

Combo Counter 3.0

Students: Calvin Mark, Daniel Mato

Mentor: *Michael H. Bucar*

Instructor: *Dr. Masoud Sadjadi*, Florida International University

Problem

With the rise of “Smart” workout technology, no current option exists for those who desire an impacted, high energy cardio based workout similar to what one would find in a training gym/ fight club. Combo Counter is designed to be a workout application which is connected to a “smart” punching bag. This bag will be able to track data metrics such as force and combo effectiveness. The goal of Combo Counter is to connect socially with others while achieving real goals and improving stamina, strength and form!

Current System

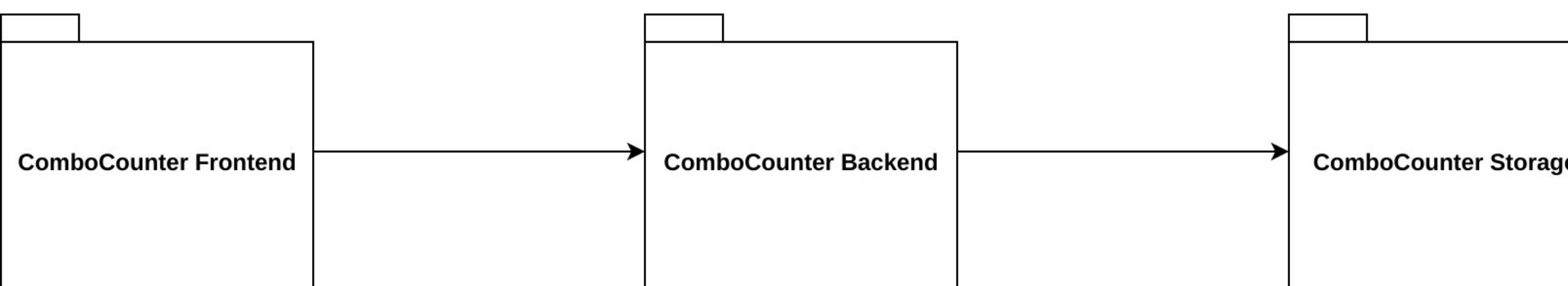
The user interface was tweaked to give a sleek look and emphasize what the user would want to focus on. Additionally, with more modes and ways to compete with others, there are far more ways to enjoy one's exercising experience. Inspiration was taken from the initial rough drafts accompanied by our design ideas to bring about a great user experience with audio and visual cues and detailed information.

Requirements

- Provide a dynamic themes for the user
- Historic view of their previous workout sessions
- Allow the customizable User Experience
- Visually consistent throughout
- Ensure reliable use of data

System Design

Combo Counter implements a 3-tier system consisting of frontend, backend and database as well as a hardware arduino component.



