True Force Technologies Arcade Weight Rack

Design Document

Team Number: sdmay26-40

Client: True Force Technologies

Advisor: Matt Post

Team Members/Roles:

Jacob Garcia - Frontend Development
Dylan Longlett - Frontend Development
Andy Drafahl - Team Manager, Frontend Development
Fadi Masannat - Frontend Development
Sofi Gutierrez - Backend Development
Parnika Dasgupta - Advisor/Client Liaison, Backend Development
Colin Yuska - Electronics

Team Email: senior_design@iastate.edu

Team Website: https://sdmay26-40.sd.ece.iastate.edu/

Revised: 12/7/2025, Version 10.0

Executive Summary

The True Force Technologies Arcade Weight Rack project focuses on transforming an existing isometric strength measurement device into an interactive experience suitable for public events and customized corporate use. The current Force Rack hardware can measure pull force, but it is limited in its ability to engage users outside of simple data display. Through multiple client meetings and iterative clarification, our team identified a broader need for a system that feels fun, intuitive, and visually appealing, while still maintaining accuracy and reliability. This led to two primary deliverables: an arcade version designed for high-energy environments such as fairs and concerts, and a customizable version intended for company events that require branded content, unique units, or custom messaging. Both versions share the same foundational application and hardware.

Our key requirements were shaped directly by client input. These include real-time force visualization, user interactions that are simple and welcoming, optional leaderboards that can be toggled on or off to avoid intimidation, the ability to upload logos and custom text, support for multiple unit types such as pounds, newtons, or themed units, and the need for a high quality physical printout to justify event pricing. We also learned from the client that animations must remain gender neutral and on brand, and that all data for the arcade deliverable should remain local rather than online. These user and client expectations have guided both design scope and development priorities.

Our design uses a React application as the core of the interface. It connects to the existing STM32 Nucleo WB55 microcontroller through Bluetooth to receive force data, and it incorporates a MariaDB backend for storing leaderboard entries and customization settings. LED strips run through the rack and are programmed to give immediate feedback to the user in a way that visually matches both the arcade and corporate themes. Early in the project, the Bluetooth integration represented the highest risk because it depended on another team working in parallel. To stay productive, we focused on UI development, mock data pipelines, animation experiments, and the customization workflow until stable Bluetooth data became available. This allowed our team to make progress without being blocked by upstream dependencies.

At this stage, we have completed early versions of the Arcade and Custom deliverable screens, set up the development environment, currently creating database structures, built working Bluetooth discovery and data reading, and clarified artistic direction with the client. We shifted away from concepts such as the Sword in the Stone theme and moved toward superhero-style avatars with a clean and simple background. We also confirmed that the final arcade version will run on a Raspberry Pi while the custom version will operate from the existing React application. These decisions allow the system to scale and adapt to both high throughput public settings and branded corporate events.

Overall, our progress aligns well with the requirements and user needs identified throughout the semester. The next steps involve refining animations, integrating real force data consistently into the UI, implementing the printout and email workflow, improving database logic for storing session data, and completing full testing. With the core architecture, design direction, and communication patterns already in place, the project is well positioned to continue toward a polished and reliable final product.

Learning Summary

Development Standards and Practices Used

Our team relied on practical and commonly used software development practices throughout the project. We adopted an Agile workflow supported by weekly sprints, a Kanban board, and regular client and team meetings. Version control was handled through Git and GitLab, and we established branching and merge guidelines early in the semester after running into several repository issues. React development followed component based design practices, which helped us divide responsibilities and maintain consistent UI patterns. For the Bluetooth workflow, we coordinated closely with the partner team responsible for the hardware side to ensure compatibility and avoid duplicate work. We also integrated ongoing code refactoring, design reviews, and mandatory React learning sessions to maintain quality and consistency across the project.

As for standards, the most relevant industry practice is the use of Bluetooth Low Energy communication from the STM32 Nucleo WB55 board. On the software side, our work aligns with common web standards, React best practices, and database management practices using SQL based systems. For UI and UX decisions, we followed accessibility minded choices such as clear typography, readable color contrasts, and simplified flows requested directly by the client. These practices helped ensure that our design remained maintainable, intuitive, and aligned with real world expectations.

Summary of Requirements

Arcade Mode Requirements

- Real time force visualization
- LED animations that match intensity
- Gender neutral animated avatars
- Horizontal screen layout
- Local storage for leaderboard data
- Persistent arcade scores
- Physical printout of results

Custom Mode Requirements

- Ability to upload images and logos
- Ability to add custom text or units
- Simple form based customization workflow
- Optional leaderboard visibility
- Local data storage
- Printable summary page for events

Shared Requirements

- Accurate communication with the hardware through Bluetooth
- React based UI that is flexible and easy to navigate
- Clear feedback during and after each pull
- Consistency between animations, data visuals, and LED patterns

---

Applicable Courses from the Iowa State University Curriculum

- EE 201, Electronic Circuits
- CPR E 281, Digital Logic
- CPR E 288, Embedded Systems
- SE 309, Software Development Practices
- SE 319, Construction of User Interfaces
- SE 417, Software Testing

These courses together gave us a foundation in full stack development, state management, and system level thinking, all of which were necessary for building a dual mode interactive system.

New Skills and Knowledge Acquired Outside the Curriculum

Working on this project required our team to learn several skills that are not formally taught in the Iowa State curriculum. One major area was modern React development. Although some team members had partial exposure, the project required consistent use of hooks, component composition, styling libraries, and responsive layout techniques. We also learned practical Git collaboration strategies, including how to handle merges, prevent repository breakage, and structure branches in a way that supports multiple parallel deliverables.

Our team also picked up new skills in UI animation workflows, concept art refinement, and the process of translating client feedback into interactive screens. We learned how to build and refine a Bluetooth pipeline in a real application context, including testing data flow from another team's hardware. Another significant area of growth involved communicating with a real client, reading ambiguous requirements, and asking the right questions to clarify scope. Finally, we gained experience coordinating two separate software deliverables that must eventually connect, which required a deeper understanding of scalability, maintainability, and long term product vision.

Table of Contents

1. Introduction

1.1. Problem Statement

Our client, True Force Technologies, has already developed a Force Rack that measures the force exerted by a user pulling up on a bar. However, as it stands, the device only outputs data in a chart format. While this has some practical applications (fitness, training, etc.), our client looks to make something more engaging for a general audience in order to market the Force Rack, similar to the many strength arcade games popular at public events (boxing machines, grip strength testers, etc). Our challenge, then: "How can we take a static deadlift bar and make it exciting enough for a wide audience to try out, regardless of their strength level?"

This question led to the development of a King Arthur-inspired concept, where users attempt to "pull the sword from the stone" by exerting force on the bar. Instead of simply displaying numbers, the system would use animations, lights, and rankings to simulate the legendary challenge, making it feel like a game rather than a test. The rankings could range from "Squire" to "Knight" to "Arthur," giving users a sense of achievement and fun regardless of their actual strength. (This concept was later abandoned in favor of a more simple deadlift model, but this still reflects our brainstorming process.)

However, our client posed an additional challenge: "What about corporate events?" Our client wanted to be able to rent out the Force Rack, and while the King Arthur theme might be passable for a corporate setting, companies would likely want to insert their own branding and measurements of strength. So, we brainstormed a variation to our arcade idea: for corporate events, we'd offer a stripped-down version of the software with a simple form for adding visual assets. By allowing organizations to customize the experience, adding logos, changing units to pizzas, cases of soda, etc., or adjusting difficulty, we'd be able to target a corporate audience without drastically altering our existing software.

1.2. Intended Users

From the outset, our client had two specific user groups in mind (which is why we are focusing on two, rather than the typical three). While the client's original intention with the True Force Rack was to use it for physical training purposes, they gave us a different area of focus. The two groups are as follows: passersby at public events and companies looking to host events of their own. Our team suggested alternative audiences, but the client specifically wanted us to invest in these two.

Let's call a person within our group of passersby, Greg. Greg is walking along the street at an event with friends and notices that something to the left has caught their attention. They're debating amongst themselves if they should go try out the Force Rack, and Greg is considering trying it himself. However, he needs to feel comfortable and confident enough to approach the Force Rack; if the game feels too serious or the competition too stiff, he's likely to feel intimidated and pass it up entirely. Greg also needs to feel like he's going to have fun; if the game seems bland or unrewarding, he might see it as a waste of time and move on with his friends. We aim to address both of these needs with features like leaderboards that can be hidden with a simple button press to eliminate the pressure of competition, or animations and lighthearted rankings to make the game more motivating and fun to play, to name a few.

Let's call a person within our company group, Chelsea. Chelsea is the manager of a pizzeria and wants to spice up a company party. She sees the Force Rack at an event and thinks it might be just what she needs. However, she isn't sold on the arcade theme. Instead, Chelsea would like to personalize it for the party to make it special for the employees at the pizzeria. She wants a way to display the pizzeria's logo, or potentially change the animation so that boxes of pizza are being lifted, rather than a sword from a stone. However, Chelsea assumes that it would be complicated to set these customizations up herself, so she would need a quick and foolproof way to pull it off. She's on the fence about the Force Rack, but if those sorts of features were available, she would rent it for the party without hesitation. We aim to address these needs with a simple form that would allow users like Chelsea to customize the Force Rack experience without needing any background in programming.

2. Requirements, Constraints, And Standards

2.1. Requirements & Constraints

List all requirements for your project. Separate your requirements by type, which may include functional requirements (specification), resource requirements, physical requirements, aesthetic requirements, user experiential requirements, economic/market requirements, environmental requirements, UI requirements, and any others relevant to your project. When a requirement is also a quantitative constraint, either separate it into a list of constraints, or annotate at the end of the requirement as "**(constraint)**." Ensure your individual requirements are realistic, specific, and reflective or in support of user needs, and ensure your collective requirements are comprehensive.

Functional Requirements

These requirements specify what the system must do.

- The system shall accurately read and digitize isometric force data from the Force Rack's load cells.
- The system shall process raw sensor data to provide a stable, responsive input signal for the game application.
- The system firmware shall transmit processed force data to the main game application with minimal latency. **(constraint)**
- The game shall feature clear visual progression as someone accomplishes the goal.
- The system shall include an "attract mode" or idle screen that runs when no one is playing to draw user interest.
- The system shall track and display a user's score during gameplay.
- The system shall feature a leaderboard that displays the top scores.

