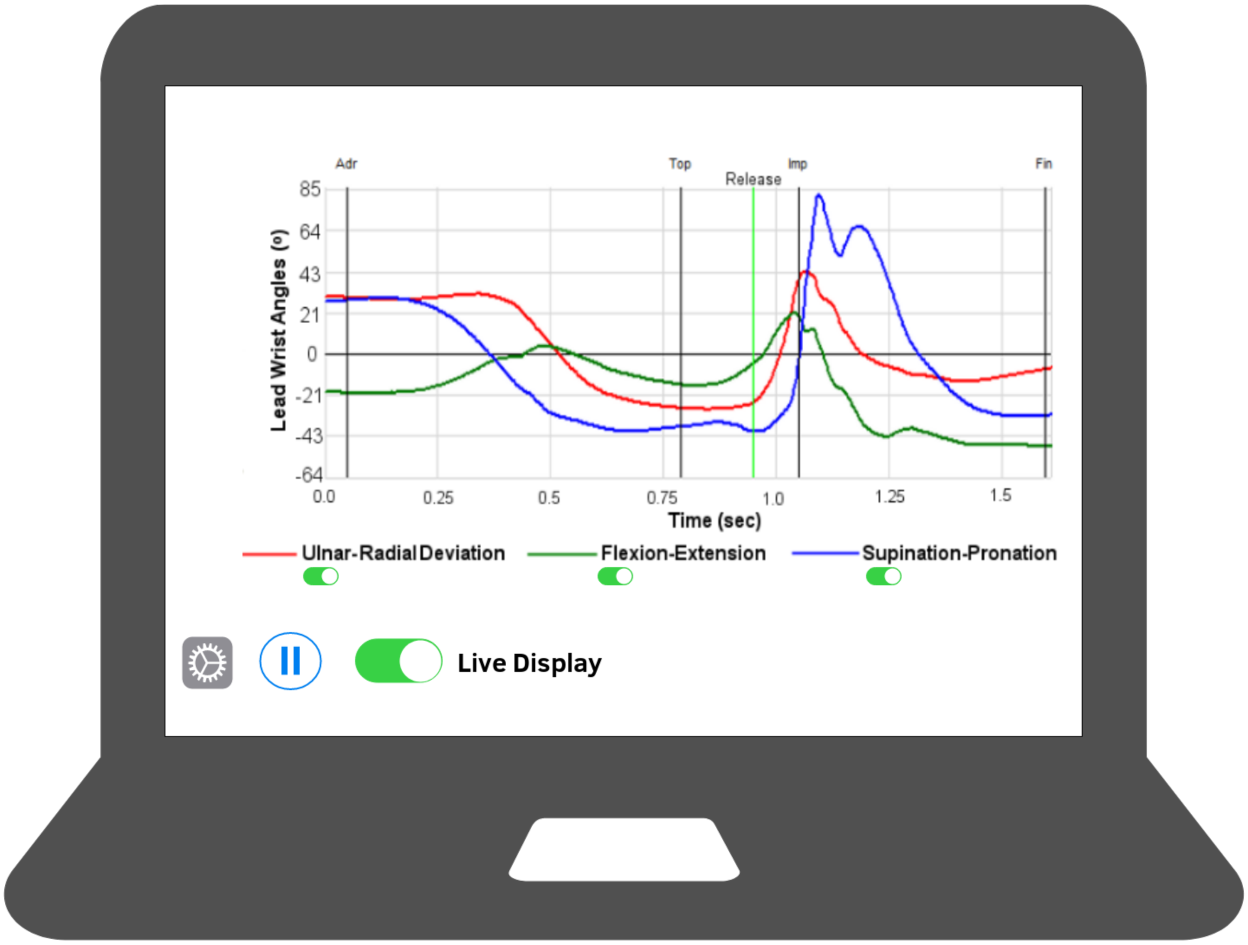

Figure 3.5. Mockup of Coaching Application Display

3.2.3.1. Implementation

For the GUI, there are three common options: a display in the web browser with AJAX, a display using Electron, or a GUI made in Python using the QT framework. A web browser has the advantage of being portable, as well as powered with Javascript; however, it would require client/server communication overhead and extra development. A GUI in Python would also be portable and powerful, but the QT framework is proprietary and thus does not fulfill the open-source requirement of the manufacturability design constraint. On top of that, it would also require communication overhead between itself and the chosen *noble* library. The software will use an Electron application as it had all of the perks of the web page but did not require client/server communication overhead and delay.