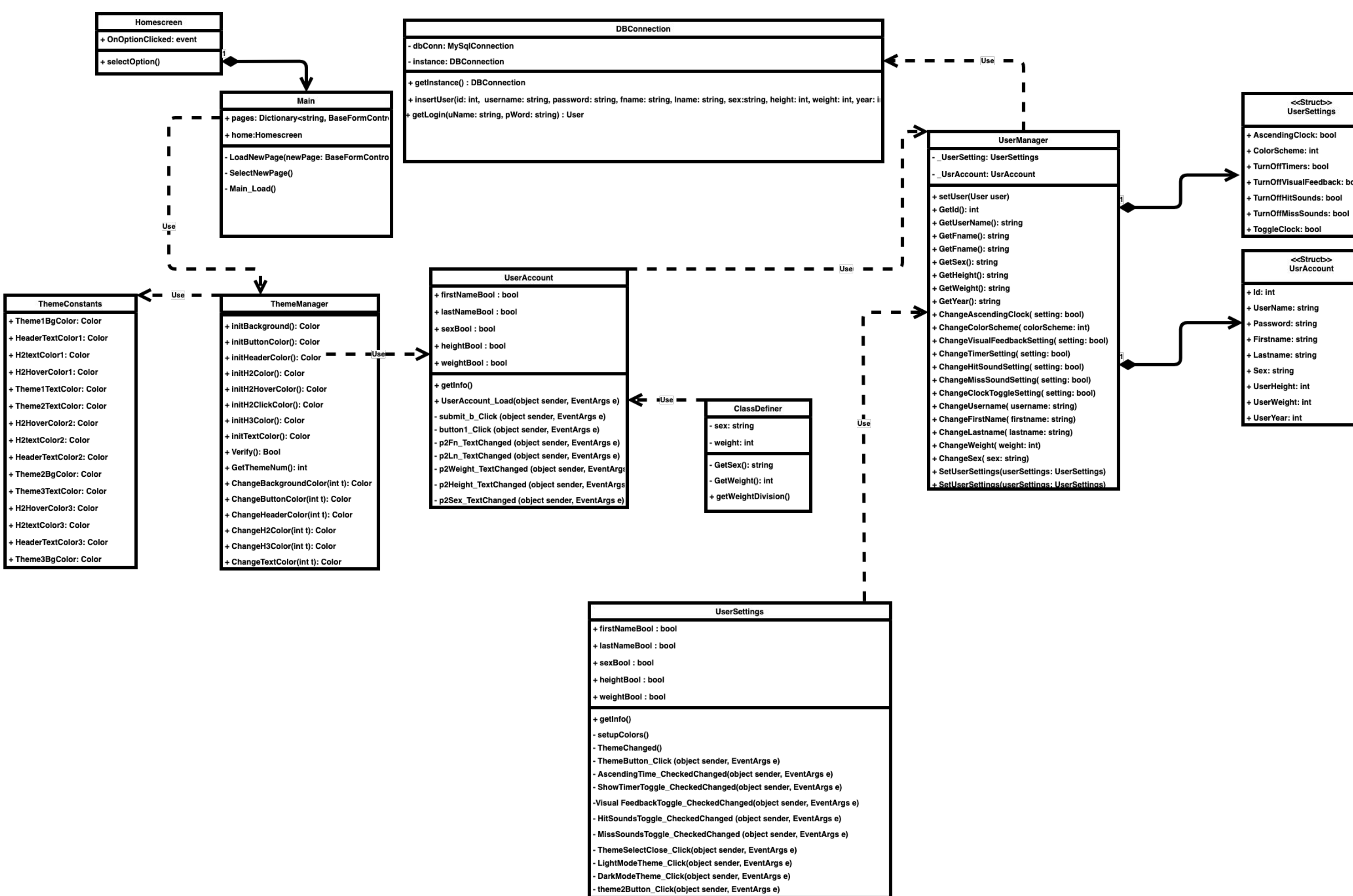
*Listed classes below were added in this version**

The front end was presented by Windows Forms manager which allowed for native design of most of the UI. Additional APIs were developed to foster custom form elements that could be uniform throughout the program. Forcing forms to only have one instance solved memory issues down the road while allowing for dynamic user themes to exist.

- **ClassDefiner.cs** - Calculates user weight division/class automatically based upon user input
- **DBConnection.cs** - Manages database logic. All functions with SQL syntax would be called here through the User Manager
- **EventManager.cs** - Manages front elements to ensure consistency throughout for version 3.0)
- **History.cs** - Manages a list of sessions so that they are accessible throughout the application
- **Session.cs** - Container to store a workout session
- **ThemeConstants.cs** - Contains individual color elements unique to theme
- **ThemeManager.cs** - Manages Overall themes and how they are implemented
- **Tools.cs** - Contains static helper methods to avoid code duplication
- **User.cs** - User object
- **UserManager.cs** - Handles communication to database