User Interface (UI) & User Experience (UX) Requirements

These requirements define how the user interacts with the system.

- The UI shall provide clear, on-screen instructions on how to interact with the system.
- The system shall give the user immediate and continuous visual feedback that directly corresponds to the amount of force they are applying.
- The system shall provide auditory feedback (sound effects, music) to enhance user engagement.
- The game must be initiated through a simple, single action.
- All on-screen text must be large and legible from a distance of at least 5 feet. **(constraint)**
- The user experience is intended for a "walk-up-and-play" arcade environment, requiring no prior knowledge of the system.

Hardware & Physical Requirements

These requirements relate to the physical components of the project.

- The system will use a dedicated display screen.
- All electronic components (PCB, power supply, processing unit) shall be contained within a durable, professionally presented enclosure.
- The physical design must be robust enough to withstand repeated, forceful use in a public setting.
- The system must integrate seamlessly with the client-provided Force Rack barbell and frame.

Resource & Economic Requirements

These requirements concern the resources and budget for the project.

- The project must be completed within a two-semester timeframe. **(constraint)**
- The project must utilize the existing codebase and hardware provided by Strength Technologies LLC as a starting point. **(constraint)**
- The total cost for all new components (PCB, display, processor, enclosure, etc.) shall not exceed the allocated senior design budget. **(constraint)**
- The final prototype must serve as a viable proof of concept for the client.

Aesthetic Requirements

These requirements define the visual and auditory style of the project.

- The game's visual design shall have a consistent and engaging arcade-style theme.
- The sound design, including music and sound effects, shall be cohesive with the arcade theme.
- The physical enclosure for the hardware shall have a clean design that conceals wiring and appears like a finished product.

2.2 Engineering Standards

What Engineering standards are likely to apply to your project? Some standards might be built into your requirements (e.g., many projects use 802.11 ac wifi standard) and many others might fall out of design.

Q1)

Engineering standards are essential because they ensure consistency, safety, reliability, and interoperability across systems and technologies. Standards provide a common framework for design, implementation, and testing, which helps teams avoid errors, reduce development costs, and improve user experience. For example, applying human centered design standards ensures that the arcade game interface is intuitive and easy for users of varying experience levels. Similarly,

software and communication standards reduce technical risks, such as connection failures or data inconsistency.

Q2)

ISO 9241-210 (Human-Centered Design for Interactive Systems)

- **Purpose:** Provides guidelines to make interactive systems more usable and user-friendly.
- **Intended Outcome:** Ensures the interface is intuitive, safe, and enjoyable, reducing user errors and improving engagement.
- **Relevance:** For the arcade game, this standard guides the design of menus, touch inputs, and feedback systems so that users can navigate the game naturally and without frustration.

IEEE 802.15.1 (Bluetooth Standard)

- **Purpose:** Specifies protocols for short-range wireless communication between devices.
- **Intended Outcome:** Ensures reliable, standardized data transfer over Bluetooth, including device pairing, signal strength, and interference management.
- **Relevance:** This standard is crucial for connecting the weight rack sensors to the mobile app without communication errors or data loss.

IEEE 828 (Software Configuration Management Standard)

- **Purpose:** Provides guidelines for managing software changes, version control, and tracking modifications throughout the development lifecycle.
- **Intended Outcome:** Improves software reliability, maintains traceability, and ensures systematic handling of updates or bug fixes.
- **Relevance:** Our game software will follow this standard to prevent version conflicts, track feature additions, and ensure all team members work on a consistent codebase.

Q3)

Relevance to the Project

All three standards directly support the success of the project:

- **ISO 9241-210** improves user interaction and reduces learning curves for players.
- **IEEE 802.15.1** guarantees that the fitness hardware communicates reliably with the mobile app, which is critical for real-time tracking of exercises.
- **IEEE 828** ensures that software development remains organized, reproducible, and maintainable, reducing the risk of bugs or system failures.

Q4)

Team Review of Standards

After reviewing individual selections, other team members focused on additional standards such as:

- **IEEE 802.11ac Wi-Fi** for possible wireless communication in areas where Bluetooth may be insufficient.

- **ISO/IEC 25010 (Software Quality Requirements and Evaluation)** for evaluating software usability, performance, and reliability.
  These selections complement the original standards by providing additional coverage of software quality and wireless communication protocols.

Q5)

Modifications to Project Design

To incorporate these standards, the following modifications will be made:

1. **User Interface:** Redesign menus, buttons, and feedback cues to align with human-centered design principles.

2. **Wireless Communication:** Implement Bluetooth protocols according to IEEE 802.15.1 for all sensor-to-app data transmissions.

3. **Software Management:** Use version control tools and follow IEEE 828 guidelines to track changes, manage releases, and document updates.

4. **Testing and Evaluation:** Add usability testing sessions and wireless connectivity tests to validate compliance with ISO 9241-210 and IEEE 802.15.1.

3 Project Plan

3.1 Project Management/Tracking Procedures

We are using an agile methodology to keep track of our project goals, tasks, and deadlines. Within the agile methodology, we chose a Kanban-style agile development process for the first semester. Transitioning into a SCRUM agile development process as we move into the second semester. We chose this because the initial work will be more To-do and task-oriented, as it is more theoretical and prep-based. As we move into the second semester, we will transition to the SCRUM agile methodology because it will match our software-oriented design process as we continually make more progress on our application through sprint iterations.

3.2 Task Decomposition

When we create a Task, normally the first thing we do is break it down into sub-tasks and add deadlines, assignees, and expected deliverables. The big/general task is put as a swimlane in the sprint, then inside the swimlane, we add our tasks that make up the big task, and we can break those down into even more subtasks if necessary.

3.3 Project Proposed Milestones, Metrics, and Evaluation Criteria

Mile Stone 1: Fixing Prior Issues

BlueTooth will be fully fixed and functional. React app will be able to connect to TrueForce Rack in order to test physical functionality.

Mile Stone 2: Build Arcade and Custom Deliverables

Software frameworks for both the Arcade and Custom deliverables will be designed. They will represent the flow that participants will experience in a proof of concept.

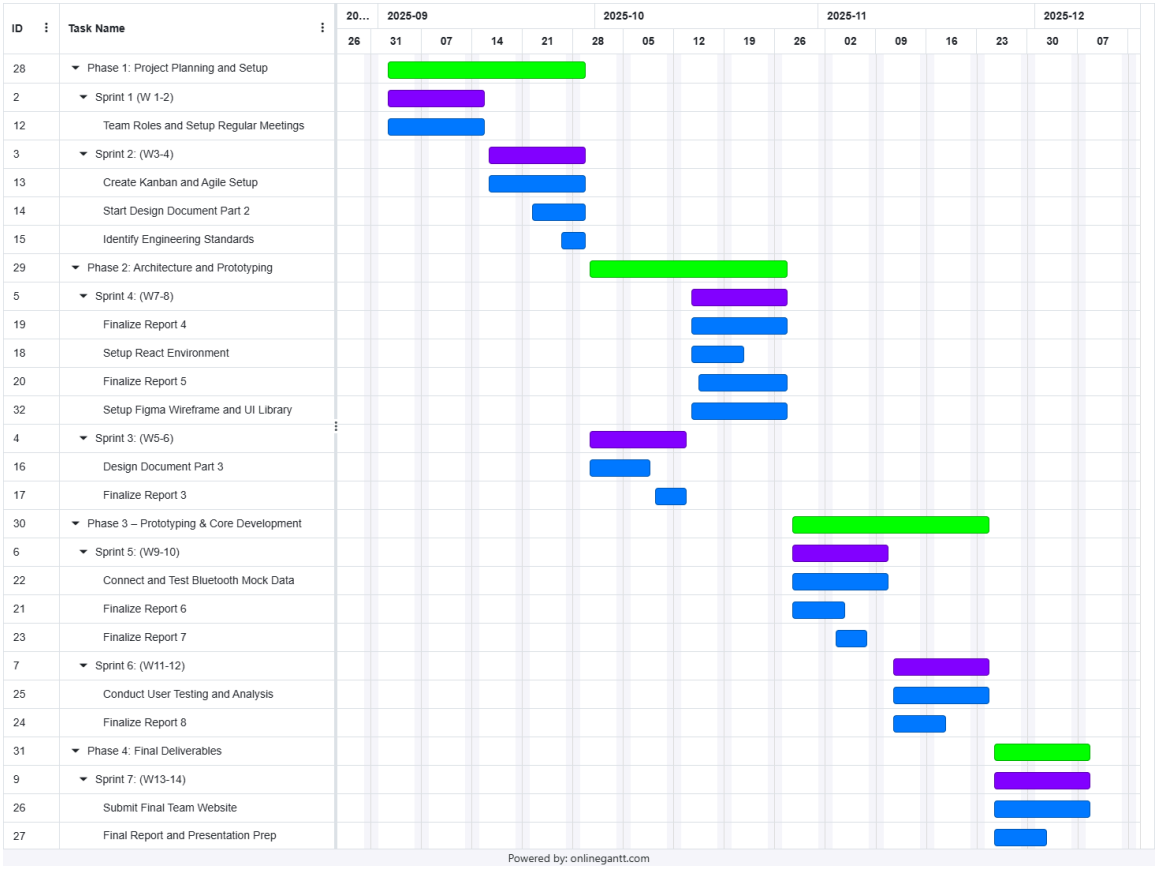
Mile Stone 3: Full Electronic Design

Any physical portions of the design will be combined into a single unit, to be implemented into the final design.

Mile Stone 4: Completed Design for Customer Testing

Physical and Software portions of the project will be finalized and combined into a cohesive final product.

3.4 Project Timeline/Schedule



** **Annotations - Phases are in Green, Sprints in purple and tasks in blue***

Our schedule is structured around seven sprints following a two-week Agile cycle. Each phase builds toward specific technical milestones and course deliverables. In Phase 1, the team focused on project planning, defining requirements, and establishing Agile tracking procedures (Team Contract, Reports 1–2). Phase 2 centered on system architecture and prototyping, including UI wireframes, environment setup, and high-level design documentation (Reports 3–5). Phase 3 covers early implementation and integration work, such as connecting the Bluetooth mock data and developing the first functional prototype (Reports 6–8). Finally, Phase 4 is devoted to testing, polishing, and final deliverables, culminating in the Final Report, Team Website submission, and faculty presentation during Week 15.

