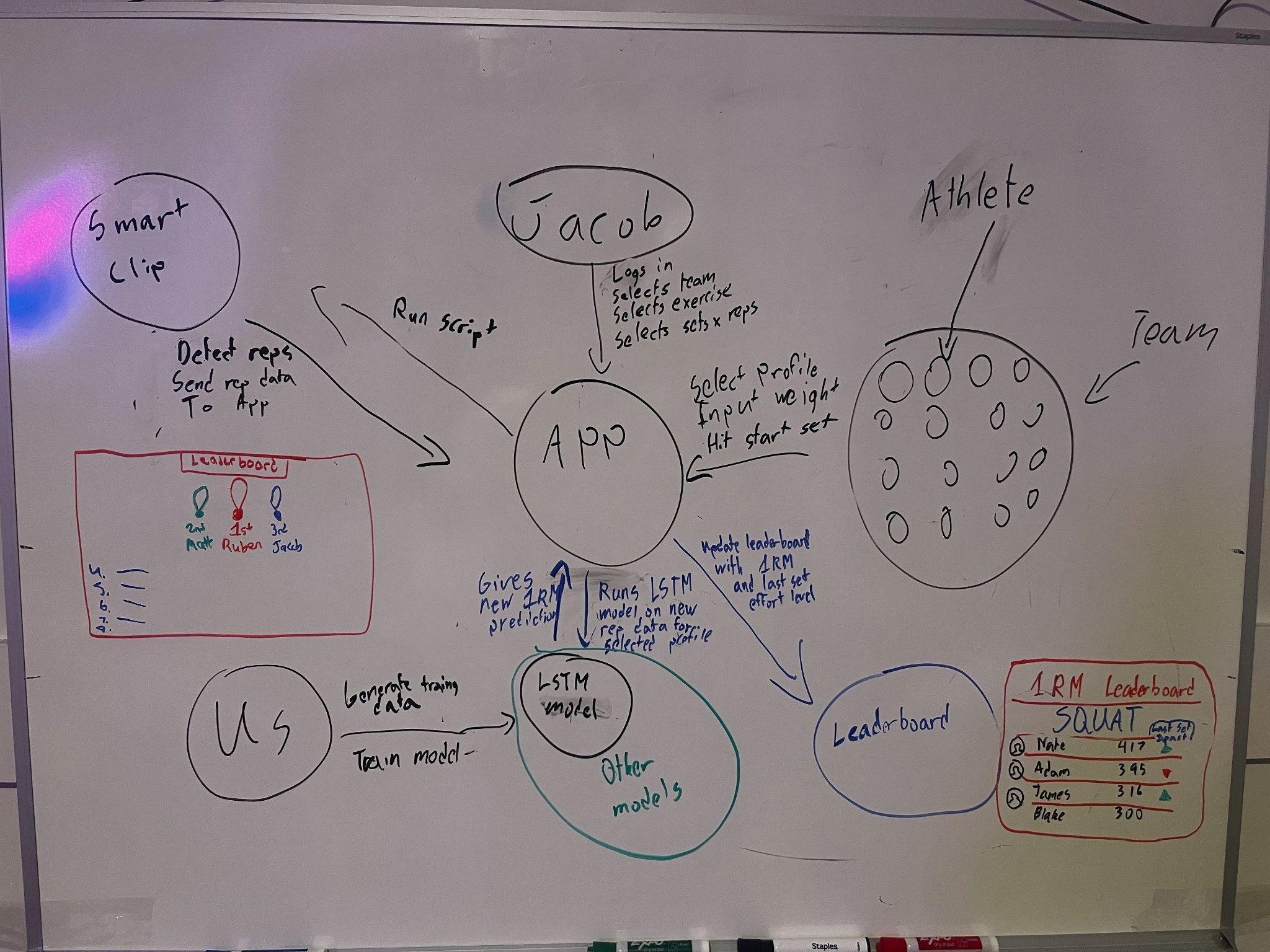
3. Best way of classifying individual reps (using acceleration, velocity, etc.)

