Blackjack Buddy

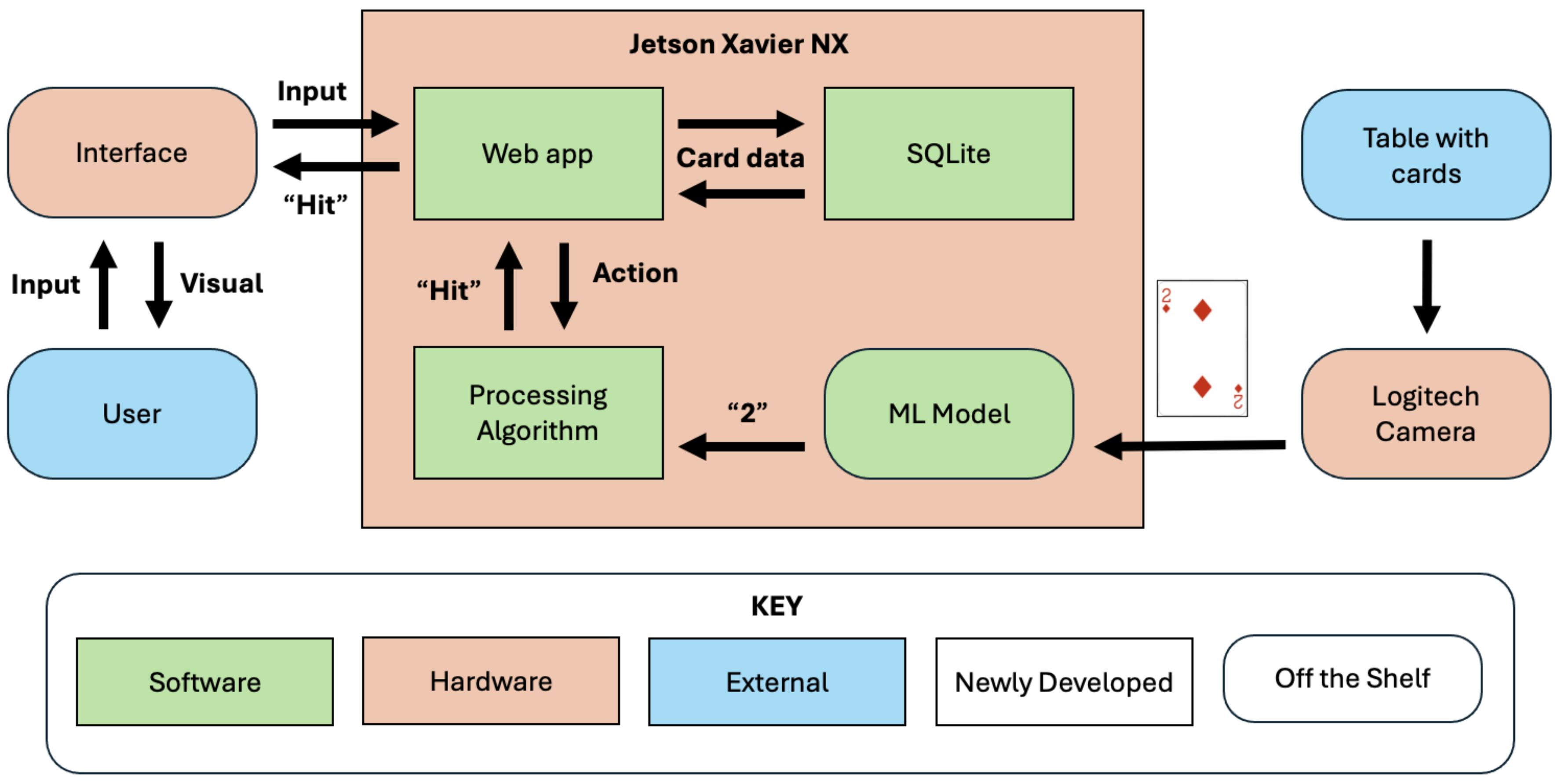
B0: Nicholas-Mesa Cucalon, Joe Sunil Inchodikaran, Lohith Tummala
18-500 Capstone Design, Spring 2025
Electrical and Computer Engineering Department
Carnegie Mellon University