This timeline ensures that progress is distributed evenly across the semester while leaving adequate time for integration, client feedback, and final testing.

3.5 Risks and Risk Management/Mitigation

Our project follows an Agile development process using Github for version control and agile for the primary frontend framework. Our team will eventually split into two subgroups; one will focus on the arcade version, and one will focus on the customizable company version. We will be having weekly sprint reviews where each sprint review will include a risk review checkpoint to update work progress and identify emerging issues.

Task	Risk	Probability	Severity	Mitigation
React Frontend development	Limited experience may slow down the progress	0.6	High	Our team decided to do a basic tic tac tutorial together to get familiar with React
Data storage and Backend setup	The existing database might not be enough and might be incompatible with our work	0.5	Medium	The Backend people can rebuild the database (using Mariadb, etc)
Printable report generation	Generating printable documents directly from the app might be difficult	0.6	High	If this fails, we could potentially export data to csv or pdf for manual printing
Cross team coordination	Misalignment with bluetooth team might cause delays	0.6	High	Weekly communication with the other team. Frequent update exchanges between the two liaisons of the two groups
Hardware LED and Display Integration	Hardware integration might be difficult since the group is Software Engineering heavy	0.5	Medium	We could all sit together and research more and help our one EE member

3.6 Personnel Effort Requirements

Task	Description	Team Member	Estimated Hours	Notes
Frontend setup	Setting up the React App locally, integrating the github repo, configuring testing environments	Jacob and Andy	40	Installing, testing,
Backend design and setup	Establishing backend logic for data transfer, and storage. Creating end points for leaderboard and user inputs	Sofi and Parnika	30	Schema setup and testing with mock setup
Bluetooth data integration	Connecting React frontend to Bluetooth	Colin, Fadi, Dylan	30	Dependent on the other team. Coordination with other team is important
Printable Report Feature	Generating Printable summary reports on paper	Jacob, Dylan, Colin	15	Needs heavy brainstorming
Leaderboard and analytics	Implementing Leaderboard logic and basic analytics	Sofi, Parnika	15	Main focus on usability
Agile Management	Maintaining Kanban board, sprint planning, team communication	Sofi, Fadi	8	Weekly updates
Documentation and Client Interaction	Constant communication with Advisor, client, and team	Parnika, Andy	8	Weekly summaries and updates to be sent to both

				client & advisor

3.7 Other Resource Requirements

Our project is mostly software-based. There aren't a lot of resources we would need to complete the project. The only critical component is the Bluetooth module, which we need to send data from the app to the Arcade Rack. All the other resources can be clearly accessed on campus or from the client's funding budget.

4 Design

4.1 Design Context

4.1.1 Broader Context

The True Force Technologies Arcade Weight Rack is designed to serve two primary communities: general event attendees (e.g., fairgoers, festival participants) and companies seeking unique entertainment options for private events. These groups interact with the system directly through gameplay and branding, and indirectly through the cultural and economic ripple effects of public installations and corporate rentals.

Public Health, Safety, and Welfare

- Encourages physical activity in a fun, low-pressure setting, promoting wellness among casual users.
- Prioritizes safety through robust hardware and intuitive UI/UX that minimizes misuse or injury risk.
- May increase interest in strength training and fitness through gamified interaction.
- Supports job opportunities in event staffing, hardware maintenance, and software customization for private clients.

Global, Cultural, and Social

- Reflects global trends in gamification and interactive entertainment.
- The deadlift theme is universally appealing, while the customizable deliverable allows adaptation to diverse branding needs.
- Avoids exclusionary practices by offering adjustable difficulty and optional leaderboard visibility, making it accessible to users of varying backgrounds and abilities.
- Respects professional ethics by ensuring transparency in data handling and avoiding misleading strength metrics.

Environmental

- Uses electronic components and display hardware, introducing considerations around energy consumption and e-waste.
- Designed for durability and reusability, minimizing the need for frequent replacements.
- Prioritizes sustainable materials where possible and avoids single-use components.
- Has a low environmental footprint during events due to its self-contained design.

Economic

- Must remain affordable for both public event organizers and companies renting it for private functions.
- Offers recurring revenue opportunities through rentals and branded experiences.
- Leverages existing hardware and codebases to reduce development costs and maximize ROI for the client.
- Modular design allows scalability across different event sizes and budgets, enhancing market viability.

4.1.2 Prior Work/Solutions

Several products and research efforts have explored gamified strength measurement and interactive fitness installations. Notable examples include:

- Grip strength testers and boxing arcade machines, which provide basic force feedback and rankings but lack customization or modern UI/UX.
- Smart gym equipment like Tonal and Peloton which integrate sensors and feedback but are designed for personal fitness rather than public engagement.
- Prior senior design projects at Iowa State have explored sensor integration and arcade-style feedback, but none, to our knowledge, have combined hardware robustness with customizable branding.

Relevant literature includes:

[1] J. Kim, H. Park, and S. Lee, "Design and Implementation of Interactive Fitness Equipment Using Force Sensors," IEEE Access, vol. 8, pp. 123456–123465, 2020.

[2] M. Smith and A. Johnson, "Gamification in Public Health: A Review of Arcade-Based Interventions," in Proc. IEEE Int. Conf. on Healthcare Informatics, 2021, pp. 45–52.

[3] T. Nguyen et al., "Customizable UI Frameworks for Embedded Systems," IEEE Trans. on Human-Machine Systems, vol. 51, no. 3, pp. 210–220, 2022.

The following is a comparison of existing products to our solution:

Feature	**Existing Products**	**Our Solution**
Custom branding support	Limited	Full UI customization
Real-time force feedback	Basic	Responsive, animated
Arcade-style engagement	Yes	Enhanced with themes and ranking
Private event adaptability	No	Yes
Hardware robustness	Varies	Designed for public use

Pros:

- Two deliverables for different use cases (arcade and private event)
- Modular software architecture
- Engaging UI/UX with adjustable difficulty

Cons:

- Requires Bluetooth integration and real-time data processing
- Hardware constraints may limit scalability

4.1.3 Technical Complexity

Our project meets the criteria for high technical complexity through its integration of multiple subsystems, each requiring distinct engineering principles:

Subsystems and Principles

- Sensor Data Acquisition: Uses load cells and analog-to-digital conversion, requiring knowledge of signal processing and calibration.
- Bluetooth Communication: Implements IEEE 802.15.1 protocols for low-latency wireless data transfer.
- Frontend Game Application: Built in Python with real-time animation, requiring asynchronous data handling and responsive UI design.
- Frontend Custom Application: Built in React with real-time animation, requiring asynchronous data handling and responsive UI design.
- Backend Infrastructure: Manages user scores, leaderboard data, and customization assets using MariaDB.
- Hardware Enclosure: Designed for durability and aesthetics, involving mechanical design and user safety considerations.

Challenging Requirements

- Real-time force visualization with minimal latency
- Customizable UI for non-technical users
- Robust integration with existing hardware
- Compliance with multiple engineering standards (ISO 9241-210, IEEE 828, IEEE 802.15.1)

Together, these elements demonstrate a project scope that exceeds typical academic prototypes and aligns with industry-grade expectations for interactive systems.

4.2 Design Exploration

4.2.1 Design Decisions

1. Splitting into Arcade and Custom deliverables.

As the team spoke with the client about locations that he had been using the Force Rack and locations where he hoped it would soon gain traction, it became clear that the team couldn't just make one version of the app that was universally well-suited for every situation. That's why they split the app into two deliverables: An Arcade deliverable for events, fairs, etc., where random passersby would try out the Force Rack quickly, and a Custom deliverable for company parties, training, etc., where companies would upload custom assets and change color themes to match their business. This has allowed the team to specialize the app for multiple cases without having to create two separate apps or abandon a core part of the Force Rack's functionality.

2. Horizontal instead of vertical displays.

At first, the team considered the option of a vertical screen next to the Force Rack. They thought this would be better suited for the visuals of the Arcade deliverable (power bars filling up vertically, weights going upward, etc.). However, once the client expressed how much they've enjoyed having a

horizontal display so far, and once the team considered the difficulty that a vertical display might bring for PC users, they scrapped the idea and settled on a horizontal display. This choice will likely make the development process simpler, as the team won't have to worry about adaptive displays.

3. Individually lit LEDs.

Before the team started working on the Force Rack, the client had added some generic LED strips under the rack to give it some ambient lighting. However, the client didn't have any specific or responsive control over these LEDs. The team decided that they would add their own LED strips, with the ability to change color, dim, and power on/off individual LEDs. These would take the data transmitted from the app and use it to display a vertically growing meter of power being exerted by the user. This design choice greatly improved the aesthetic appeal of the arcade rack.

4.2.2 Ideation

One design choice that took a great deal of deliberation was the theme for the app. Because the team knew they were making an arcade game, they wanted it to be highly interactive, easily recognizable, and most of all, fun. However, nearly every team member had a different idea of what that should look like. Over the course of many brainstorm sessions and spirited discussions, they considered these themes:

1. Flappy Bird.

Lifting up on the bar would raise or lower the bird, allowing users to navigate a maze of pipes. This was easily recognizable, but the team scrapped it because it discouraged the user from applying constant force.

2. The Incredible Hulk.

This was floated pretty briefly, but it matched the lifting theme. It was recognizable, but obvious licensing issues meant that it wasn't a viable option.

3. King Arthur/The Sword in the Stone.

This was the definite front-runner for a long time. It was recognizable, and the team could avoid copyright issues because King Arthur is in the public domain. However, after much debate, it was ditched; the team deemed the act of pulling up on a horizontal bar too physically different from pulling up on a sword and decided it would reduce user immersion.

4. Standard Deadlift.

At first, the team avoided it since it seemed like a bland theme, but after all the different theme ideas fell through, they realized it was pretty logical–no motion is going to match a deadlift better than a deadlift. The team was satisfied with the idea that they could add all sorts of visual flair to that theme to make it more exciting.

5. Pizzas, sodas, pies, oh my!

In addition to the basic deadlift theme, the team needed a visual that would allow for customization on the part of companies renting the Force Rack. Thus, they came up with a simple system to build on the already existing deadlift visual–companies would upload an image of an item they wanted to lift (pizzas, boxes, you name it), and the deadlift bar would be swapped out with the uploaded image, creating another fun theme based around lifting specific items.

After considering all 5 options, the team settled on options 4 and 5.