2. Getting velocity data on arduino



## **1. Hardware & System Overview**

1. **Smart Clip (Arduino + IMU)**
   * **Sensors**: Built‐in IMU on the Nano 33 (accelerometer, gyroscope).
   * **Battery‐Powered**: Must be low‐power and efficient.
   * **Primary Tasks**:
     + Collect motion data during each rep.
     + Perform basic on‐device filtering/feature extraction (to reduce BLE bandwidth).
     + Transmit extracted features (and/or partially processed raw data) via Bluetooth to the iPad.
2. **iPad App**
   * **User Interface**:
     + Trainer chooses team/exercise/set–rep scheme.
     + Athlete selects their profile, enters the load for the set.
   * **Data Processing**: Receives real‐time data from the clip, runs a lightweight ML or calculation to produce an immediate (or near‐immediate) 1RM estimate and “effort” rating.
   * **Communication**: Sends data/estimates to the backend (could be a local server or a cloud DB).
3. **Backend Database / Server**
   * Stores all session data (reps, loads, velocity features, user IDs).
   * Runs or hosts more advanced or periodic ML models (long‐term analytics, advanced personalization).
   * Provides updated 1RM estimates or recommended intensities back to the iPad if needed.
4. **Leaderboard / Team View**
   * A display (within the iPad app or a web interface) that shows the updated 1RM estimates for each athlete in real time.

## **2. Step‐by‐Step Data Flow**

### **Step 1: Athlete & Exercise Selection**

1. **Trainer** sets up the exercise station on the iPad:
   * Chooses the **team** (group of athletes).
   * Selects the **exercise** (e.g., bench press, squat).
   * Defines the **planned set–rep scheme** (if relevant).
2. **Athlete** comes to the station:
   * Chooses their **profile** from a list on the iPad.
   * Inputs the **weight** they are about to lift.

*Why This Matters:*This step ties the upcoming sensor data to a **specific athlete**, **exercise**, and **load** so the system can properly store and label it.

### **Step 2: Data Collection on the Smart Clip (Arduino)**

1. **IMU Recording**
   * The Nano 33 Sense Rev2 collects accelerometer + gyroscope data continuously from the moment the athlete begins the set.
   * Sampling Rate (e.g., 100–200 Hz) is high enough to capture rep dynamics but balanced for battery life.
2. **On‐Device Signal Processing** (Optional / Recommended)
   * **Sensor Fusion Filter**: Madgwick/Kalman to estimate orientation and subtract gravity accurately.
   * **Velocity Calculation**: Integrate net acceleration over time to get velocity, corrected for orientation.
   * **Rep Detection**:
     + Identify concentric/eccentric phases or zero‐crossings in velocity to segment each repetition.
     + Could track how many reps have been detected so far.
3. **Feature Extraction**
   * For each rep, compute minimal or “first‐level” features (to reduce BLE bandwidth). Examples:
     + **Peak Concentric Velocity**
     + **Mean Concentric Velocity**
     + **Rep Duration** or Time Under Tension
     + **Velocity Drop** from prior rep (if capturing multiple reps in a set)
     + (Optional) **Segmented Velocity** if you do 4–10 bins, but watch memory usage.
4. **Buffering & Preparation for Transmission**
   * The Arduino aggregates rep features (and possibly some raw data slices if needed) until the set is complete or the athlete racks the bar.
   * Minimal overhead is key to ensure quick Bluetooth transfer.

### **Step 3: Transmission of Set Data to the iPad**

1. **Bluetooth LE (BLE) Communication**
   * Once the athlete finishes the set, the Arduino sends a JSON or binary payload to the iPad.
   * Payload includes:
     + **Athlete ID** (from iPad pairing context)
     + **Exercise Type**
     + **Load (kg/lbs)**
     + **Number of Reps**
     + **Rep‐by‐rep features** (peak velocity, mean velocity, rep duration, etc.)
2. **iPad Receipt & Parsing**
   * The iPad listens for the BLE packet, parses the JSON/binary structure into a local data object.

