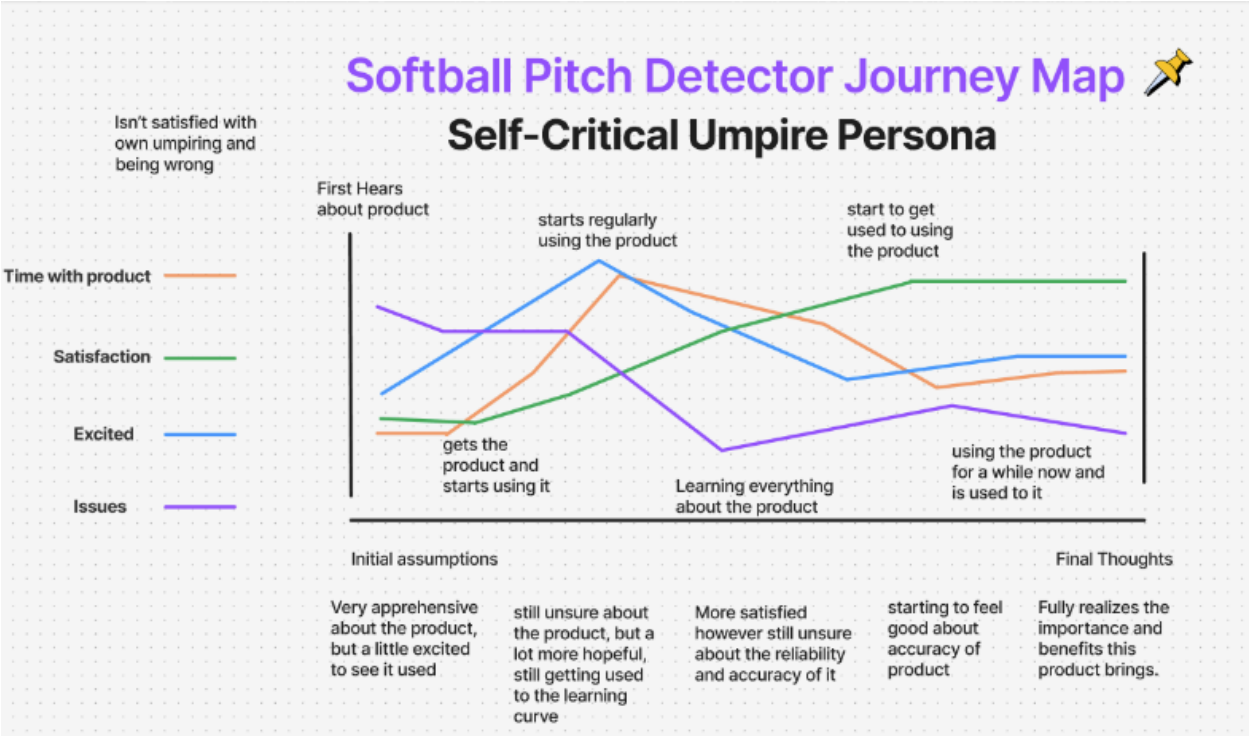
Design Check-in

Journey Map



The journey map was made to explicitly represent a self-critical umpire persona who is not satisfied with their calling abilities or wants officiating assistance.

At the beginning of the umpire's journey with our application, they may hear about the product and be unsure of the accuracy or ease of the product but interested in the potential benefits an additional officiating tool can provide their recreational league. Finally, the umpire decides to download our application and try to use it within a game.

When starting any application, there is a slight learning curve in learning the app's format. Our application's clear guidance, how-to's, and streamlined setup make for a very assistive process for all umpires who are first-time users. After an umpire reads and understands the setup/calibration steps and completes them the first time, they may be more hopeful and excited to have a new tool for managing pitches.

Continuing to use the app, the umpire experiences more satisfaction over time and becomes a regular user. Their games now have a reliable tool and umpire working to solidify the integrity of the game and its rules. Players and the umpire are now very satisfied with the product.

Pros/Cons Table

Pros	Cons
• Portability/ease of access (mobile application) • Simultaneous iOS & Android development (Flutter) • Real-time analysis	• Mobile processing limitations • Accuracy in variable conditions (e.g. lighting, camera position)

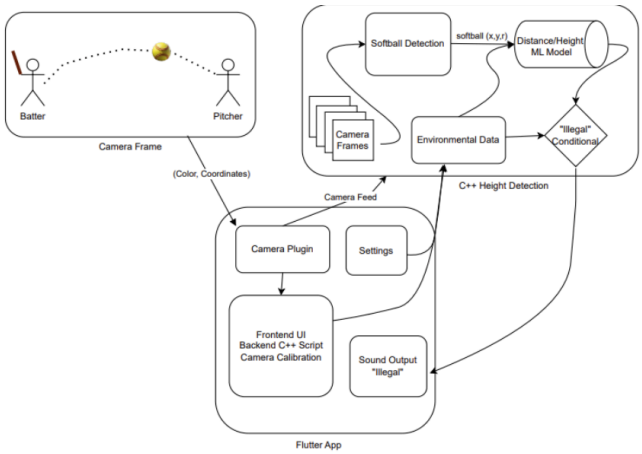
Moving into the pros and cons of our current design, our mobile app design is highly portable and user-friendly, allowing referees and players to set up before a game quickly. Developing in Flutter enables us to support iOS and Android development simultaneously, saving lots of development time. At the same time, OpenCV allows for real-time pitch analysis, delivering accurate feedback during play.