Electron is an open-source, cross-platform, desktop application framework that combines the Chromium browser rendering engine and Node.js [18]. Using the React framework inside Electron, the application will respond to both real-time display frames and full-swing display frames [19]. Real-time display frames will be instant measurements containing one data point, while full-swing display frames will be a combination of measurements contained in an array that represents a snapshot of the user's whole swing. To accomplish the design constraint for display, these measurements will be displayed in graphs to the user to show hand pressure, hand acceleration, hand rotation, wrist flexion, and wrist extension over time. Figure 3.6 is a sketch of the display.

3.2.3.2. Interfacing

The display will receive sensor frames from the backend application using inter-process communication and the SQLite database. The display can query the database for historical swing values as well as receive updates from the backend server. Having both persistent storage and event-driven updates allows the user to examine swing data live or at a later time.

3.2.4. Software State Diagram

Figure 3.6 shows how the software application will function and conditions relevant to state changes.

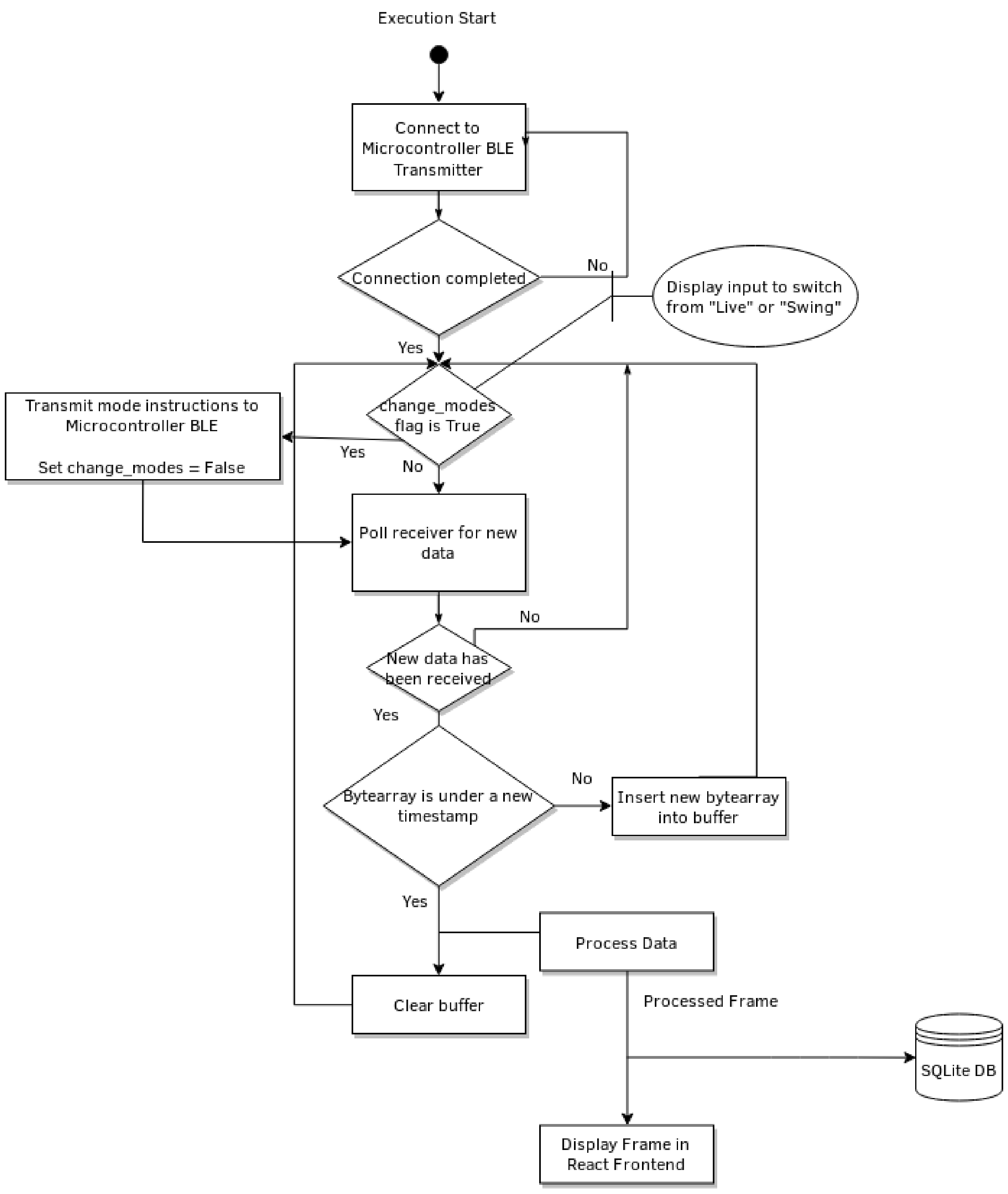
Execution Start
Connect to Microcontroller BLE Transmitter
Connection completed
No
Yes
Display input to switch from "Live" or "Swing"
Transmit mode instructions to Microcontroller BLE
Set change_modes = False
change_modes flag is True
Yes
No
Poll receiver for new data
No
New data has been received
Yes
Bytearray is under a new timestamp
No
Insert new bytearray into buffer
Yes
Process Data
Clear buffer
Processed Frame
SQLite DB
Display Frame in React Frontend