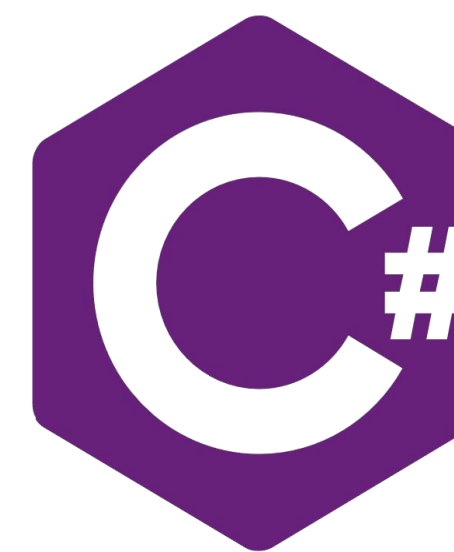
Communication to the database was called from a single class that would be handled by a manager (UserManager.cs) before being sent to DBConnection where calls to SQL database are made.. Before sending, user input is cleaned and verified for use. Changes in user settings, themes and user account info are now handled dynamically and consistently.

Object Design



Implementation

- Visual Studio Stack Utilized
- C# / C++
- Arduino IDE
- Visual Studio IDE
- MySQL Database Server
- MySQL Database Connector for C#
- Windows Forms / Controls
- Github



Verification

Application was tested manually as more features were added or old features were changed. Primary focuses were fluidity of User Interface and the handling and implementation of data to and from the Database

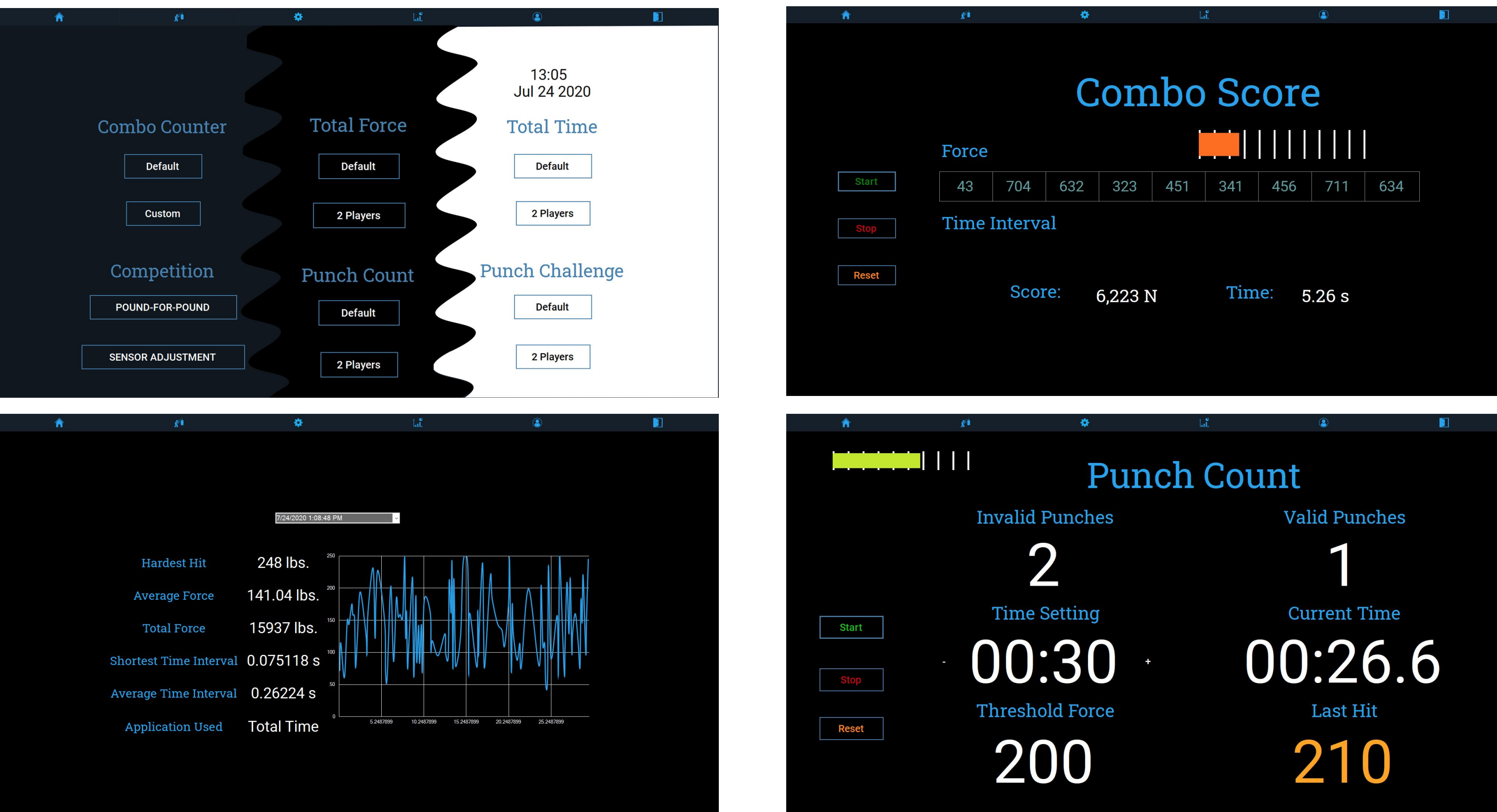
This was done with test scenarios of test cases to determine that the application was behaving according to the requirements of the product owner.

