ENGPHYS 4A06:
Engineering Physics Design & Synthesis Project

Automated Blackjack Dealer Using Computer Vision:
Blackjack Orama

Team Members: Samuel Ahlin, Muqtadir Hussain, Sunil Balendran

CONTENTS

Elevator Pitch

Blackjack is a card game where players compete to reach a sum of 21 without going over. It traditionally relies on human dealers to manage the game.

But what if the table could run itself?

We're bringing a machine-learning twist to a $250 billion industry with ***Blackjack Orama***[1]—an automated system that uses computer vision to recognize gestures, delivering a seamless experience that you can recreate even at home, all while eliminating the need for a dealer.

With ***Blackjack Orama*** we're redefining Blackjack, one gesture at a time.

[1] Orama means 'vision' in Greek to represent the use of computer vision in our product.

Executive Summary

BlackJack Orama, as seen in Figure 1, is an automated Blackjack system that merges gesture recognition, computer vision, a mechanical dealer, and a custom graphical user interface (GUI) to deliver a seamless card game experience. The user interacts with the game through hand gestures detected by a Landmark-based model, which classifies actions such as "Hit" (index finger pointing) and "Stand" (open palm). These gestures are processed by a Python-driven Blackjack logic engine that determines the next move. If a card is to be dealt, a command is sent via serial communication to an Arduino-controlled mechanical dealer, which uses a servo-driven mechanism to dispense a physical playing card. A camera mounted above the table captures the dealt card, which is recognized by a YOLOv8 model trained to identify card ranks and suits. This information is sent back to the game logic to update scores, display real-time game states, and provide visual feedback through a Tkinter-based GUI. The GUI shows player hands, scores, and dealer actions, enhancing clarity and user engagement. This closed-loop system allows for intuitive, interactive gameplay that combines AI perception, physical responsiveness, and user-friendly interface design.

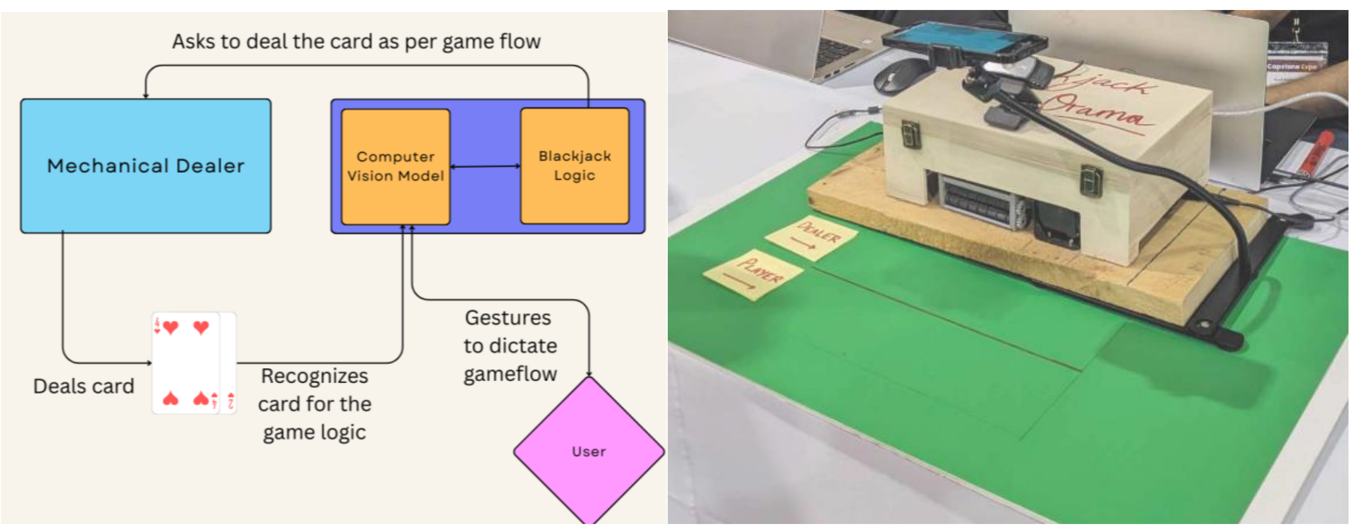


Figure 1. System flow overview and final working design demonstrated at capstone expo.

Mechanical System:

The mechanical system of our autonomous Blackjack dealer underwent a structured and iterative design process, evolving from an initial LEGO-based prototype into a fully integrated, precision-engineered card dispensing unit. This system serves as the physical interface between the digital game logic and the tangible deck of cards, autonomously executing the act of dealing cards during gameplay.