### **Step 4: Real‐Time 1RM & Effort Estimation on the iPad**

Once the iPad has the **rep features** and the **known load**, it can run a lightweight ML or formula‐based estimation:

1. **Immediate (Physics/Formula‐Based) Calculation** *(Optional but useful)*
   * For quick user feedback, use a known velocity–load formula (e.g., a simple derivation from González‐Badillo or a minimal velocity threshold approach).
   * E.g., 1RM^formula=function(load,peakVelocity) \widehat{\mathrm{1RM}}\_{\text{formula}} = \text{function}(\text{load}, \text{peakVelocity})1RMformula​=function(load,peakVelocity)
2. **Short‐Term ML Model** (On the iPad)
   * **Model Type**: Could be a small **Gradient Boosted Regression** or **Neural Network**.
   * **Features**: The extracted rep features (peak velocity, mean velocity, velocity drop, rep duration, etc.). Possibly athlete’s recent historical 1RM or overall experience level as an input.
   * **Output**: More refined 1RM estimate for that set.
3. **Effort (RPE) or “%1RM” Estimate**
   * The iPad can also compute how close the load was to the predicted 1RM. If load is 80 kg and 1RM estimate is 100 kg, then the user lifted ~80% 1RM.
   * Display that in real time: “Estimated 80% of your max,” or “Your last rep was near 9 RPE.”
4. **Display to Athlete**
   * The athlete sees an updated 1RM estimate, plus an “effort” rating or predicted %1RM.
   * They can decide to adjust load for their next set if needed.

### **Step 5: Sync to Backend & Database Update**

1. **Data Packaging**
   * The iPad packages all raw or feature data + the iPad’s immediate 1RM estimate + any relevant meta info (athlete ID, exercise, timestamp).
2. **Upload to Backend**
   * A simple REST/GraphQL endpoint or WebSocket to your backend server.
   * The backend stores:
     + **Set–Level Data**: (weight, number of reps, time, iPad’s immediate 1RM estimate).
     + **Rep–Level Features**: (peak velocity, mean velocity, velocity drop, etc.).
3. **Database**
   * The backend writes this data into a user‐specific table or document.
   * This data is then ready for **long‐term analytics** or advanced modeling.

### **Step 6: Long‐Term Analytics / Advanced Modeling (Backend)**

Periodically (nightly or weekly), the backend can run heavier ML tasks:

1. **Aggregated Historical Data**
   * For each athlete, gather all prior sets (including loads, rep features, iPad predictions, etc.).
   * Potentially incorporate any actual 1RM tests if you have them, or near‐max lifts for ground truth.
2. **Advanced Model**
   * **Model Type**: LSTM, Transformer, or Bayesian Hierarchical Model to discover user‐specific velocity–load relationships over time.
   * **Input**: Time‐series of sets (each set’s features) plus the athlete’s ID and possibly anthropometrics if known.
   * **Output**: A refined or personalized 1RM curve, including an uncertainty band (confidence interval).
3. **Update Athlete Profile**
   * The new, refined 1RM estimate is stored in the database as the “current best guess” of the athlete’s max.
   * This can get sent back to the iPad app (e.g., next time the app syncs, it sees the updated 1RM).

### **Step 7: Leaderboard & Team View**

1. **Leaderboard Construction**
   * The backend aggregates all athletes’ latest 1RM estimates.
   * Could rank them within the team or show progress bars, percentages, or other comparative data.
2. **iPad Display**
   * The trainer or the athletes can tap “Leaderboard” to see real‐time or recent 1RM updates.
   * Encourages friendly competition and shows who is improving.

## **3. Summary of Models Needed**

Below is a concise overview of **which models** are used and **where**:

1. **On‐Device Rep Detection / Filtering** (Arduino)
   * **Type**: Non‐ML or very simple rule‐based approach (peak detection, thresholding).
   * **Purpose**: Real‐time segmentation of reps, gravity compensation, velocity calculation.
   * **Why**: Minimizes data streaming, ensuring only essential features are sent to the iPad.
2. **Immediate “Physics” Formula** (iPad)
   * **Type**: Analytical formula (e.g., velocity–load relationship from the literature).
   * **Purpose**: Provide instant feedback.
   * **Why**: Very fast, no heavy computation needed, baseline reference.
3. **Short‐Term Regression / GBM** (iPad)
   * **Type**: A small, efficient regression model (e.g., gradient‐boosted trees).
   * **Input**: Rep features (peak velocity, mean velocity, time under tension, etc.) + load. Possibly user’s historical 1RM if known.
   * **Output**: A more refined 1RM estimate.
   * **Why**: Allows customization to user specifics better than a blanket formula. Quick enough to run on the iPad CPU.
4. **Long‐Term LSTM / Transformer** (Backend)
   * **Type**: Time‐series or sequence model.
   * **Input**: The athlete’s entire history of sets (possibly tens or hundreds of sessions).
   * **Output**: A personalized 1RM estimate with an uncertainty interval.
   * **Why**: Over time, more advanced models can tease out patterns from large amounts of data, adapt to the athlete’s changes in strength, detect plateaus, etc.