4.2.3 Decision-Making and Trade-Off

The team weighed whether each theme was:

1. Recognizable (first-glance identification from users)

2. Practical (users would use the Force Rack properly with this theme)

3. Royalty-free (can we use this theme without copyright issues)

4. Immersive (does the theme match the physical action the user is doing)

5. Fun (would users actually enjoy using the rack with this theme)

	Recognizable	Practical	Royalty-free	Immersive	Fun
Flappy Bird	✅	❌	❌	❌	✅
The Incredible Hulk	✅	✅	❌	❌	✅
King Arthur	✅	❌	✅	❌	✅
Standard Deadlift	✅	✅	✅	✅	✅
Custom Item	❌	✅	✅	✅	✅

After considering these options, the standard deadlift and custom item were the clear winners.

4.3 Proposed Design

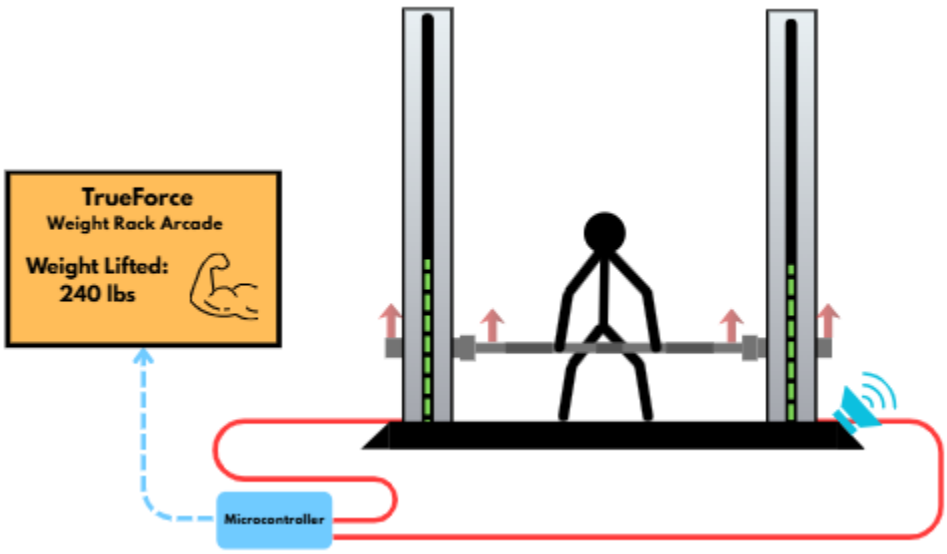
4.3.1 Overview

The True Force Technologies Arcade Weight Rack is an interactive weight rack that turns a deadlift exercise into an arcade-style game. By measuring the force applied by lifting up on the bar, the arcade rack gives audio and visual feedback through game design elements in order to facilitate a fun experience. The True Force Rack is made up of three key major components: the weight rack, the microcontroller, and the display screen.

The Force Rack consists of two metal pillars attached to the opposite ends of a base. The metal pillars are notched in a way to hold a metal deadlift bar in place that spans between the towers. On the towers are strips of LEDs that give visual feedback on how much weight is being lifted. Also connected to the metal poles are load sensors that detect the force being generated by the participant. This signal from the load sensors is then sent to the microcontroller.

The Microcontroller interprets the analog signal from the load sensors connected to the rack and sends the data via Bluetooth to the display. The microcontroller also sends signals back to the LEDs connected to the Weight Rack, as well as the speakers in the base.

The Display is a TV screen that mirrors the laptop or iPad that is running the game application. The application provides the weight rack with the software functionality needed to run as an arcade game or as a custom game.



Visual Representation of the True Force Technologies Arcade Rack

4.3.2 Detailed Design and Visual(s)

The True Force Arcade Rack is an arcade game with the goal of testing someone's strength via a deadlift exercise and displaying their results. The Arcade Rack consists of three separate main elements, each with their own subsystems. The three main elements, as well as their components, are summarized below.

Physical Weight Rack:

- Large platform to act as the base.
- 2 Metal Towers that connect the deadlift bar and pull sensors
- Deadlift bar that the participate is pulling on
- 2 Pull Sensors that measure the amount of force generated upwards by the participant
- Individually Addressable LED Strips that progressively light up with more force applied
- Speaker located within the base that gives audio feedback

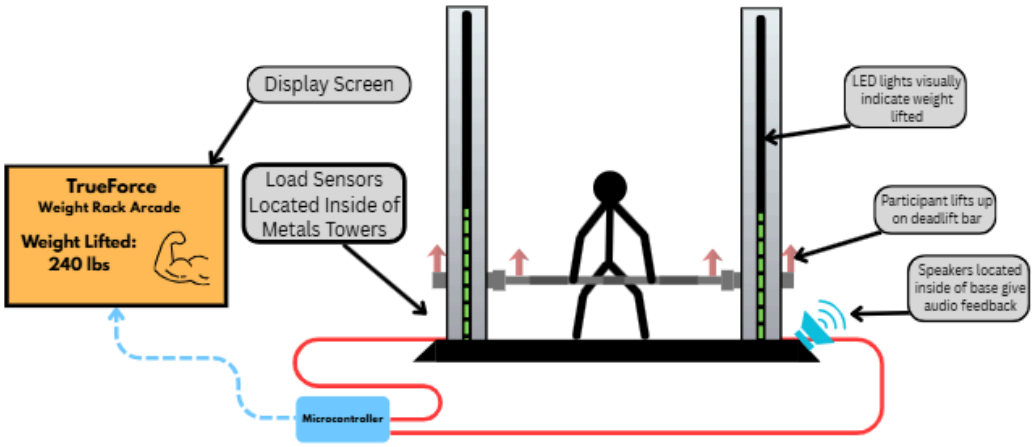
STM32 Nucleo-WB55 Microcontroller:

- GPIO Pins that receive input/output signals
- Bluetooth Transmitter/Receiver that communicates with the display

- Analog-to-Digital Converter that takes analog pull sensor data and converts it into digital data to be sent to the display

Display/Arcade Application:

- iPad or Laptop that runs the software the connects to the microcontroller
- TV that is screen-mirrored to the iPad or Laptop
- The Play Settings Screen allows users to choose the specific settings
- Leaderboard Screen, optional screen that displays top placements
- Printer that automatically prints off results for participants



Visual Representation of the True Force Technologies Arcade Rack

The Physical Weight Rack's two metal towers are connected to the pull sensors embedded in the base board through metal pins. The Metal tower's edges are notched like a traditional weight rack in order to hold the deadlift bar in place while in use, but also to allow disassembly. The pull sensors measure the force being exerted upwards and output an analog voltage signal to the microcontroller. The LED light strips on the front of the towers are individually addressable. This specification was chosen due to the ability to decide which lights are to be turned on and off. In the design, this feature will receive input from the microcontroller and visually indicate how much weight is being lifted.

The STM32 Nucleo-WB55 Microcontroller's main function is to act as the brain of the arcade rack. The microcontroller receives inputs and outputs through the board's General Purpose Input Output or GPIO pins. The firmware uploaded to the Nucleo determines what the specific GPIO pins are mapped to, as well as where the interpreted data goes. One specific subsystem in the Nucleo is the ADC. After receiving the analog voltage signal from the weight rack's pull sensors, the data that is read off the pin is transmitted to the analog-to-digital convertor on the nucleon. This sub-system converts the raw pull sensor data into a more readable digital signal to be sent to the web app through Bluetooth. Some of the GPIO pins are set as outputs in order to send data to the LEDs and speakers.

The screen connected to the Force Rack will mirror a display from an app that can run on a PC or tablet. The Arcade and Custom apps are separated into several views. Users are greeted by a title screen, and then asked to choose from several settings. Originally, the app branched into two paths so that the Arcade and Custom deliverables were different “modes” within one app, which is reflected in the Figma design below. Now, these modes are separated into two apps, but they are still functionally similar, with the Arcade app favoring game-style elements like sound effects and colorful visuals while the Custom app relies more on detailed output. There are views for initial setup, gender selection, a play view that displays when the user is on the Force Rack, a settings page, a results page, a toggleable leaderboard page, and a printout page to show users their stats.

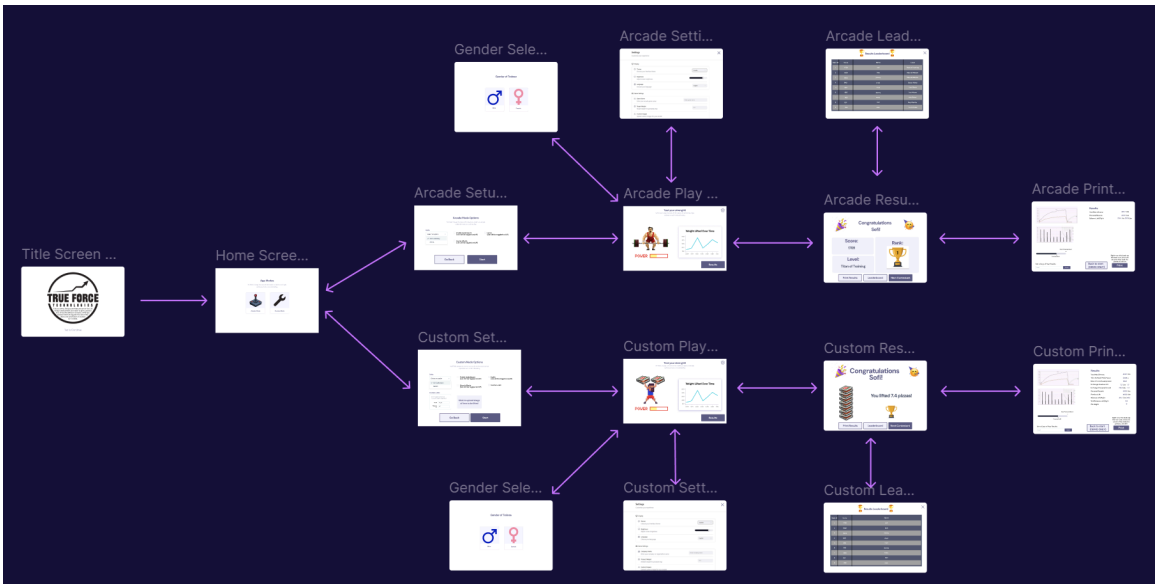


Diagram of Views Within the App

4.3.3 Functionality

In a real-world context, the True Force Arcade Rack is designed to operate in two primary versions: the Arcade deliverable and Custom deliverable. These modes share certain functionalities, but the Arcade mode is written in Python while the Custom mode is written in React.

