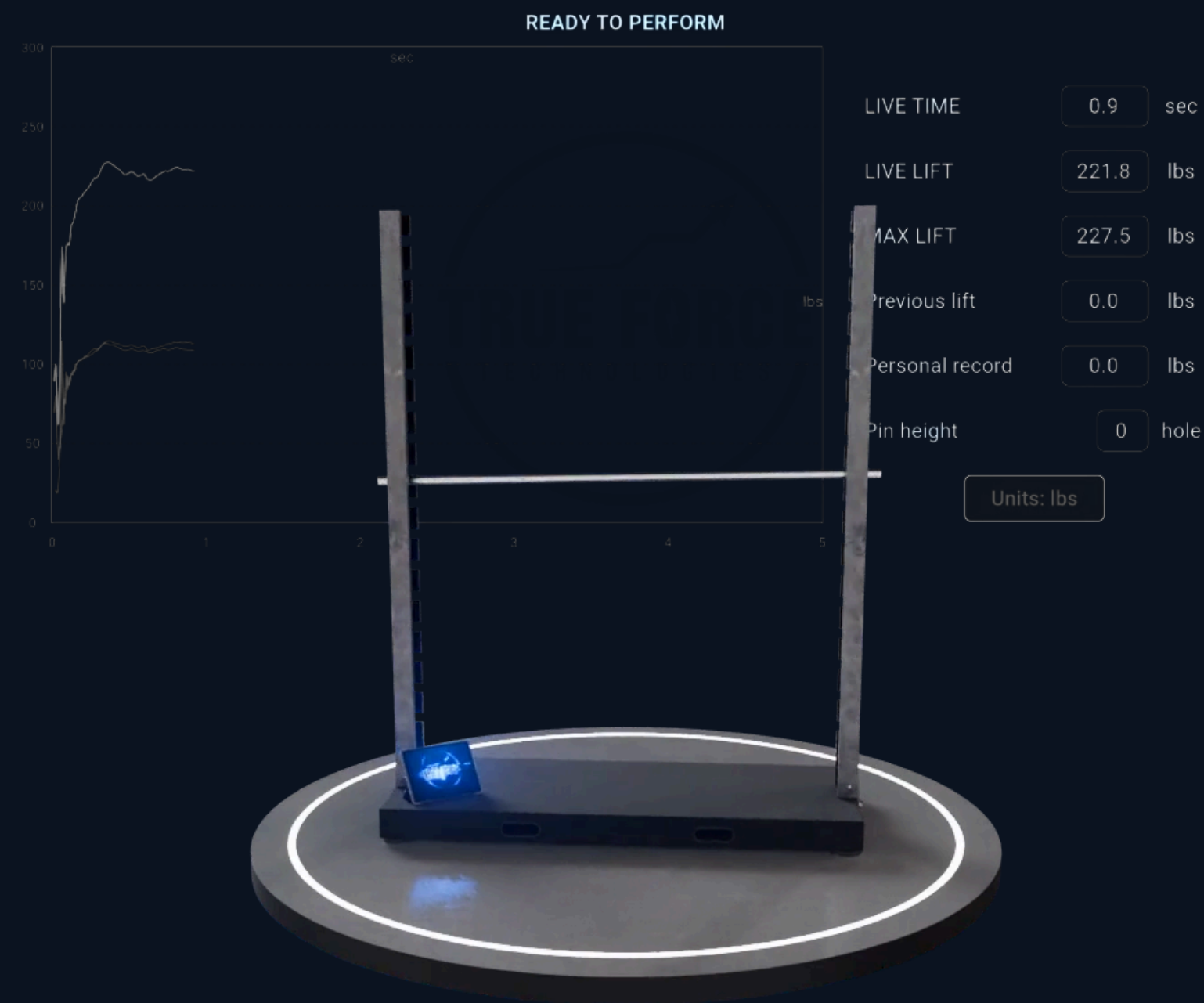


TRUE FORCE ARCADE RACK

Presented by
sdmay26-40



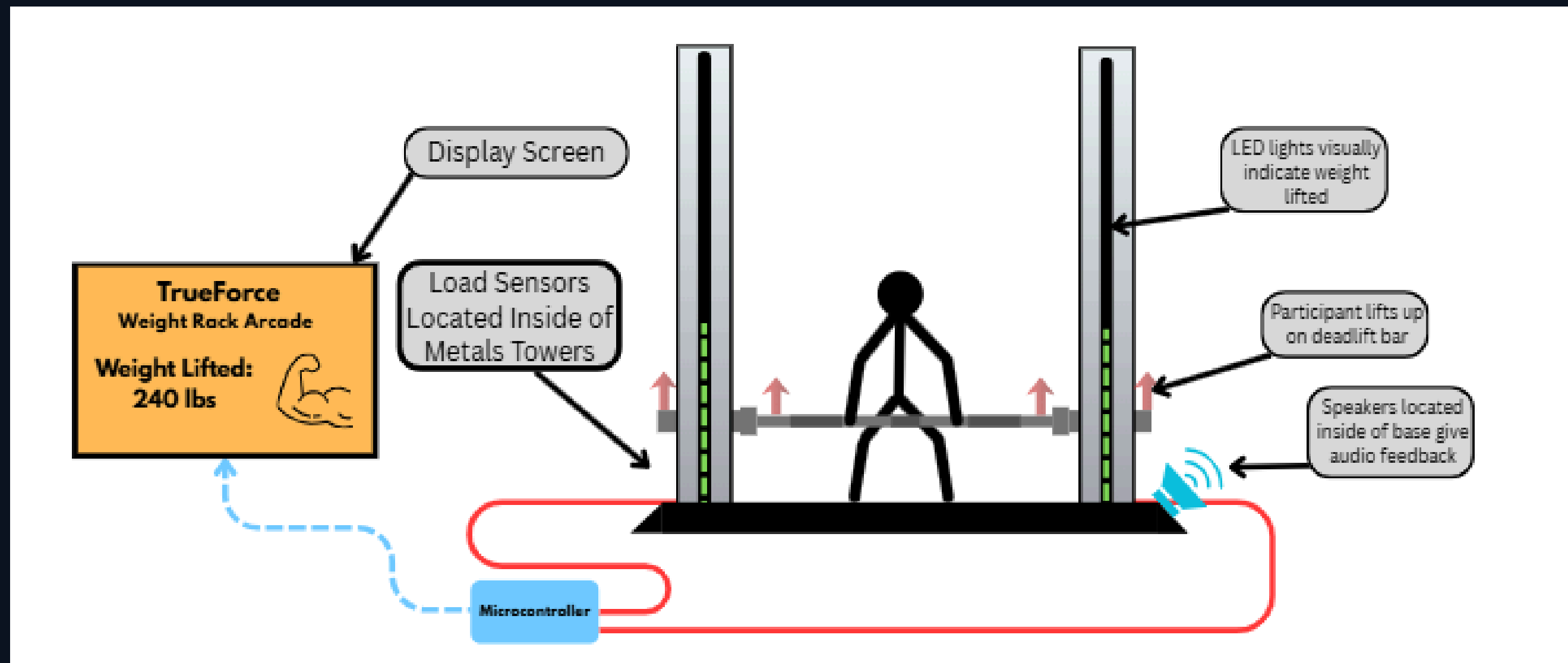
PROJECT OVERVIEW



ATHLETIC CLUB MEMBERS / BENCH PRESS / BENJAMIN

PROBLEM

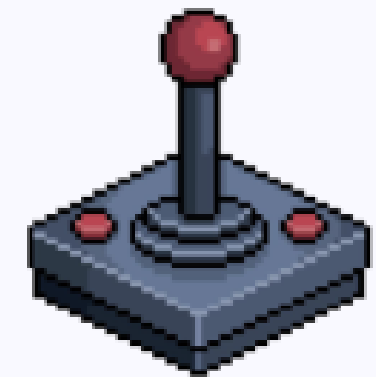
- The Force Rack already exists, but the client wanted more interactivity.
- The client raised the challenge, “What about corporate events?”