## **4. Putting It All Together**

1. **Athlete logs in, selects load, performs a set.**
2. **Arduino** collects IMU data, filters, detects reps, extracts minimal velocity features.
3. **Arduino** sends these features via BLE to the **iPad** at set completion.
4. **iPad** runs:
   1. **Quick formula** for immediate “ballpark” 1RM.
   2. **Short‐Term Regression** for a better near‐instant estimate.
   3. Displays updated 1RM and %1RM (effort level) to the athlete.
5. **iPad** sends the set data + estimates to the **backend**.
6. The **backend** stores everything in the database and periodically runs an **advanced LSTM** model to refine each athlete’s 1RM.
7. The refined 1RM populates the **leaderboard** and is available for future sessions (the iPad can pull down the updated figure).

This **integrated approach** balances **real‐time feedback** (on the iPad) with **rich historical modeling** (in the backend). Athletes and trainers see immediate results, while the system grows more accurate and personalized over time via advanced analytics in the cloud. This ensures a seamless user experience, robust data tracking, and continually improving 1RM estimates.

OpenBarbell (open source code): [squatsandsciencelabs/OpenBarbell-V3](https://github.com/squatsandsciencelabs/OpenBarbell-V3?tab=readme-ov-file#OpenBarbell%20V2)

Necessary Hardware

Arduino Nano 33 BLE Sense- <https://www.amazon.com/Nano-BLE-Sense-Rev2-ABX00069/dp/B0BQHXVSGM/ref=asc_df_B0BQHXVSGM?mcid=f8a43e1b67ad3a9aacdbfa9cb290e320&hvocijid=15449847686793993850-B0BQHXVSGM-&hvexpln=73&tag=hyprod-20&linkCode=df0&hvadid=721245378154&hvpos=&hvnetw=g&hvrand=15449847686793993850&hvpone=&hvptwo=&hvqmt=&hvdev=c&hvdvcmdl=&hvlocint=&hvlocphy=9060451&hvtargid=pla-2281435179338&psc=1>

TP4056 LiPo Charging Module- <https://www.amazon.com/HiLetgo-Lithium-Charging-Protection-Functions/dp/B07PKND8KG/ref=asc_df_B07PKND8KG?mcid=582532d88b30334da0eb72382cf0d57a&hvocijid=806802862098911036-B07PKND8KG-&hvexpln=73&tag=hyprod-20&linkCode=df0&hvadid=721245378154&hvpos=&hvnetw=g&hvrand=806802862098911036&hvpone=&hvptwo=&hvqmt=&hvdev=c&hvdvcmdl=&hvlocint=&hvlocphy=9060451&hvtargid=pla-2281435178618&psc=1>

Breadboard- <https://www.amazon.com/ELEGOO-tie-points-breadboard-Arduino-Jumper/dp/B01EV640I6>

Jumper Wires - <https://www.amazon.com/California-JOS-Breadboard-Optional-Multicolored/dp/B0BRTJXND9/ref=asc_df_B0BRTJXND9?mcid=c0c5869d519b33f28fdb4e6a5bfc68e5&hvocijid=11941716491220606205-B0BRTJXND9-&hvexpln=73&tag=hyprod-20&linkCode=df0&hvadid=721245378154&hvpos=&hvnetw=g&hvrand=11941716491220606205&hvpone=&hvptwo=&hvqmt=&hvdev=c&hvdvcmdl=&hvlocint=&hvlocphy=9060451&hvtargid=pla-2281435178298&psc=1>