Arcade Deliverable

In this deliverable, the True Force Rack is in public social environments such as fairs, entertainment districts, or university events. Its goal is to attract passersby through visual and audio feedback.

When a participant approaches the rack, a simple prompt is displayed on the screen of the Rack that invites them to enter their initials and gender to personalize their attempt. Once confirmed, the user grabs the deadlift bar with both hands and pulls upward as hard as they can. The system continuously measures the applied force via the pull sensors and transmits these readings via Bluetooth to the display. In real time, the screen displays their current pull force as well as an animation that changes depending on the amount lifted. In addition, the LED lights on the front of the machine are progressively lit up as more weight is pulled.

When the pull ends, the system records the maximum amount of force the participant pulled and compares that to the stored leaderboard scores. This public display encourages competition and engagement among other passersby.

Custom Deliverable

The custom deliverable is designed for private events or corporate gatherings. While the gameplay remains similar, arcade elements are reduced in favor of more specific settings to allow for increased configurability.

Event hosts can upload their organization's specific logos, colors, and other features through a customization screen. One example could be a pizza company changing the weight measured from pounds to pizzas, resulting in the final outcome showing how many pizzas a participant could lift.

After the configuration, participants at the events follow the same sequence. Enter information, test strength, and view results. But the focus shifts from a competitive atmosphere to more personal engagement and brand connection.

4.3.4 Areas of Concern and Development

Our current design for the True Force Arcade Rack satisfies the majority of the functional requirements given by the client. The hardware successfully measures applied force and uses that data to inform the display and hardware functions on what to do. The Python frontend delivers real-time visual and audio feedback, fulfilling the entertainment and engagement goals for the Arcade deliverable, while the React frontend delivers the fine-tuned customizability needed for the Custom deliverable.

From a user perspective, the system meets core expectations for accessibility. The experience requires no prior knowledge, allowing for easy, immediate participation. For corporate clients, the customizable interface allows for easy implementation of their own logos and branding.

However, one of our primary concerns for delivering on all of the requirements is the client's lack of distinction between expectations for this group and the partner group more focused on the web app. During meetings, expectations set for the week can come up out of nowhere. But when thought of in the context of being for the other group, they start to make more sense. So one of the primary concerns is discerning where the expectations lie between groups when they aren't explicitly stated. One way to remedy this would be to make sure communication is clear, with both the client as well as the other group working in close connection to this group. This would solve confusion going forward with which group needs to work on what.

4.4 Technology Considerations

Our system integrates several distinct technologies across software, communication, and interactive hardware components. Each was selected based on usability, cost, compatibility with existing infrastructure, and scalability for future deployments.

Technologies Used

- Bluetooth (IEEE 802.15.1): Used for wireless transmission of force data from the load cell sensors to the game application.
    - *Strengths*: Low power, widely supported, suitable for short-range communication.
    - *Weaknesses*: Susceptible to interference in crowded environments; limited bandwidth.
    - *Trade-offs*: Chosen over Wi-Fi due to simplicity and lower energy demands. Wi-Fi (IEEE 802.11ac) may be considered for future versions requiring higher throughput or longer range.
- React Frontend: Powers the arcade-style game interface and customization form.
    - *Strengths*: Modular, widely supported, ideal for dynamic UIs.
    - *Weaknesses*: Steep learning curve for team members unfamiliar with React.
    - *Trade-offs*: Chosen for its scalability and community support. Initial development slowed due to onboarding, mitigated by team tutorials.
- MariaDB Backend: Stores leaderboard data, user inputs, and customization assets.
    - *Strengths*: Lightweight, open-source, compatible with existing infrastructure.
    - *Weaknesses*: May require schema adjustments to support future analytics or reporting features.
    - *Trade-offs*: Chosen over heavier alternatives (e.g., PostgreSQL) for simplicity and ease of integration.
- Lights and Sound Integration: Electronics work centers on adding LED lighting and sound effects to enhance user engagement.
    - *Strengths*: Provides immediate feedback and visual excitement; aligns with arcade-style experience.
    - *Weaknesses*: Requires careful timing and synchronization with game logic; limited EE resources on team.
    - *Trade-offs*: Chosen instead of custom PCB development to focus efforts on user experience and minimize hardware complexity.

Design Alternatives Considered

- Bluetooth vs Wi-Fi: Bluetooth (IEEE 802.15.1) was selected for sensor communication due to its low power consumption and simplicity. Wi-Fi (IEEE 802.11ac) was briefly considered as a fallback in environments where Bluetooth might be insufficient, but was deprioritized due to higher energy demands and complexity.
- Cloud Storage vs Local Database: Cloud-based storage solutions were considered for scalability and remote access. However, the team opted for a local MariaDB instance to reduce cost, simplify deployment, and maintain control over data during development and testing.
- Off-the-Shelf Electronics vs Custom Hardware: Rather than designing a custom PCB, the team chose to focus on integrating modular components (LEDs, speakers) to enhance user experience. This decision reduced hardware complexity and aligned better with the team's software-heavy skill set.

4.5 Design Analysis

Our team has made consistent progress throughout the planning and early implementation phases. We began by establishing an Agile workflow, using Kanban-style tracking in the first semester and transitioning to SCRUM methodology for iterative development in the second. This structure has helped us manage tasks, adapt to evolving technical needs, and maintain momentum across sprints.

On the frontend, we successfully set up the React development environment and completed tutorials to build foundational knowledge. Wireframes and a UI component library were created in Figma to guide the visual design of the arcade-style interface and the customization form for private event use. On the backend, we initiated schema design using MariaDB, with mock data testing planned to support leaderboard functionality and user input storage.

Bluetooth integration is being developed by a separate team, but we are in active coordination with them to ensure compatibility and timely data exchange. For hardware, our focus has shifted toward integrating LED lighting and sound feedback to enhance user engagement. This decision replaced earlier considerations of custom PCB design and better aligns with our team's software-heavy skill set.

While the design remains feasible and aligned with client goals, several challenges have emerged:

- Bluetooth integration is a high-risk area due to its technical complexity and our reliance on another team's timeline.
- Printable report generation is still being scoped. The challenge lies in determining how to implement a feature that emails a summary directly to the user in a way that is both intuitive and technically feasible.
- Hardware integration requires careful timing and testing for LED and sound synchronization, and our team has limited electrical engineering resources.

Despite these challenges, we are confident in the overall feasibility of the project. The modular design allows for fallback options if certain features cannot be implemented in full. Our next steps include:

- Completing the backend schema and connecting it to frontend components.
- Continuing coordination with the Bluetooth team and preparing for mock data testing.
- Prototyping LED and sound feedback tied to gameplay events.
- Implementing leaderboard logic and testing score tracking.
- Finalizing the customization form and validating branding asset uploads.

Regular sprint reviews and risk checkpoints will continue to guide development and ensure that emerging issues are addressed promptly. We remain focused on delivering a polished, engaging, and client-ready system by the end of the development cycle.

5 Testing

Our testing strategy is centered on ensuring the reliability and usability of the TrueForce mobile application. The core of our project is the interaction between the user's phone and a custom hardware device via Bluetooth Low Energy (BLE). Therefore, our testing philosophy is to test early and often, with a strong focus on the components that handle BLE communication and user interface responsiveness.

A unique challenge for our project is the dependency on the physical hardware. To overcome this, our testing plan includes using mock data and simulated BLE devices. This allows us to test the application's logic and UI independently of the hardware, ensuring that we can develop and test components in parallel and catch bugs early in the development process.

5.1 Unit Testing

Units Being Tested: We focus on the smallest testable parts of our application. This includes individual React components like ColorPicker.tsx, utility functions (e.g., for converting color formats), and the business logic contained within our custom hooks like useBLE.ts. For example, we test that the ColorPicker component correctly updates its state when a user selects a new color.

How: We use a testing framework to write individual test cases for each unit. For a component, we might simulate a user interaction (like a button press) and then check that the component's state or rendered output changes as expected. For a function, we provide a specific input and assert that the function returns the correct output. We heavily rely on mocking to isolate the unit under test from its dependencies, such as the native BLE module.

Tools: Our primary tools for unit testing are Jest, a popular JavaScript testing framework, and React Native Testing Library, which provides utilities for testing React Native components in a way that closely resembles how a user interacts with them.

5.2 Interface Testing

Interfaces: The key interfaces in our design are the connection between the UI components and the application logic (e.g., CustomizeMode.tsx screen interacting with the BLEContext), and the interface between the application and the external BLE hardware. This latter interface is defined by a specific BLE service and characteristic UUIDs, and a data protocol for sending commands like color changes.

How: We test the interface between UI and logic by writing tests that render a component and mock the context or hook it depends on. For example, we can test that the ScanDevices.tsx screen correctly displays a list of devices when the useBLE hook provides a mock list of peripherals. To test the app-to-hardware interface, we simulate a BLE peripheral and verify that our application sends correctly formatted data to the correct characteristic when a user performs an action, like selecting a color.

Tools: We use Jest and React Native Testing Library to test the integration between our software components. For hardware interface testing, we use a mocked version of the react-native-ble-plx library to create a virtual BLE device that our app can interact with during tests.

5.3 Integration Testing

Critical Integration Paths: The most critical integration path is the end-to-end user flow for connecting to a device and controlling it. This path includes: 1) scanning for available devices, 2)

selecting a device from the list, 3) establishing a connection, and 4) sending a command (like changing a color) and receiving a confirmation. This path is critical because it encompasses the core functionality required by the project.

How: We test this by writing integration tests that simulate this entire workflow. The test will navigate through the different screens of the app, from the ModeSelectionScreen to ScanDevices and finally to CustomizeMode. We use mock navigation and a mock BLE device to simulate the entire process without needing physical hardware. The test asserts that the application state is updated correctly at each step (e.g., the app shows a "connected" status) and that the final command is sent correctly.

Tools: We use Jest and React Native Testing Library for our integration tests. We also leverage mock implementations of React Navigation to simulate screen transitions.