Figure 3.6. Software State Diagram

3.2.5. Use Cases

Figures 3.7 and 3.8 describe the sunny and rainy day use cases for Golf Glove, respectively. Sunny day operation occurs when the wrist-mounted controller and the computer application connect properly.

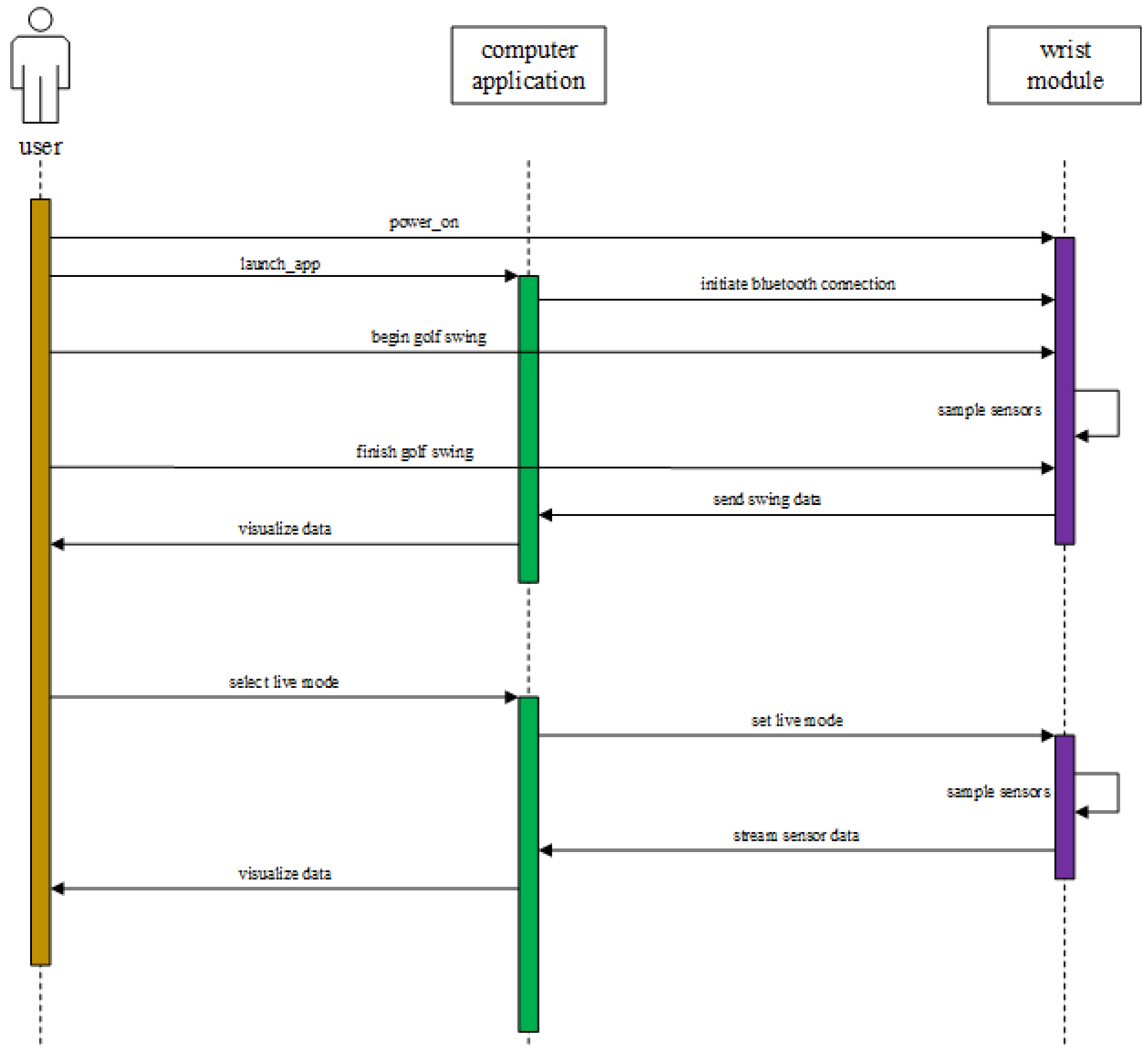

Figure 3.7. Sunny Day Use Case

Rainy day operation occurs when the wrist-mounted controller does not have adequate charge to power on or the Bluetooth connection between the wrist-mounted controller and coaching application cannot be established.