Product Pitch

There is often a learning curve for playing the popular casino game, Blackjack, especially for advanced players who wish to count cards. This is dually because sources online can be unreliable and because receiving real-time feedback on gameplay isn't always available. **Blackjack Buddy** is a portable system that serves as an educational tool for blackjack players of varying experience levels. Our product detects played cards and produces the optimal decision to the user upon request via a web application. We support various counting and betting strategies in our product. As a result, users will think more critically about their gameplay in blackjack, which helps with improving profits and decision making.

Our software design meets use case requirements by detecting the cards at a high accuracy and consistently outputting the optimal decision quickly after a new cards is detected. The physical design is portable, agnostic to the table used, and the UI is intuitive to use.

System Architecture



Conclusions & Additional Information

Successes:
Blackjack Buddy offers a real-time feedback mechanism to help players of many skill levels improve their craft. Our design meets all specifications and received positive feedback from user testing, as desired.

Learning Lessons:
We learned about the bugs and mishaps that occur with integrations, especially when migrating the ML model to the Jetson, when version differences need to be fixed. This prompted us to start developing interfaces between components much earlier.

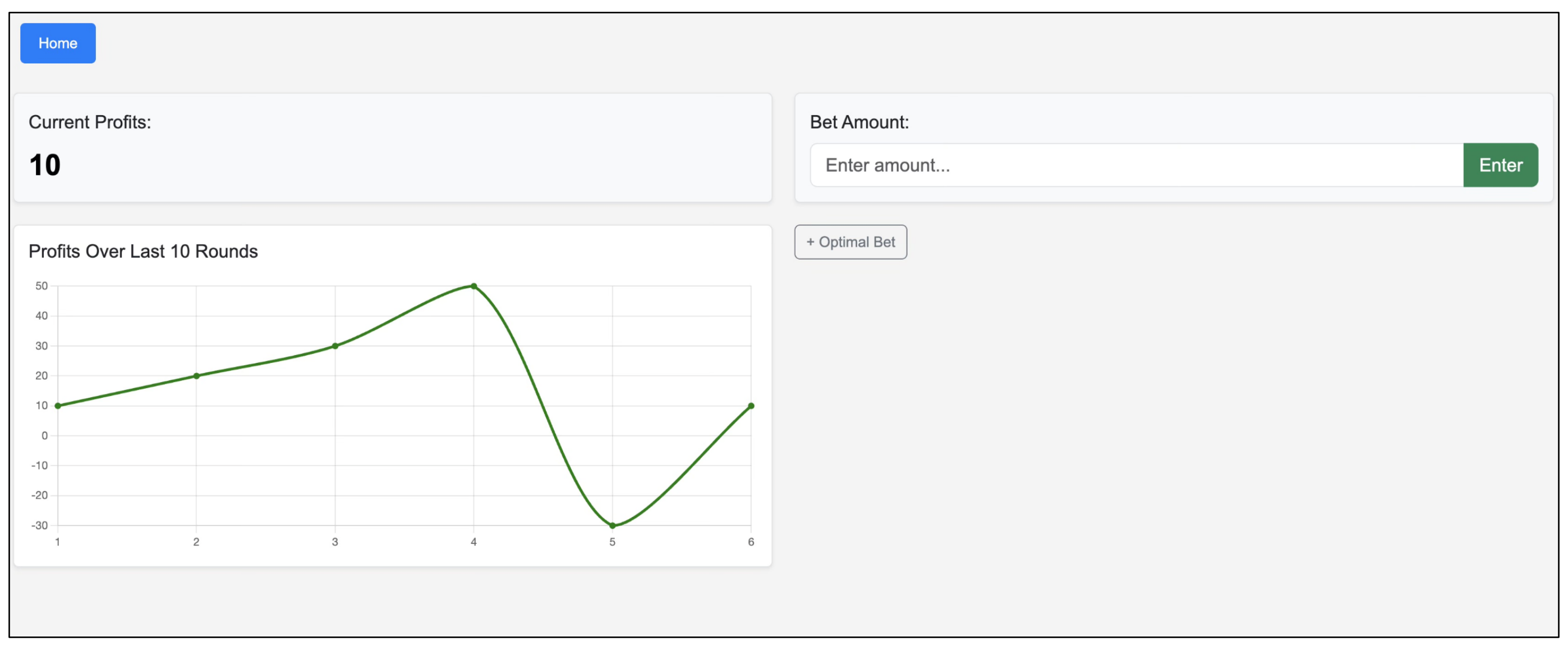
Improvements:
We wish to add support for multiple players and their hands. We also wish to integrate a multi-camera system so that we can detect various hands on a larger table. A cool extension of this project would be a pair of glasses with camera mini-screen.