Test Case Id	unitTest_ComboCounter_02
Purpose	CC User Calvin wants to view his previous workouts
Pre-conditions	Danny is on the main page, has a valid account, has previous workouts in the database
Input	Danny clicks on the History icon on the top bar
Expected Output	The application navigates to the history page and Danny's previous workouts are shown with the correct data
Status	Pass

Test Case Id	unitTest_ComboCounter_03
Purpose	CC User Danny wants to change his first name to Daniel
Pre-conditions	Danny is on the User Account page, has a valid account
Input	Danny presses the edit button, inputs 'Daniel' into the First Name field, and presses the submit button
Expected Output	Danny's first name is updated in the database to be Daniel
Status	Pass

Test Case Id	unitTest_ComboCounter_06
Purpose	CC Developer want to ensure there is only one instance of each page
Pre-conditions	Combo Counter must be running
Input	Any icon is pressed multiple time in sequence <i>(user settings icon is selected for the purpose of this test)</i>
Expected Output	Task Manager shows a single instance of selected page
Status	Pass

Screenshots



Summary

To add improvements upon the foundation of the previous version. The database has been restructured to be more efficient. New columns were added to implement new features. Additionally, improvements were made to the overall user experience such that the application can be more responsive to user interaction while being tailored to the users preferences.

Improvement Details

- **Improved functionality & UI/UX:**
 - Dynamic Themes
 - Added fonts to the UI to avoid using the default Windows fonts
 - Added visual feedback to the workout applications
 - Standardized UI elements throughout the program
 - Standard coloring of elements
 - Removed low quality elements
 - Elements are now centered
 - Added icons to better describe feature to user
- **Database Improvements:**
 - Database code was spread out throughout the application but is now contained in one class
 - Restructured Database to implement new features
 - User can now interact with database to make changes to User Account
 - Added a system to track workout sessions and keep track of the history of workouts
 - Added User Settings to allow the user some level of customization of the program
- **Bug fixes:**
 - Fixed bug creating multiple instances of pages
 - User was originally unable to exit or quit application
 - Fixed a bug where the program would incorrectly calculate the time if the start button was pressed multiple times
 - Some applications would create multiple timers causing race conditions which would make the timer shown to be inaccurate
 - Double Clicking would cause strange behavior, resulting in loss of data
 - Changed the paths for sound effects to be relative instead of absolute to make it less user specific
 - Moved commonly used functions to a static tools class to remove code duplication

Acknowledgement

The material presented in this poster is based upon the work supported by Michael H. Bucar. We/I thank Michael H. Bucar for assistance, cooperation and mentorship that I/we received throughout this process.