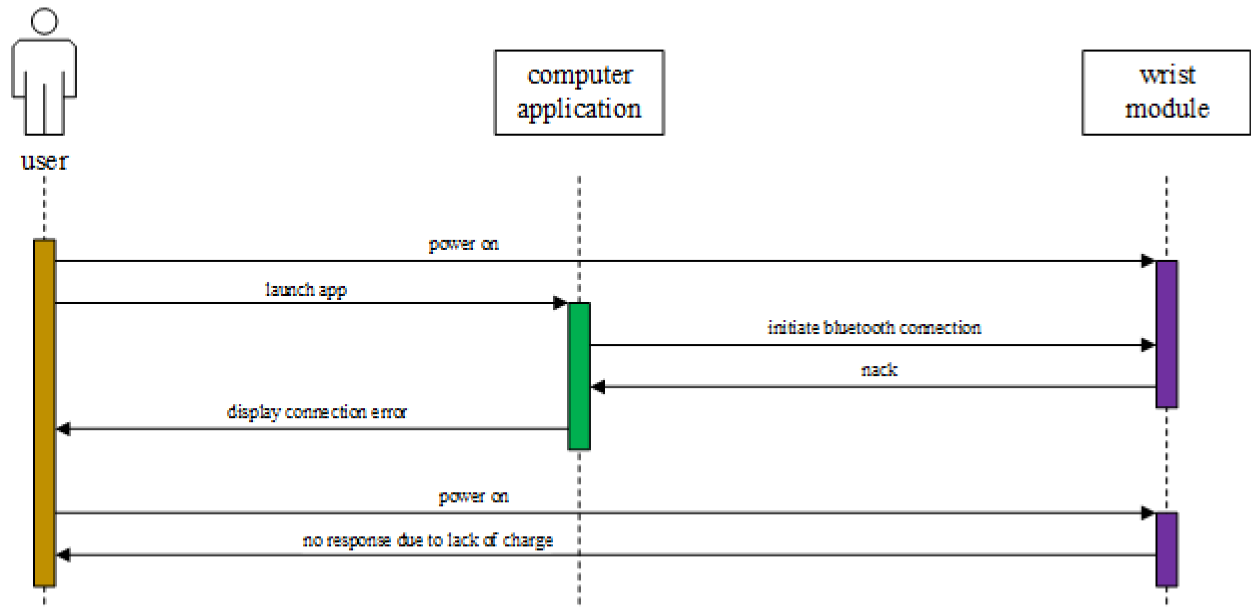

Figure 3.8 Rainy Day Use Case

3.3. References

[1] "Liquid Wire Technology," Liquid Wire. [Online]. Available: https://liquidwire.io/technology. [Accessed: 26-Oct-2018].

[2] "Hand Grip Strength: age and gender stratified normative data in a population-based study," BMC Res Notes, Apr. 2011.

[3] E. Komi, J. Roberts, and S. Rothberg, "Measurement and analysis of grip force during a golf shot," Proceedings of the Institution of Mechanical Engineers Part P Journal of Sports Engineering and Technology, Jun. 2008.