CONTEXT

Two Audiences:

- Public Events (fairs, festivals) → Arcade Deliverable
 - Highly detailed visuals (animations, LEDs, sound effects)
 - Fixed “set design” for quick walk-up play
 - Competitive features like leaderboards
- Corporate/Private Events → Custom Deliverable
 - Simplified visual layout
 - Options to personalize
 - Focus on brand connection



Arcade Mode



Custom Mode

REQUIREMENTS

CORE

- React App + Python App
- Bluetooth
- Animated Avatars
- Local Data Storage
- Units: lb/kg/Newtons/custom units

Arcade Deliverable

- Super-hero style avatars
- Local leaderboard with toggle visibility
- Male/Female selection
- Persistent leaderboard data
- Printable leaderboard summary

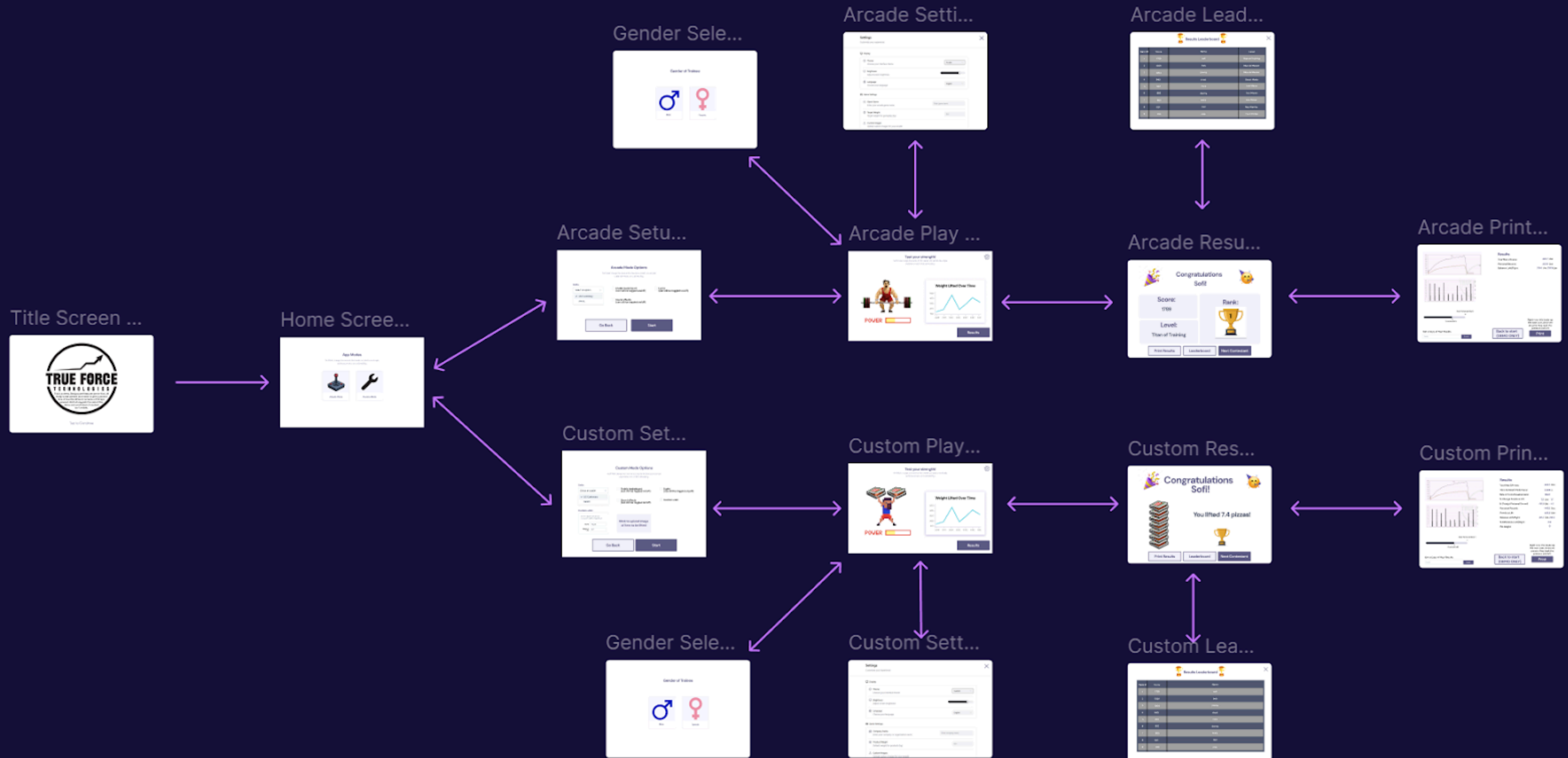
Output

- Clean and formatted print ready reports
- Optional email delivery

Custom Deliverable

- Admin-coach-player hierarchy
- Add custom units (pizzas, boxes, etc)
- Cloud based storage
- Printable and emailable reports
- No logins for players, only coaches