5.4 System Testing

System Testing Strategy: Our system testing is performed manually on physical hardware. The strategy is to execute a series of test cases based on user stories that reflect our project requirements. For example, a requirement might be "the user must be able to select a custom color for the device." The system test would involve an actual user taking a phone, installing the app, connecting to the physical hardware, navigating to the color selection screen, and verifying that the device's light changes to the selected color. We test on both iOS and Android devices to ensure cross-platform compatibility.

Sufficiency of Tests: While our automated unit, interface, and integration tests cover the application's logic and component interactions, they cannot fully replace system testing. The full suite of automated tests ensures the software is robust, but only manual system testing can verify the actual interaction with the physical hardware and uncover issues related to Bluetooth signal strength, timing, or platform-specific quirks.

Tools: The tools for system testing are simple: an iPhone and an Android phone with the application installed, the physical BLE hardware prototype, and the developer tools provided by Xcode and Android Studio for real-time logging and debugging.

5.3 Regression Testing

Our regression testing focuses on preventing new features from disrupting existing behavior. Each time we add or change functionality, we run our full suite of Jest tests. This includes unit tests for components and hooks, interface tests for BLE communication, and integration tests for full workflows. The most important features to protect are scanning for devices, connecting, sending color commands, and updating device state in the UI. These map directly to core project requirements, so our regression approach is driven by those requirements. Our tools include React Native Testing Library and mocked BLE modules. These allow us to confirm that previously working logic still behaves as expected before merging any changes

5.4 Acceptance Testing

To show that both functional and non functional requirements are met, we create acceptance test cases based on the final project requirements. Each test case describes a user-level action, the expected outcome, and the condition for passing. Examples include connecting to a device,

selecting colors, and verifying that the hardware responds correctly. We involve our client by giving them a build of the app and a hardware unit. They follow the same acceptance test cases and confirm that the behavior matches expectations. Their approval serves as final confirmation that the system meets the project goals.

5.5 User Testing

We gather real user feedback by letting test users interact with the app and hardware in short guided sessions. They perform tasks such as pairing a device or adjusting light colors while we observe ease of use, confusion points, and response time. After the session, they fill out a short survey covering clarity, comfort, and satisfaction. This feedback helps us refine navigation, labeling, and responsiveness. It also shows us which parts of the workflow need simplification before final delivery.

5.6 Results

So far our automated tests pass across all major components, confirming that BLE scanning, navigation flows, and color formatting logic behave as intended. Early user tests showed that users easily understood the connection flow but wanted faster feedback when selecting colors. This led us to optimize update timing in the BLE hook. System tests on real hardware verified that color commands work on both platforms, though Android required minor timing adjustments. Next steps include final acceptance testing with the client and more user testing to confirm UI polish and responsiveness.

6 Implementation

At this point in the project, our team has completed several early implementations that form the foundation for both the Arcade and the Custom deliverables of the True Force Technologies Arcade Rack. Much of our work has focused on setting up the development environment, building functional components in React, testing early Bluetooth communication, and establishing a structure that will support animation, LED control, and local data storage later in the project.

The React and Python applications are currently the most developed portion of the system. We have implemented the base routing structure, the initial screen layouts for Arcade and Custom modes, and several reusable UI components that will support both deliverables. The team has already created early versions of the Home screen, the Arcade lifting screen, the Custom mode setup screen, and preliminary leaderboard views and graphs. During this early phase, we relied on mock data for things like leaderboards. This allowed us to build and refine layouts, test the responsiveness of the interface, and adjust our design direction based on feedback from the client. We also completed the first working version of Bluetooth discovery and data reading within the React app. This work was done once the partner team resolved initial broadcasting issues on the STM32 WB55 board. Our team can now detect the device, read incoming packets, and verify that the data pipeline works in real time. These early tests relied on simple printed values and color changes to confirm that communication was reliable.

For visual design, our team created initial wireframes for app flow and explored design styles for the game's avatar, settling on 16-bit video game graphics. They looked at animators that aligned with this style, and created an animation proposal document that is currently being refined with the help of the client's input. In addition, the team is writing their own chiptune music for the arcade deliverable to make it even more authentic.

Overall, the early implementation work has successfully established the core structure of the application, validated the Bluetooth pipeline, and created the foundation for both visual and data based components. With these elements in place, the project is well prepared for the integration and refinement phases that will happen in the next stage of developmen.

7 Ethics and Professional Responsibility

Engineering ethics for our project centers on creating an interactive system that is safe, reliable, honest in its communication, respectful of users, and developed with the care expected of engineering professionals. Because our work blends hardware, embedded firmware, and a React-based software application, we ground our ethical framework in both the ACM/IEEE Software Engineering Code of Ethics and the seven professional responsibility areas described in McCormack et al. (2012). Our approach also reflects Pritchard's emphasis on character, imagination, competence, and the ongoing responsibility engineers carry even when "no one is watching".

Our team views professional responsibility as more than following rules. It involves acting with integrity, maintaining honest communication with stakeholders, prioritizing safety, and continuously improving the quality and impact of our work. Throughout this project, we have been intentional about designing a system that serves users ethically in both public arcade environments and private corporate settings.

7.1 Areas of Professional Responsibility/Codes of Ethics

The table below adapts the structure of Table 1 from McCormack et al. (2012) by aligning each professional responsibility area with the Software Engineering Code of Ethics (ACM/IEEE Joint) and describing how our team has interacted with each responsibility in our project.

Area of Responsibility	Definition (In Our Words)	Relevant SE Code Principle	How Our Team Has Engaged With This Responsibility
Work Competence	Performing work with high technical quality, accuracy, and reliability.	*Principle 3: Product*: "Ensure that their products and related modifications meet the highest professional standards possible."	We conducted team React tutorials, validated Bluetooth protocols with mock devices, and documented the backend schema carefully to avoid data corruption. This aligns with maintaining competence and delivering high-quality software.
Financial Responsibility	Managing resources, time, and budget responsibly.	*Principle 5: Management*: "Use resources in a way that ensures project quality while meeting economic constraints."	We prioritized reusing the client's hardware and codebase, avoided unnecessary purchases, and maintained features

			within the scope of a two-semester timeline.
Communication Honesty	Sharing information truthfully, clearly, and in good faith.	*Principle 2: Client and Employer*: "Be honest about limitations, risks, and progress."	Our team maintained honest updates with the client, especially when expectations seemed mixed across teams. Weekly sprint reviews ensured transparency about delays and data-integration risks.
Health, Safety, and Well-Being	Protecting users from harm and minimizing system risks.	*Principle 1: Public*: "Act consistently with the public interest."	We incorporated intuitive UI, large, readable text, robust LED feedback, and safe hardware interactions to prevent misuse. We also designed configurations to avoid misleading strength readings.
Property Ownership	Respecting intellectual property, data, and client assets	*Principle 6: Profession*: "Respect the integrity of others' work."	We handled client-provided code, assets, and hardware carefully, ensuring no unauthorized sharing. All theme assets are either original, licensed, or created via public-domain designs.
Sustainability	Minimizing environmental impact and designing durable systems	*Principle 1 & 4: Public and Judgment*: "Consider broader impacts, including environmental effects"	The design reduces waste by relying on existing hardware, creating durable enclosures, and avoiding unnecessary single-use materials. We recognized sustainability as weaker and are working to improve it.

Social Responsibility	Ensuring the project benefits society and supports fair access	*Principle 1: Public*: "Approve software only if it is safe and enhances quality of life"	We plan on building adjustable difficulty levels, optional leaderboard visibility, and walk-up-and-play accessibility to avoid discouraging or excluding participants. Also theming and styling is gender neutral.

Table 7.1 — Professional Responsibility Areas and Alignment to SE Code of Ethics

Team Strength Area

Our team is currently performing especially well in Communication Honesty.
We have established clear communication patterns through weekly sprint reviews, detailed GitHub issue updates, and shared UI/UX decisions with both the client and subteams. This transparency has prevented misalignment and allowed us to identify risks early. It reflects what McCormack et al. describe as students' tendency to excel in communication honesty relative to other categories .

Area Needing Improvement

Sustainability is the area where we need the most improvement.
Similar to the student results reported by McCormack et al., sustainability was initially under-discussed by our team, and we viewed it as only moderately relevant . While we have minimized new hardware purchases and focused on durable components, we have not yet evaluated long-term environmental impact, power consumption, or recyclability.

Going forward, we plan to explore reducing power draw during idle or attract-mode periods.

7.2 Four Principles

The table below connects the four ethical principles from Beauchamp (2007) to each broader context area in Section 4.1.1 of our design document.

Broader Context Area	Beneficence	Nonmaleficence	Respect for Autonomy	Justice
Public Health, Safety, Welfare	Encourages physical activity and creates a positive lightweight fitness experience	Avoids unsafe hardware interactions through robust enclosure and clear UI	Users can hide leaderboards and choose custom difficulty to reduce pressure	Scoring is consistent and fair, preventing biased or misleading metrics
Global, Cultural, Social Context	Provides an inclusive	Avoids stigmatizing	Players freely choose how they	Ensures the design is equally

	experience that appeals to diverse groups	rankings or exclusivity	engage (quick play, custom deliverable, anonymous play)	usable by people with varying abilities
Environmental Context	Durable hardware reduces long-term waste	Minimizes material waste and avoids unnecessary components	Stakeholders can choose lower-power modes or reduced LED intensity	Environmental impact is not unfairly shifted to users or clients
Economic Context	Supports affordable rentals and increases client revenue	Avoids unnecessary costs and prevents over-engineering	Clients can customize branding without technical knowledge	Maintains accessibility so smaller organizations are not priced out

Key Area of Importance

Public Health, Safety, and Welfare: Beneficence

This is the most important pair for our project because the Force Rack is physically interactive, public-facing, and used by people of all ages and strengths. Our team ensures beneficence by designing intuitive UI, clear visual feedback, and accurate force reporting. By reducing confusion and preventing misuse, we directly enhance public experience, aligning with the SE Code's emphasis on acting in the public interest.

Area Where the Project Is Currently Lacking

Environmental Context: Nonmaleficence

Electronic waste and power consumption are areas where the design has limited positive impact. We mitigate this by reusing hardware, but we can improve by documenting repair options, selecting more efficient LED patterns, and reducing idle-mode power draw. These improvements ensure the environmental drawbacks do not outweigh the stronger benefits in health, accessibility, and social usability.

7.3 Virtues

Drawing from Pritchard's emphasis on character, competence, and imagination as core components of responsible engineering practice , our team selected the virtues most essential to our work.