[4] "FS R ® 400 Series Data Sheet," 2017. [Online]. Available: https://cdn2.hubspot.net/hubfs/3899023/Interlinkelectronics November2017/Docs/Datasheet_FSR.pdf. [Accessed: 25-Oct-2018].

[5] Adafruit Industries, "Pressure-Sensitive Conductive Sheet (Velostat/Linqstat)," adafruit industries blog RSS. [Online]. Available: https://www.adafruit.com/product/1361. [Accessed: 26-Oct-2018].

[6] "Acceleration," Quintic Sports. [Online]. Available: https://www.quinticsports.com/wp-content/uploads/2016/06/Case-Study-12-Acceleration.pdf. [Accessed: 25-Oct-2018].

[7] Loopy, "Golf Swing Sequence and Timing – The Downswing," Golf Loopy - Play Your Golf Like a Champion, 14-Oct-2015. [Online]. Available: http://www.golfloopy.com/golf-swing-sequence-and-timing-downswing/. [Accessed: 25-Oct-2018].

[8] "LSM9DS1 Datasheet." [Online]. Available: https://www.st.com/resource/en/datasheet/lsm9ds1.pdf. [Accessed: 25-Oct-2018].

[9] "Bosch Sensortec BMX055 Datasheet." [Online]. Available: https://ae-bst.resource.bosch.com/media/_tech/media/datasheets/BST-BMX055-DS000-02.pdf. [Accessed: 25-Oct-2018].

[10] "mCube MC6479 Preliminary Datasheet." [Online]. Available: https://www.mouser.com/datasheet/2/693/MC6470-Preliminary-Datasheet-APS-048-0033v1.6--1489118.pdf. [Accessed: 25-Oct-2018].

[11] "Enhanced Low Power, BR/EDR/BLE Bluetooth 5.0 SOC Datasheet," 12-Jul-2018. [Online]. Available: http://www.cypress.com/file/414181/download. [Accessed: 25-Oct-2018].

[12] "Bluetooth® Low Energy (BLE) SoC." [Online]. Available: http://ww1.microchip.com/downloads/en/DeviceDoc/IS1870_71-Bluetooth-Low-Energy-BLE-SoC-DS60001371E.pdf. [Accessed: 25-Oct-2018].

[13] "Li-Polymer Battery Technology Specification," Adafruit Industries. [Online]. Available: https://cdn-shop.adafruit.com/product-files/1578/C1854 PKCell Datasheet Li-Polymer 503035 500mAh 3.7V with PCM.pdf. [Accessed: 25-Oct-2018].

[14] "Robot?," Newegg- Computer Parts, Laptops, Electronics, HDTVs, Digital Cameras and More![Online]. Available: https://www.newegg.com/Product/Product.aspx?Item=9SIA4SR6YH7231&ignorebbr=1&nm_mc=KNC-GoogleMKP-PC&cm_mmc=KNC-GoogleMKP-PC-_-pla-_-AT - Batteries & Accessories - Replacement-_-9SIA4SR6YH7231&gclid=Cj0KCQjw08XeBRC0ARIsAP_gaQBDs4SNdrioxPFeVZ5y1T08FjWVodO-wwc_uowzROIrS8EiuxSbWOAaAlTUEALw_wcB&gclsrc=aw.ds. [Accessed: 26-Oct-2018].

[15] "MX2400 Alkaline-Manganese Dioxide Battery Datasheet." [Online]. Available: http://professional.duracell.com/downloads/datasheets/product/Ultra Power/Ultra-Power_AAA_MX2400.pdf. [Accessed: 25-Oct-2018].

[16] "Manganese Dioxide Lithium Coin Batteries: Individual Specifications." [Online]. Available: https://www.alliedelec.com/m/d/6ec5c9a8833414eb34f877f612fc890f.pdf. [Accessed: 25-Oct-2018].

[17] "noble/noble," GitHub, 08-Jun-2018. [Online]. Available: https://github.com/noble/noble. [Accessed: 26-Sep-2018].

[18] "Electron," Electron. [Online]. Available: https://electronjs.org/. [Accessed: 26-Sep-2018].

[19] "React – A JavaScript library for building user interfaces," React. [Online]. Available: https://reactjs.org/. [Accessed: 26-Sep-2018].