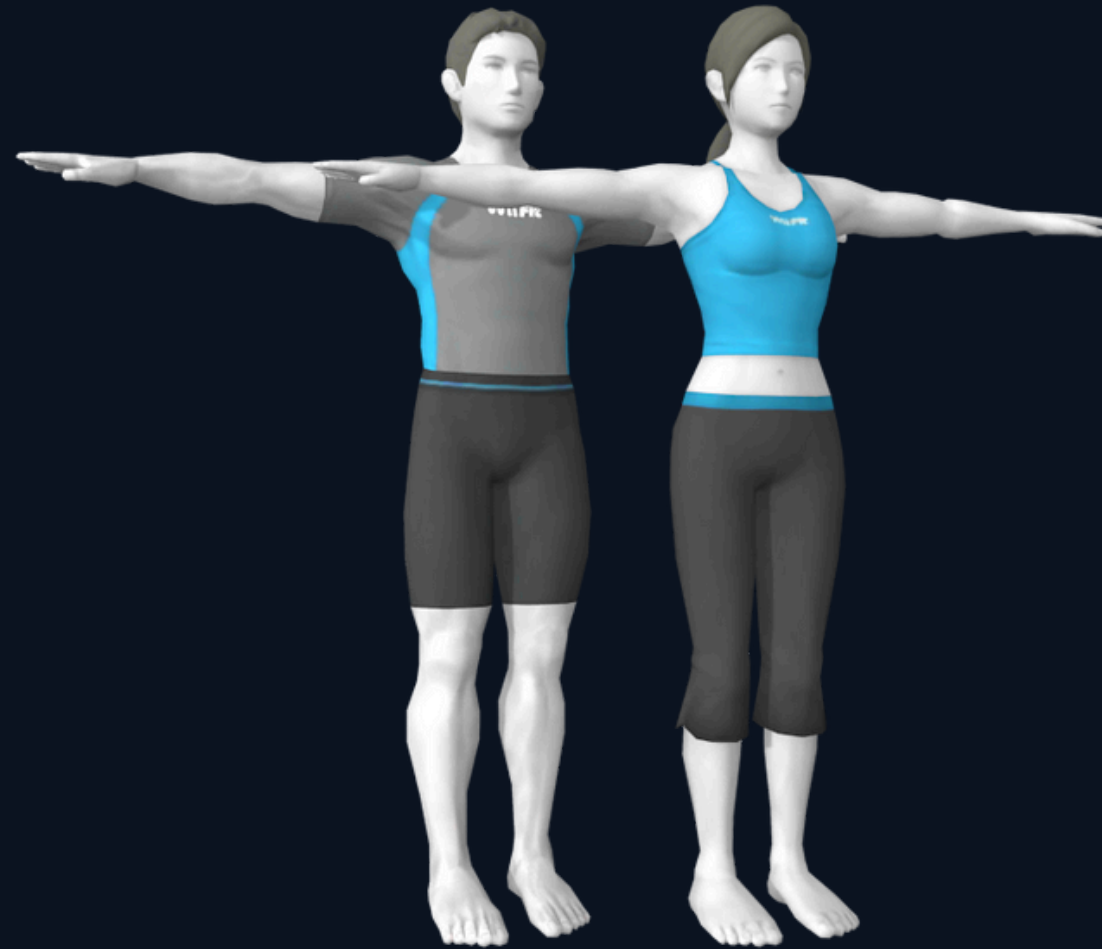
DESIGN - FIGMA APP FLOW



AVATARS AND ANIMATION



16-bit retro style



Neutral characters



Superhero aesthetic

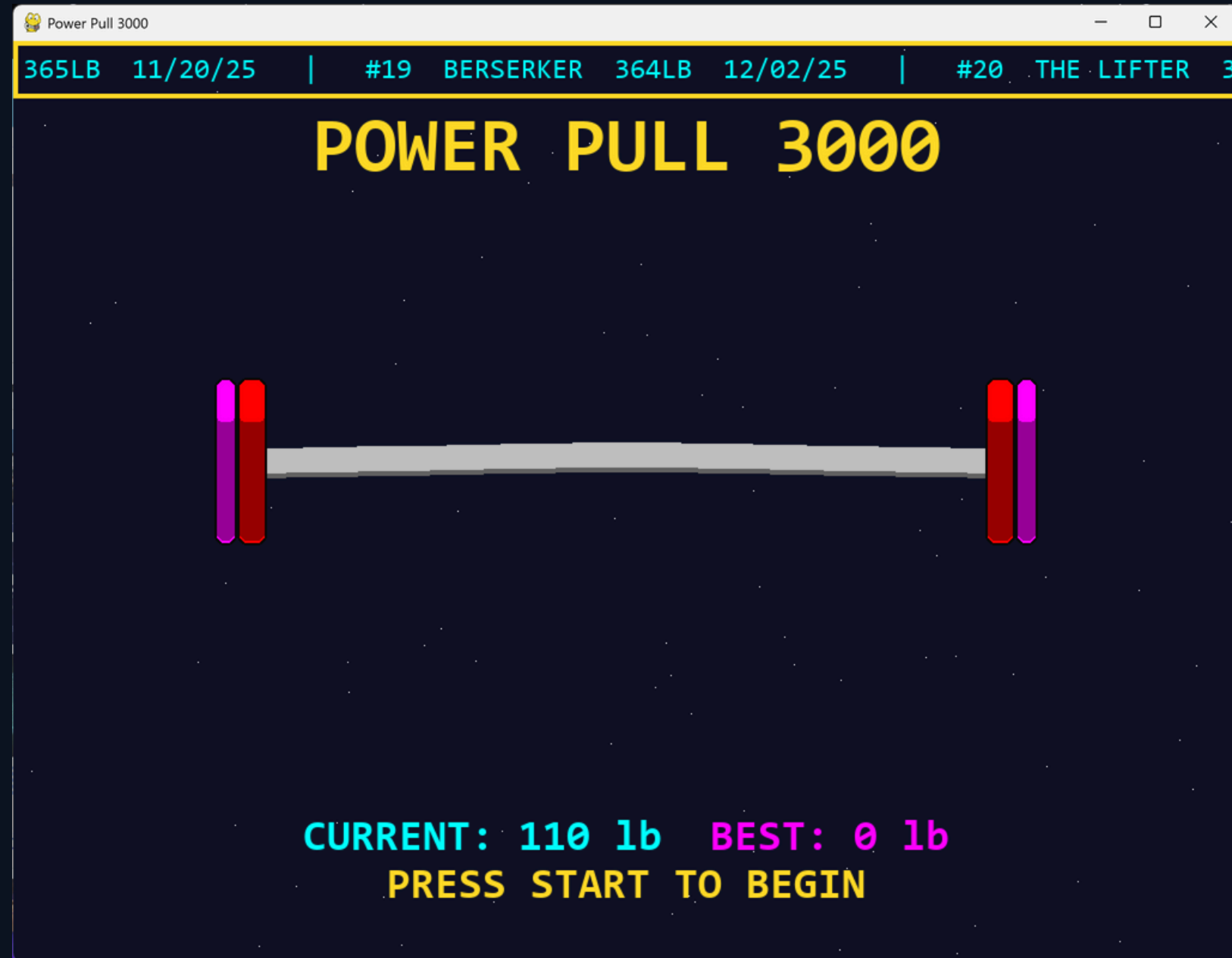
DESIGN - CODEBASE ARCHITECTURE

- Clean separation of concerns, reusable components, and code that's easy to understand and modify.
- Rebuilt architecture around reusable UI components, clean separation of concerns, and consistent design patterns.
- Screens now use a structured layout: UI components, smart containers, and future hooks for settings, animations, and image handling.
- Scalable foundation ready for backend integration, dynamic data loading, polished animations, sound, and full user experience enhancements.