Team-Level Virtues

1. Integrity
Acting honestly in our communication, code quality, and representation of system capabilities. We maintain this virtue by documenting limitations openly, identifying risks early, and avoiding exaggeration when reporting progress.

2. Perseverance
Committed effort is needed to solve issues like Bluetooth integration, UI synchronization, and LED timing. We support perseverance by breaking difficult tasks into sprints, pairing team members on challenging components, and celebrating milestones.

3. Competence
Competence is not static. We reinforce it by learning new frameworks (React), conducting mock BLE testing, and reviewing each other's code. This aligns with the SE Code's expectation that engineers maintain professional growth.

Fadi Masannat, Demonstrated Virtue: Creativity:

Why is it important to you?
Creativity is important to me because it allows engineering solutions to go beyond baseline functionality and become experiences that feel polished, intuitive, and enjoyable. It's also a core part of building systems that are not only technically sound but also memorable and scalable.

How have you demonstrated it?
I demonstrated creativity by contributing to theme ideation, shaping the UI styling direction, and helping brainstorm the corporate customization workflow. I also applied creativity in designing the app architecture so that it remains sophisticated, scalable, and aligned with real-world development expectations.

Undemonstrated but Important Virtue: Patience
Why is it important to you?
Patience is important because complex systems rarely work on the first attempt, and careful, steady debugging often leads to better long-term reliability than rushing through problems. It also helps maintain team morale and prevents avoidable technical debt.

What might you do to demonstrate that virtue?
I can demonstrate patience by slowing down during troubleshooting, documenting each step more carefully, and giving myself time to understand issues before jumping to solutions. I also plan to ask clarifying questions earlier and approach problems with a mindset that values thoroughness over speed.

Parnika Dasgupta, Demonstrated Virtue: Communication:

Why is it important to you?
Effective and clear communication is very important in all parts of life. No one can read our mind, so it is important for us to clearly tell people what we are worried about/frustrated

about/anxious about the project in order for them to understand why we are trying to get something done and why we are behaving in a certain way. It is also good to ask for help when something is not clear. I would rather take some extra time talking to someone about a doubt I have, rather than spending hours on something that is not correct in the first place.

How have you demonstrated it?
Throughout this semester, I have acted as a bridge between the Faculty advisor, the client, and my team. I have always communicated with my team before sending emails with questions to our Faculty advisor. I have been on top of setting up meeting times, informing our Faculty advisor about everything, and making sure everyone's opinion is heard before making a decision.

Undemonstrated but Important Virtue: Punctuality
Why is it important to you?
Personally, I like to respect people's time and show up to places at the time I previously agreed upon. Everyone has busy schedules, so it is very important to keep that in mind and not keep someone waiting.

What might you do to demonstrate that virtue?
I have shown up to all our team meetings and Client meetings every week and on time. To demonstrate this virtue, I will keep showing up to meetings on time, and submit my assignments on time.

Sofi Gutierrez, Demonstrated Virtue: Adaptability:

Why is it important to you?
Adaptability is important because projects often change in ways we don't expect—whether it's technical challenges, shifting priorities, or new ideas. Being adaptable allows me to respond quickly and support the team wherever needed, which helps keep progress steady and collaboration strong.

How have you demonstrated it?
I've demonstrated adaptability by contributing in several areas of the project. I created visual assets to support the design direction, assisted with backend development when extra help was needed, and provided guidance on hardware-related questions using my Computer Engineering background. My experience with embedded systems and software integration allowed me to bridge gaps between components and serve as a resource for the team. This flexibility helped maintain momentum and strengthened our overall collaboration.

Undemonstrated but Important Virtue: Resourcefulness
Why is it important to you?
Resourcefulness is important because when projects don't go as planned, having past experiences with tools, languages, and methodology helps the team continue to move forward and know where to start after problems/challenges arise.

What might you do to demonstrate that virtue?
I demonstrated resourcefulness in multiple areas. One example is using my past internship experience with Agile methodology to help set up our scrum board for sprint planning and weekly tasks. I also leveraged my combined software and hardware background to assist teammates in both areas, bridging gaps and ensuring smooth integration across the system.

Andy Drafahl, Demonstrated Virtue: Improvisation:

Why is it important to you?
As nice as it would be if everything always went according to plan, life happens. The ability to improvise has always been helpful in my life. It keeps me from freezing up in one spot when things go wrong. Instead, I can look at problems from a different angle to find a way to push through.

How have you demonstrated it?

Throughout our project, we had to make adjustment after adjustment. Communication methods, workflows, meeting times, relevant tasks, even the entire project's design, none of these things had one-and-done, concrete solutions. I had to think on my feet as a team manager to make sure that we were still able to make solid progress every week, even when there was uncertainty about the direction we were taking.

Undemonstrated but Important Virtue: Timeliness
Why is it important to you?
The earlier materials get in, the earlier my team and I can breathe easily. Knowing that we don't have to work late at night and up to the last minute is always helpful, and reduces the overall stress we experience.

What might you do to demonstrate that virtue?

I have a goal of getting my materials in ahead of schedule so that I can better assist my team instead of rushing to the finish line in a panic.

Colin Yuska, Demonstrated Virtue: Perseverance:

Why is it important to you?
Problem-solving is the essence of engineering. If people didn't have difficult problems to solve then we would all be out of a job. That's why perseverance is so important. While the answer may not be obvious or may take a significant amount of time to reach, the ability to stick it through and come to a satisfying conclusion is an engineer's most important job.

How have you demonstrated it?
My portion of the project has primarily consisted of working in embedded systems. In the class I am currently taking, this poses some challenges however. Due to me not having

completed the class I was learning as I went and had to learn ahead of the class schedule what I needed to do. This took a lot of focus and time, by learning a completely new subject for this class I demonstrated perseverance.

Undemonstrated but Important Virtue: Scheduling

Why is it important to you?

Scheduling is how a team can get from a problem to a solution efficiently. Without specific tasks designated to be done at certain times the whole process can become convoluted and much less effective. With proper scheduling individuals and a team as a collective become much more refined.

What might you do to demonstrate that virtue?

By setting specific, measurable, and reasonable goals for certain timeframes will allow me to gauge the flow of work. Whether I am completing tasks in a timely manner or are falling behind. Without these goals timing becomes a lot more fluid. Bursts of large amounts of work connected by times of much less lead to an overall less efficient outcome.

Dylan Longlett, Demonstrated Virtue: Reliability:

Why is it important to you?

Being reliable builds trust. When people know they can count on you, work flows better and the whole team feels more confident.

How have you demonstrated it?

I show up prepared, finish tasks on time, and keep my commitments. When something needs to get done, I make sure it's handled without needing reminders.

Undemonstrated but Important Virtue: Elegance in Code

Why is it important to you?

Clean, simple code is easier to read and maintain. It cuts down on bugs and makes future updates less painful.

What might you do to demonstrate that virtue?

I can focus on shorter functions, better naming, and removing duplicate logic. I can also review old code and refactor it so the overall project feels smoother and more organized.

Jacob Garcia, Demonstrated Virtue: Consistency

Why is it important to you?

Consistency creates a stable environment for the entire group. When team members produce high quality results every single time without fluctuation trust is established effectively and operations run smoothly.

How have you demonstrated it?

I manage my time efficiently to ensure deadlines are met and obligations are fulfilled. I take ownership of my workload to ensure every task is completed thoroughly without the need for external supervision.

Undemonstrated but Important Virtue: Code Modularity

Why is it important to you?

Writing code that is distinct and separated into logical units makes a system far easier to understand and maintain. It significantly reduces the risk of errors during updates and makes the codebase more robust.

What might you do to demonstrate that virtue?

I will prioritize writing smaller and single purpose functions while enforcing strict naming conventions. I will also dedicate time to refactoring previous work to ensure the logic is streamlined and free of redundancy.

8 Closing Material

8.1 Conclusion

Frontend Development:

The team has worked on setting up the React App, integrating animations, designing the avatars, and preparing the forms for customization.

Backend Development:

Sofi and Parnika have been working on organizing backend structure documentation and working with cloud-based storage for custom and arcade modes. Sofi has also been changing all the code given to us from the previous year to Java.

Bluetooth and Hardware Integration:

Bluetooth functionality was previously a concern, but it has now been fixed, allowing testing with the rack hardware.

Arcade and Custom Deliverables:

The team has split responsibilities. The Arcade deliverable will now be in Python, and the custom deliverable will remain in React. Animation proposals, avatar designs, and UI/UX considerations are still being prepared.

Administrative and Project Management:

Andy, our team manager, assigned tasks to us every week on the Kanban board. We were all expected to finish our tasks before our team meetings. Each week, we had to update our Kanban board from ‘not started' to 'in progress' to 'done' when we got everything done before our meeting.

Client Coordination:

Parnika was our Client Liaison. She acted as a bridge between the Faculty advisor, the client, and the team. She has always communicated with the team before sending emails with questions to our Faculty advisor. She has been on top of setting up meeting times, informing our Faculty advisor about everything, and making sure everyone’s opinion is heard before making a decision.

Electronics:

Colin was our electronics specialist. He was the main facilitator in the research and development of the physical electronic portions of the project. Currently, the lighting position of this end has largely been figured out, with more research being done on adding sound elements to our final design.

Goals:

- Fully functioning Arcade version with working animations, local leaderboard, and engaging superhero-style avatars.
- Printable reports for the custom mode.

- Ensure that both modes can eventually integrate with the hardware rack.
- Keep using the Kanban board to assign tasks.
- Proceed with a split development approach.

Challenges:

- When the Bluetooth was not working, we could not test with the hardware rack.
- Early indecision about a lot of ideas delayed our progress. We kept changing a lot of ideas back and forth for a few weeks.
- We needed to shift some people to Frontend because we did not have much to do for Embedded Systems (which was previously a role).
- Managing the schedules of 7 people and finding common times was very difficult in the beginning.

Future Improvements:

- Early prototyping with hardware to prevent last-minute integration issues.
- More structured decision-making on themes and avatars to avoid delays.
- Maintain clear separation of responsibilities for arcade vs custom deliverables to avoid confusion.
- Start testing early.
- Get help early when stuck.