System Description

Card Detection: To detect the dealt cards, a TensorRT Accelerated ML Computer Vision model is used. The model is specially trained to correctly detect obstructed cards in any background.

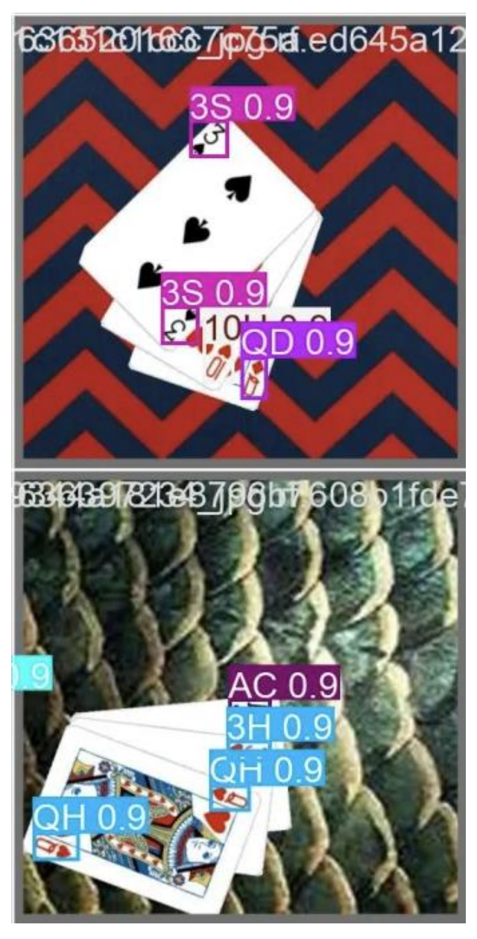
Processing and Interface: A Python web application is used to take in user input and display the optimal gameplay decision. The web app has customizable settings and tracks profits over the last ten rounds.