PROTOTYPING / TESTING

- Built early prototypes to test Bluetooth stability and real time data flow
- Implemented custom and arcade prototypes to validate responsiveness
- Identified bugs in BlueTooth connectivity and refined architecture based on findings
- Iterative and modular testing using the Jest Library and manual testing

CURRENT PROGRESS



Arcade App



Custom App

CUSTOM APP



CUSTOM APP - SETTINGS

- Set custom units
- Set color palette
- Leaderboard toggle
- Sound effect toggle
- Lights toggle
- Custom image upload

Custom Mode Options

Subtitle

Units

✓ US Customary

Metric

Click to upload image of item to be lifted

Primary Color

#34495e

Secondary Color

#2c3e50

Tertiary Color (Accents)

#3cb371

☐ Enable leaderboard

☐ Sound effects

☐ Lights

☐ Custom units

Go Back

Start

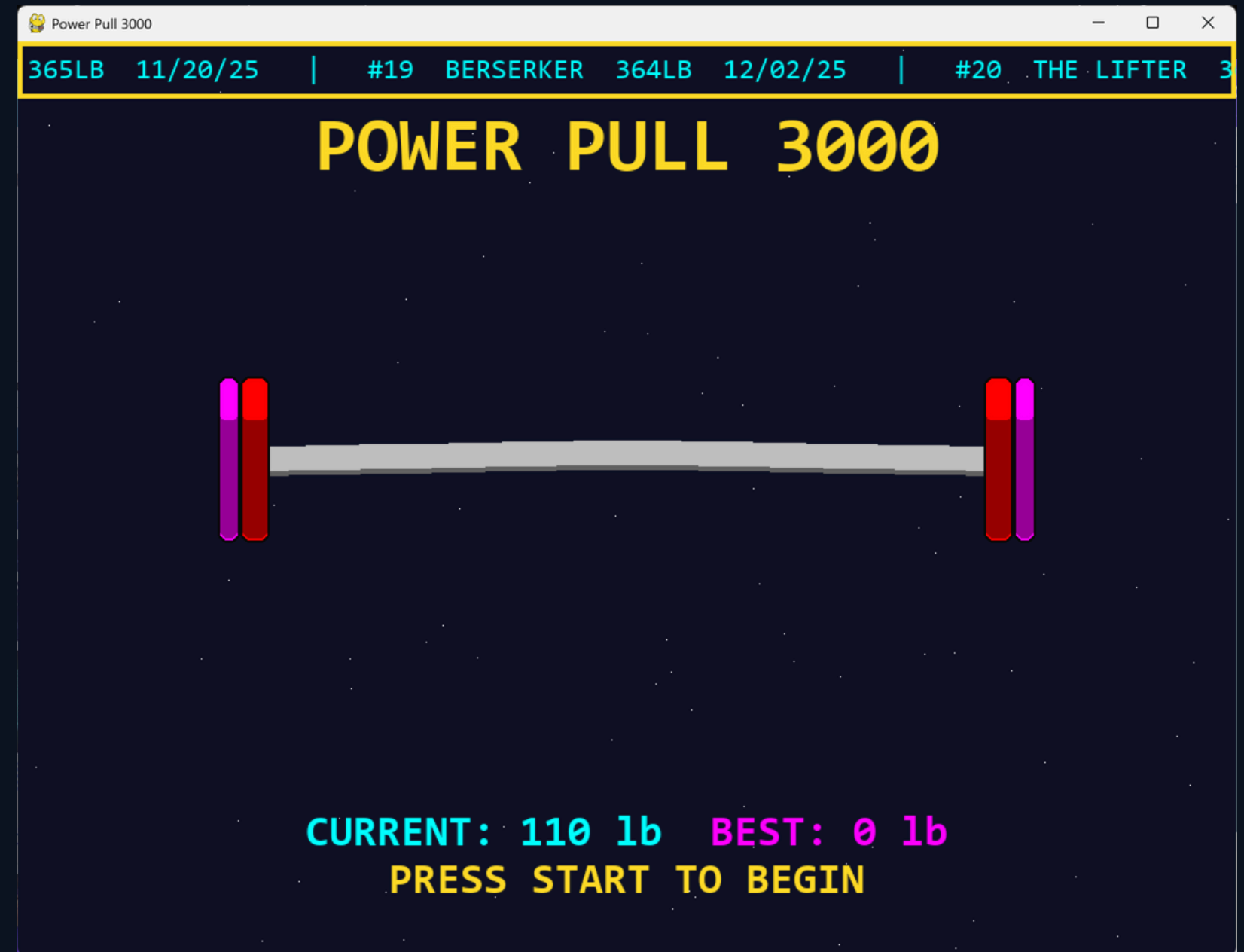
CUSTOM APP - GAME VIEW

- Dynamically changing graph to show weight lifted
- Current weight being lifted
- Max total weight lifted
- Strength bar that changes color based on level



ARCADE APP

- Why the switch from React to Python?
- Program simplicity, resilience, robust
- Easy to display and implement
- Raspberry Pi



ARCADE APP - GAME VIEW

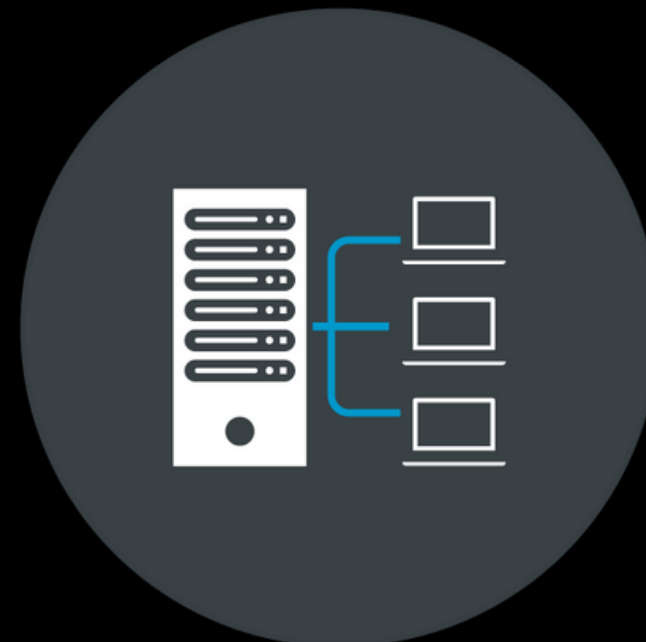
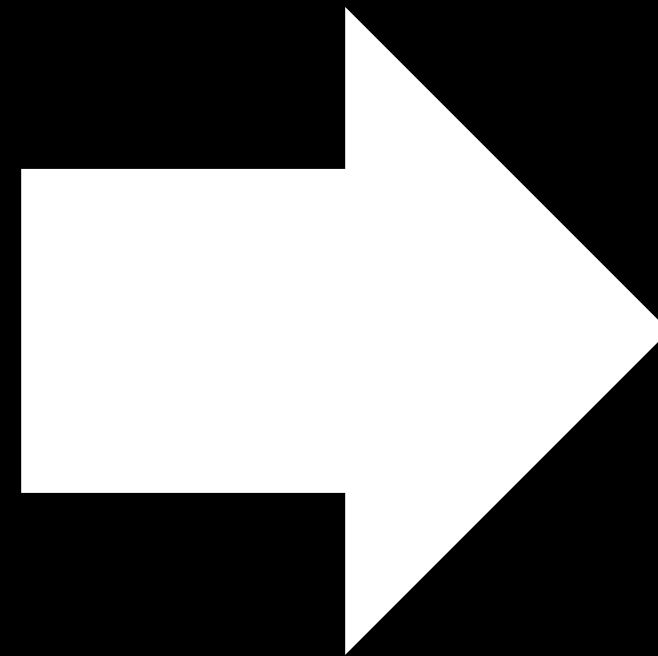
- Count down to initiate lift giving user time to set up
- Live responsive feed back to strength
- Count down to end game
- All fitted with sound effects and music for user enjoyment and emersion