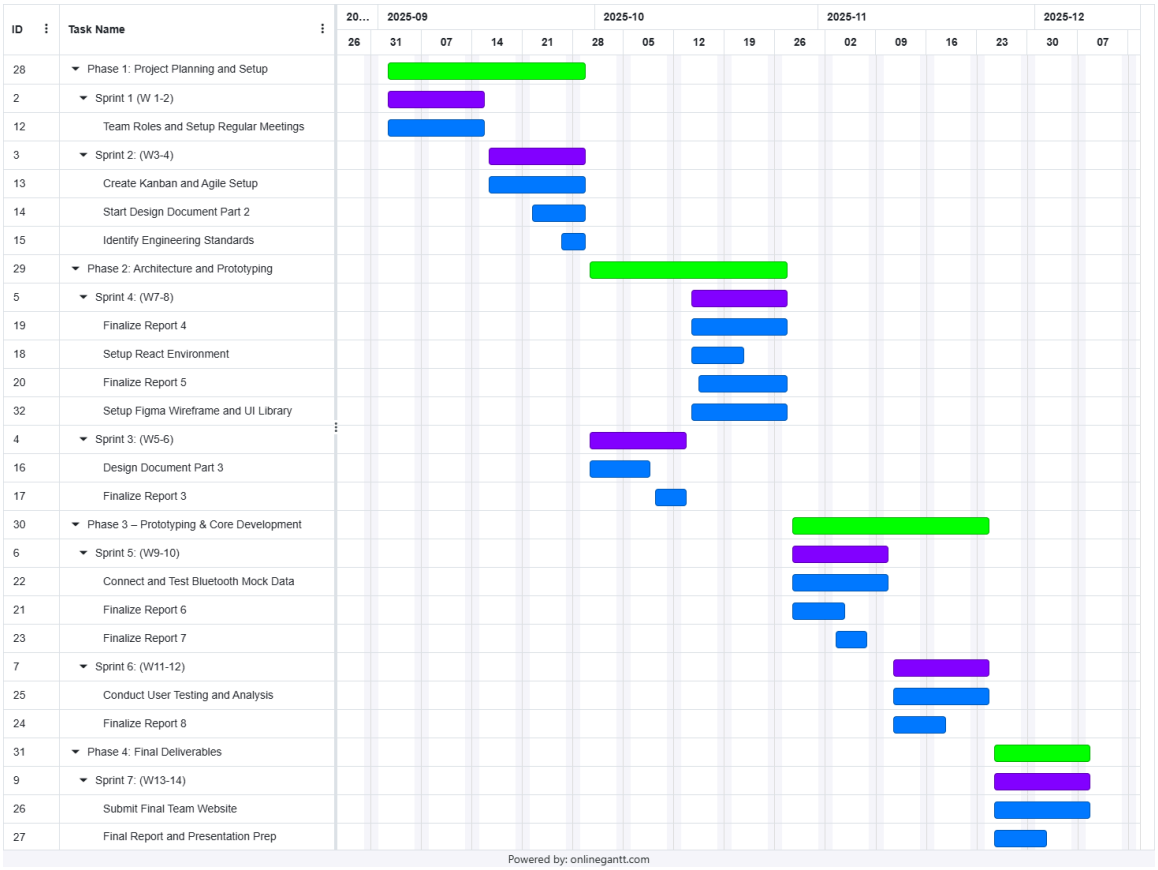
8.2 References

[1] J. Kim, H. Park, and S. Lee, “Design and Implementation of Interactive Fitness Equipment Using Force Sensors,” IEEE Access, vol. 8, pp. 123456–123465, 2020.

[2] M. Smith and A. Johnson, “Gamification in Public Health: A Review of Arcade-Based Interventions,” in Proc. IEEE Int. Conf. on Healthcare Informatics, 2021, pp. 45–52.

[3] T. Nguyen et al., “Customizable UI Frameworks for Embedded Systems,” IEEE Trans. on Human-Machine Systems, vol. 51, no. 3, pp. 210–220, 2022.

8.3 Appendices



9 Team

9.1 Team Members

- Andy Drafahl
- Colin Yuska
- Dylan Longlett
- Fadi Masannat
- Jacob Garcia
- Parnika Dasgupta
- Sofi Gutierrez

9.2 Required Skill Sets for Your Project

- Coding skills in several languages(Java, Python, etc.)
- Electronic Design
- Coding Standards
- Teamwork
- Communication
- Planning
- Graphic Design

9.3 Skill Sets covered by the Team

- Coding Skills: Fadi, Parnika, Sofi, Dylan, Andy, Jacob
- Electronic Design: Colin
- Coding Standards: Fadi, Jacob
- Teamwork: Everyone
- Communication: Parnika, Andy
- Planning: Andy
- Graphic Design: Dylan, Sofi, Colin

9.4 Project Management Style Adopted by the team

Our team adopted the Agile methodology to manage the project, as it allowed for flexibility, iterative development, and continuous feedback. We utilized Ones.com as our primary tool to implement Agile practices. Through this platform, we organized tasks using a Scrum board for user stories and priorities, managed progress with Kanban boards for visualizing workflow, and structured the project into sprints with clear goals and deliverables. This approach promoted transparency, collaboration, and adaptability, ensuring efficient progress toward our objectives.

9.5 Initial Project Management Roles

- Jacob Garcia - Frontend Development
- Dylan Longlett - Frontend Development

- Andy Drafahl - Team Manager, Frontend Development
- Fadi Masannat - Frontend Development
- Sofi Gutierrez - Backend Development
- Parnika Dasgupta - Advisor/Client Liaison, Backend Development
- Colin Yuska - Electronics

9.6 Team Contract

Team Members:

1) Fadi Masannat
2) Parnika Dasgupta
3) Dylan Longlett
4) Jacob Garcia
5) Colin Yuska
6) Sofi Gutierrez
7) Andy Drafahl

Team Procedures:

1. Meeting info:

Day: Monday (Client and Advisor Meeting) + Tuesday (Team Meeting)

Time: 1 pm (Monday), 4pm (Tuesday)

Mode: Face to face

2. Preferred method of communication updates, reminders, issues, and scheduling (e.g., e-mail, phone, app, face-to-face):

The main method of communication with each other will be over text. While communication with advisors and clients will be done over email

3. Decision-making policy (e.g., consensus, majority vote):

The majority vote was sufficient.

4. Procedures for record keeping (i.e., who will keep meeting minutes, how will minutes be shared/archived):

In every meeting, we had two people taking notes on a shared document.

Participation Expectations:

1. Expected individual attendance, punctuality, and participation at all team meetings:

We expected everyone in our group to attend our weekly team meetings on Tuesdays. We expected most people to show up during the Client meeting. One of our teammates had a work shift during the client meeting, so we excused him. We maintained good meeting notes for him to look at after his shift.

2. Expected level of responsibility for fulfilling team assignments, timelines, and deadlines:

We had a kanban board where our team leader (Andy) gave us tasks weekly. We were all expected to finish our tasks before our team meetings. Each week, we had to update our Kanban board from ‘not started' to 'in progress' to 'done' when we got everything done before our meeting.

3. Expected level of communication with other team members:

We expected everyone to look at the group chat everyday and respond no later than 24 hours. Our team members were paired together and were expected to communicate with each other every week to get the assigned task done.

4. Expected level of commitment to team decisions and tasks:

We took a majority vote for every decision. Before getting fixated on any idea, we made sure to discuss it during our team meeting and make sure all the team members were on the same page about it.

Collaboration and Inclusion

1. Skills:

- Coding Skills: Fadi, Parnika, Sofi, Dylan, Andy, Jacob
- Electronic Design: Colin
- Coding Standards: Fadi, Jacob
- Teamwork: Everyone
- Communication: Parnika, Andy
- Planning: Andy
- Graphic Design: Dylan, Sofi, Colin, Andy
- Music: Andy

2. Strategies for encouraging and supporting contributions and ideas from all team members:

We made sure that every team member had a say in what we were doing. When one of our team members did not know what something meant, we went above and beyond to help them get on the same page as us. We also had a majority vote for each idea.

3. Procedures for identifying and resolving collaboration or inclusion issues (e.g., how will a team member inform the team that the team environment is obstructing their opportunity or ability to contribute?):

At the beginning of the semester, we had a few issues with communication. We all worked towards communicating better and understanding other people's points of view. Once we all started working together as a team, we finally started getting along, agreeing on a lot of issues, and working together to fix them. Instead of ignoring the issues, we embraced them and worked our way towards solving them.

Goal-Setting, Planning, and Execution

1. Team goals for this semester:

- Fully functioning Arcade version with working animations, local leaderboard, and engaging superhero-style avatars.
- Printable reports for the custom mode.
- Ensure that both modes can eventually integrate with the hardware rack.
- Keep using the Kanban board to assign tasks.
- Proceed with a split development approach.

2. Strategies for planning and assigning individual and team work:

- Be more strict about the tasks on the Kanban Board.
- Discuss the tasks early during team meetings.
- Ask the faculty advisor for more feedback.

3. Strategies for keeping on task:

- Get more help early when stuck.
- Maintain clear separation of responsibilities for arcade vs custom mode to avoid confusion.
- More structured decision-making on themes and avatars to avoid delays.
- Start testing early.

Consequences for Not Adhering to the Team Contract

1. How will you handle infractions of any of the obligations of this team contract?

- The issue will first be addressed privately between the team member and the team manager to clarify expectations and understand any underlying challenges.
- If the infraction persists or is significant, the matter will be discussed with the full team to seek a resolution collaboratively.
- Possible actions include reassigning tasks, providing additional support or resources, adjusting deadlines, or setting clear improvement expectations.

2. What will your team do if the infractions continue?

If a team member continues to violate the obligations of the team contract despite prior discussions and corrective actions, the team will escalate the issue to the course instructor or TA for guidance. At that point, the instructor may help mediate, reassign responsibilities, or take other appropriate actions to ensure the team can meet its goals. The team's priority will remain maintaining fairness, accountability, and progress on the project.

****

a) I participated in formulating the standards, roles, and procedures as stated in this contract.

b) I understand that I am obligated to abide by these terms and conditions.

c) I understand that if I do not abide by these terms and conditions, I will suffer the consequences as stated in this contract.

1) Sofi Gutierrez DATE September 29, 2025

2) Parnika Dasgupta DATE September 30, 2025

3) Fadi Masannat DATE September 30, 2025

4) Dylan Longlett DATE September 30, 2025

5) Andy Drafahl DATE September 30, 2025

6) Colin Yuska ` DATE September 30, 2025

7) Jacob Garcia DATE September 30, 2025