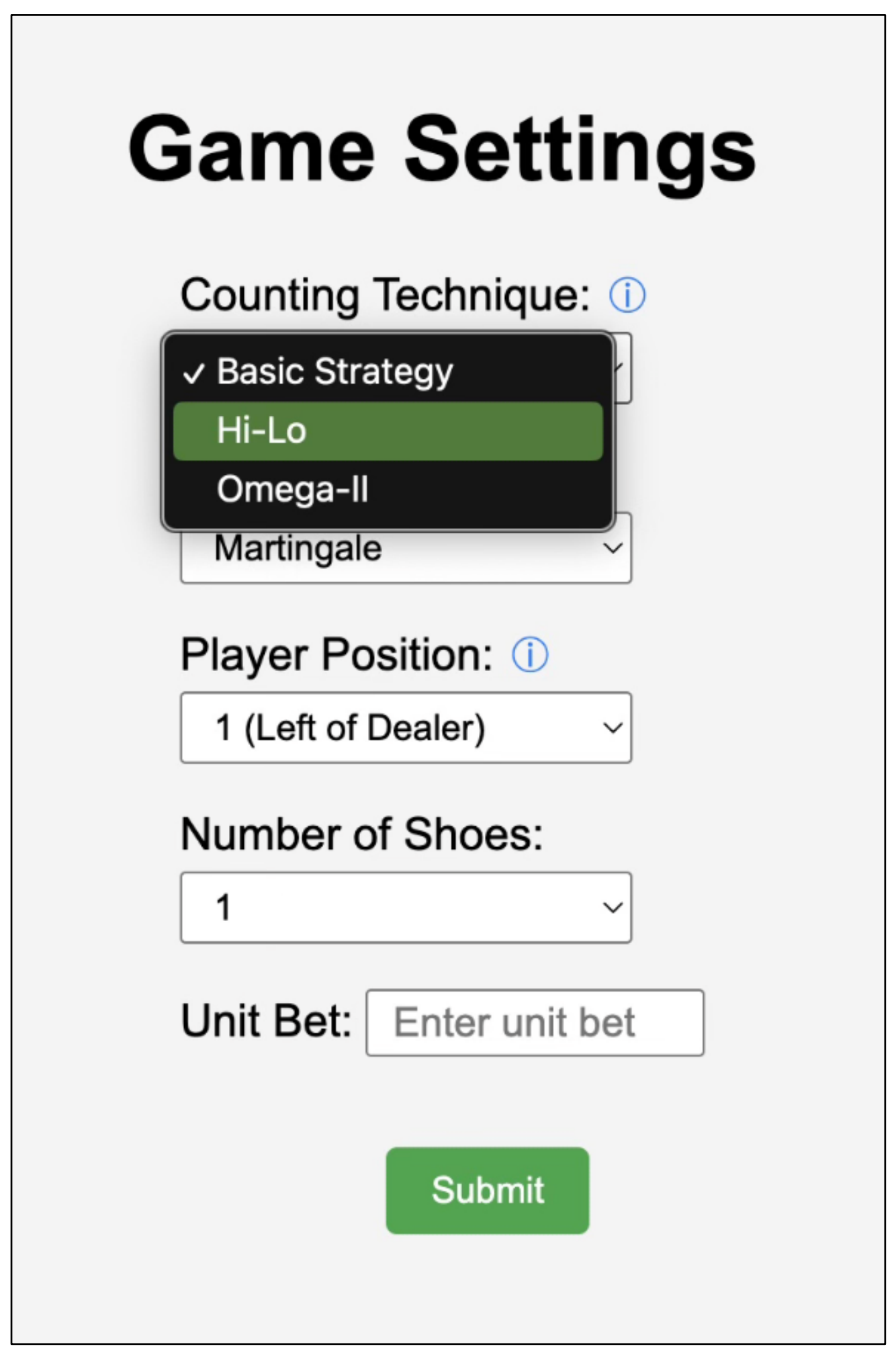
Hardware System: The web app, processing, and ML model are hosted on a Jetson Xavier NX, which is housed in a stable wooden box with a camera mount.


Current Profits:
10

Bet Amount:
Enter amount...