Electrical Tape- <https://www.amazon.com/Duck-373447-Professional-Electrical-0-75-Inch/dp/B007JSGNWU/ref=asc_df_B007JSGNWU?mcid=51d29eba7d3235b8a70ea6387b8c3970&hvocijid=15079179560037877208-B007JSGNWU-&hvexpln=73&tag=hyprod-20&linkCode=df0&hvadid=721245378154&hvpos=&hvnetw=g&hvrand=15079179560037877208&hvpone=&hvptwo=&hvqmt=&hvdev=c&hvdvcmdl=&hvlocint=&hvlocphy=9060451&hvtargid=pla-2281435179338&th=1>

Multimeter- <https://www.amazon.com/AstroAI-Digital-Multimeter-Voltage-Tester/dp/B01ISAMUA6?source=ps-sl-shoppingads-lpcontext&ref_=fplfs&smid=A2NOFZGOKNP3PJ&gQT=1&th=1>

3.7V LiPo Battery (500–1000mAh with JST connector)- <https://www.amazon.com/1000mAh-battery-Rechargeable-Lithium-Connector/dp/B07BTV3W87/ref=sr_1_1_sspa?crid=2UMOO02080W7G&dib=eyJ2IjoiMSJ9.Ygr3Lni9SNbjOgH-7HHshvzwe0sHB_e3yOiPAwbQ3ez9P9LPHp6fSyKPl2NzcWjk_3wUaSeqipwHjxYAZxPsTjcHFEJi4HNjFbPAjggJeqeDbbU5utYuoaBqztzAs7cXQYwzXbIJHJijOXL-1OV9G-dGNpS_v4t74L25OWCahyesE9CGd89tpKL5LyKQO8HIddEjiEjfZdWl9bzImdzK5xovD2fbGFAZd9ABYoET1Upe3SZs2qlKzHGX7afUWuvTR5WUblnv2P9VuzmNC9uHZ5sivRvjztilnR-iUjlMWlk.7BX4mb8LxGq0Vb3b4MLlYnnVTeB31CUJ2zSSqR99CC8&dib_tag=se&keywords=3.7v+lipo+1000mah&qid=1737092727&sprefix=3.7v+lipo+1%2Caps%2C141&sr=8-1-spons&sp_csd=d2lkZ2V0TmFtZT1zcF9hdGY&psc=1>

MCP1700 for 3.7v to 3.3v- <https://www.amazon.com/MCP1700-3302E-MCP1700-Microchip-Voltage-Regulator/dp/B084LBDPC7/ref=asc_df_B084LBDPC7?mcid=3f9cb1325b2e35b78a7eb002f2a4c20f&hvocijid=5852398271468258187-B084LBDPC7-&hvexpln=73&tag=hyprod-20&linkCode=df0&hvadid=721245378154&hvpos=&hvnetw=g&hvrand=5852398271468258187&hvpone=&hvptwo=&hvqmt=&hvdev=c&hvdvcmdl=&hvlocint=&hvlocphy=9016852&hvtargid=pla-2281435176898&psc=1>

1µF Ceramic Capacitor- <https://www.amazon.com/Cermant-Multilayer-Monolithic-Ceramic-Capacitor/dp/B0D2H87Q32/ref=sr_1_4?dib=eyJ2IjoiMSJ9.nl4sOcJ7qgTAqrB0l60YMNaCeAbs2C1PFsr-ElGbAcNqnCBNma9q-HzHP1wHeJNeeQg00DDlp-D0RyDvv9xLzzsR2a1qqYRV6ZL-7FwQCWkoslHwtcNkfLcpvVqfCO1nudr6LwkuRkiCOwC8Zf2rQ3rYDpay6BfvSKQOSIEwqaveSa9KjNJA0Qgm1hsp1GwejCYuUNalV23DqNjFX5HVzbvvwAsvYwdNNhbdTJ9J0b0.xAzb9fJsREk8Kf9PiBdszwn-OhavTOqiqNyo4-Ybkt4&dib_tag=se&keywords=ceramic%2Bcapacitor%2B1uf&qid=1737093679&sr=8-4&th=1>

JST Connector Kit- <https://www.amazon.com/300PCS-2-54mm-Connector-Housing-Adapter/dp/B0CB94MZ3M?th=1>