However, two main challenges are device processing limitations. Every mobile device is different, requiring efficient resource management in our computations, and the impact of variable lighting, camera positioning, and other environmental factors can affect softball recognition accuracy.

Technical Complexity Analysis

The technical complexity analysis of the system requires looking into each major functional component and its implementation process.

Our app processes video in real-time to capture accurate softball positions and heights. We use a hybrid machine learning approach for detection, then switch to object tracking to account for mobile limitations. Dart, the language Flutter is built upon, interfaces with our C++ tracking, allowing for alerting users to pitch calls.



As our app grows, we're continuously optimizing this entire pipeline to ensure our app runs as smoothly as possible on all devices. Continuous development will increase technical complexity as new solutions and frameworks may need to be implemented into the project's scope.

Human Needs

In starting prototypes for our design model, we must keep our user's needs at the forefront of our development process. Our project challenges and goals circulate around our users' three primary needs; users need an adaptive, accurate, and simplistic detection technique.

Initially, our application must read our environmental settings affecting our detection techniques. Our users expect to use this application with various field dimensions, ball colors, cameras, and lighting conditions. Our current mitigations to adapt our detection algorithms are to calibrate and gather information from the user, such as camera lens distortion, random ball color sampling, and field identification using OpenCV. These techniques allow us to modify our program to be individually suited for a specific condition to accommodate the user's needs.

Additionally, the user needs to be able to trust our application to make reliable and accurate calls. Our design is working to ensure an accurate and tested object detection and height calculation. As we continue to research height detection algorithms and techniques, we are improving our object detection by using both machine learning and non machine learning detection and tracking algorithms to pinpoint where a softball is within a camera frame reliably. Our team continues to develop and research new methods and tests for accuracy to accommodate the user's needs.

Finally, the users must efficiently and quickly understand the application and setup process. Our current screen sketches, in combination with the Flutter framework, allow us to prototype screens with adaptive buttons and descriptions to guide our users throughout the app. The streamlined calibration and camera placement setup allows umpires to be ready for their game officiating quickly.

Economic Needs

Since our system is applicable in many recreational settings with a large user base, we have economic needs to consider while developing our application. When evaluating our economic needs, we initially attempted to compare our economic impact with other products in the market. Unfortunately, there isn't an exact existing solution for what our pitch tracker does, so we developed our needs based on similar pitch trackers.

The first main economic need is pricing, our app will be free to download, which is much less expensive than existing pitch trackers that can range from a few hundred dollars for low-end models to a few thousand dollars for high-end models. This high pricing for products has remained consistent over our entire product research. Our pricing provides an economic benefit to be accessible to all users.

Additional to our availability relying on pricing, our users need an application that is easily accessible. Since we are developing our app in Flutter, we have access to cross-platform capabilities, which allow us to release our app on both iOS and Android. Designing a mobile application allows our app to be deployed to popular app stores and be downloadable by any

iOS and Android user. However, depending on the results of future testing, we may need to restrict our app to devices with a certain camera quality. While not ideal, we think it's safe to assume that most people would have a phone capable of running our app. Evaluating the economic impact of limiting our application is important to maintain and broaden our user base.

Another main economic need of our users is our application's convenience and ease. Our app will not require an external camera or sensors, which makes it easy for people to use it with minimal setup. Although this simplifies the user's experience, it causes a more difficult developmental approach to determine the height of the ball in a 3D space from a 2D frame. Evaluating the decision to simplify the user experience and complex the computational process, there is a greater advantage to accommodating the economic needs of our users with a single-camera approach with a mobile device than integrating a multiple-camera application that will deter many users.

Technical Needs

We must integrate several libraries and frameworks to complete our design when implementing our proposed solution. Our application relies heavily on two main modules: object detection and mobile development. Researching and evaluating our technical needs is essential to creating a reliable and efficient solution.

For our object detection module, we are using OpenCV as our primary camera managing software. This library exists for C++ and Python, allowing us to connect our program to the device's camera and continuously read in frames. Using OpenCV's camera frame retrieval, we can run a combination of machine learning and non-machine learning libraries and to detect a softball within the frame. YOLOv5 as a trained model, KCF as a tracking algorithm, and OpenCV as a color identification method. Combining these three tracking methods allows for a multi-layer efficient detection approach.

When deciding on a framework to develop a mobile application, we chose Google's platform, Flutter, which uses Dart to create widget-based environments with many customizable options for a unique user experience. Flutter also allows C++ to run asynchronously with the application, which is important for implementing object and height detection. Flutter's modules also help cross-platform development, which helps reduce development time by writing code that can be compiled as both an iOS and Android executable.

Our technical libraries and frameworks help make our application functional and efficient. Our technical complexity will grow as our development process continues to evolve. Considering and accommodating our technical needs is very important to creating a successful app.