Profits Over Last 10 Rounds




Game Settings

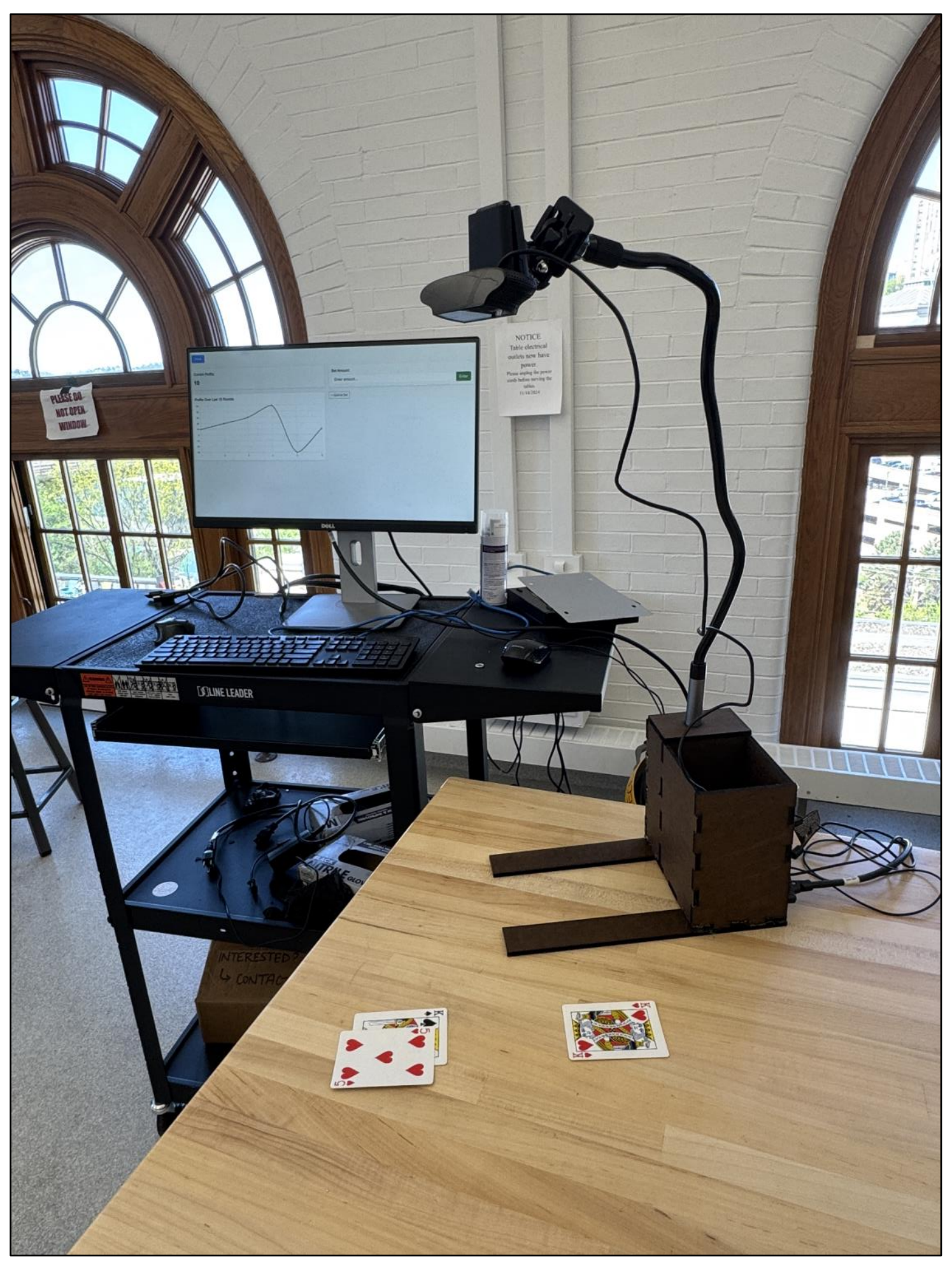
Counting Technique:
✓ Basic Strategy
Hi-Lo
Omega-II

Martingale

Player Position: 1 (Left of Dealer)

Number of Shoes: 1

Unit Bet: Enter unit bet



System Evaluation

Latency Requirements:

Metric	Target	Actual
Camera Latency	< 50 ms	33.8 ms
Detection Latency	< 300 ms	253.68 ms
Processing Latency	< 100 ms	0.1 ms

Accuracy/UI Requirements:

Metric	Target	Actual
Detection Accuracy	> 90%	98.7%
Processing Accuracy	= 100%	100%
User experience	≥ 8/10	8.5/10

Design Tradeoffs:
We opted to use TensorRT instead of optimizing on CUDA kernels for ease of deployment despite the performance loss. We also had access to a much more powerful camera (OAK-D Pro), but opted for the Logitech camera, which fit our use case.
We also had to downsize the model to accommodate the Jetson's GPU, with minimal losses in accuracy.

Electrical & Computer ENGINEERING
Carnegie Mellon