ARCADE APP - DEMO

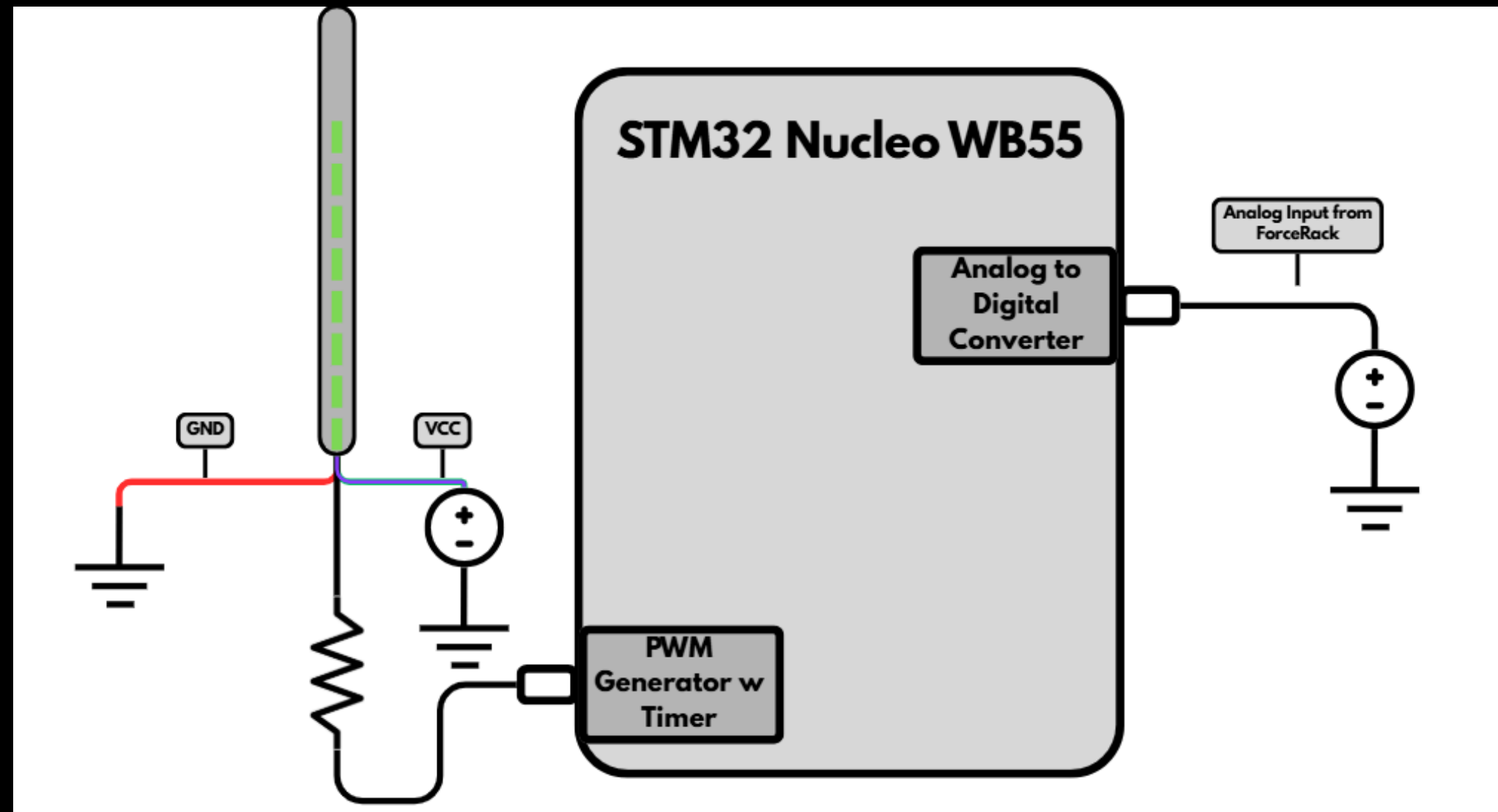


BACKEND



TRUEFORCE ELECTRONICS

- LED lights are run on the STM32 Nucleo Microcontroller



CONCLUSION

- We have a stable foundation for both deliverables, core loop fully functional
- Next: connect backend and load data (e.g.. leaderboards) instead of mock data
- Enhance animations, transitions, and sound for a polished user experience
- Key lessons: early testing, consistent architecture, and client communication