Initial Prototype: Top-Ejection LEGO Mechanism

Our first functional prototype was built entirely using LEGO blocks, seen in Figure 2. It featured a static card holder and a dynamic motorized control arm, which rotated a rubber-lined wheel mounted on an axle. The wheel rested on top of a vertically stacked deck and dispensed the topmost card by spinning briefly in the forward direction. A short reverse spin followed to re-align the deck if disturbed. This mechanism proved the concept of single-card ejection but introduced major mechanical shortcomings. As cards were dispensed, the deck height decreased, which caused the control arm to sink. Without a dynamic support structure for the motor at the opposite end of the axle, the axle would sag, causing rotational misalignment and frequent jams.

Figure 2. Initial prototype demonstrated in Fall 2024, designed primarily with LEGO.

Intermediate Design: Bottom-Ejection Concept

To address the stability issues of the LEGO prototype, the second design iteration moved the driving roller beneath the card deck. A slot in the receptacle floor allowed a rotating wheel to make direct contact with the bottom-most card, eliminating the need for a moving control arm entirely, as seen in Figure 3. Although this version was never motorized, it introduced the critical concept of bottom-up ejection: the card is not dispensed from the top but primed from below and guided out by additional mechanical components. A schematic was drawn, and a 3D-printed build was produced (Seen in Appendix: Week 4) to validate alignment, card feed angles, and component housing tolerances. The intended mechanism included two front-mounted opposing rollers, which would grip and pull the primed card out of the device, improving control and preventing misfires. Though not yet implemented in this phase, these rollers would prove crucial in the final design.

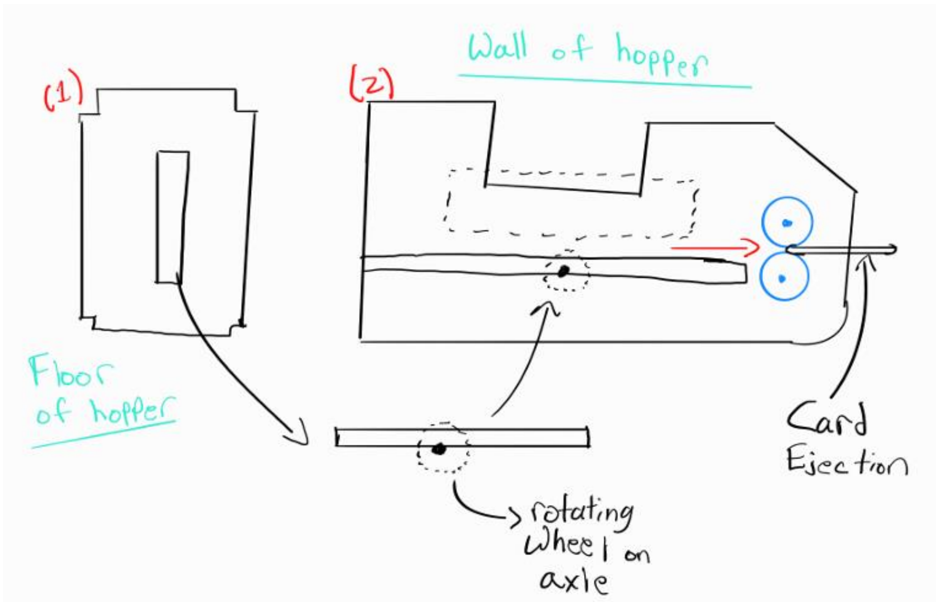


Figure 3. Conceptual design of second iteration of mechanical design

Final Design: Precision-Controlled Card Ejection System

The final mechanical build, seen in Figure 4, represents a culmination of our previous iterations, incorporating robust materials, improved friction control, and precise motor actuation. Key features include:

- **Primer Roller with O-Rings (Bottom Drive)**:
  A horizontal roller lined with nitrile O-rings (later replaced with silicone) sits beneath the bottom card. This roller "primes" the card by pushing it slightly forward into the ejection path.
- **Dual Front Ejection Rollers**:
  Two parallel rollers positioned at the front of the receptacle rotate in opposite directions to pull the card through a narrow channel. These rollers are geared together to ensure perfect counter-rotation and synchronized motion.
- **All three rollers** are supported by dual bearings, one on each side. The bearings were 3D-printed and sanded to fit the mold within our design. A wall is screwed onto the side opposite the motors, using 2.6M machine screws, which act to hold the rollers in place and prevent axle sag.
- **Stepper Motor Control & A4988 Drivers**:
  Two NEMA 17 stepper motors are used, which have 1.8-degree steps:
    - One drives the bottom primer roller.
    - One drives both ejection rollers.
    - A pair of A4988 stepper motor drivers interface with an Arduino Uno to control timing, direction, and torque.

- This setup offers:
  - Effective current control of each motor
  - High holding torque (17 Ncm) to prevent slip
  - Independent motor logic for sequential actions
  - Small heatsinks for the A9488 drivers to help with heat dissipation.
- **Custom Electronics Housing**:
  All components are enclosed in a wooden box, seen in Figure 5, which holds the Arduino, breadboard circuit, and power distribution wiring. The receptacle is modular and accessible, allowing easy reloading of cards.

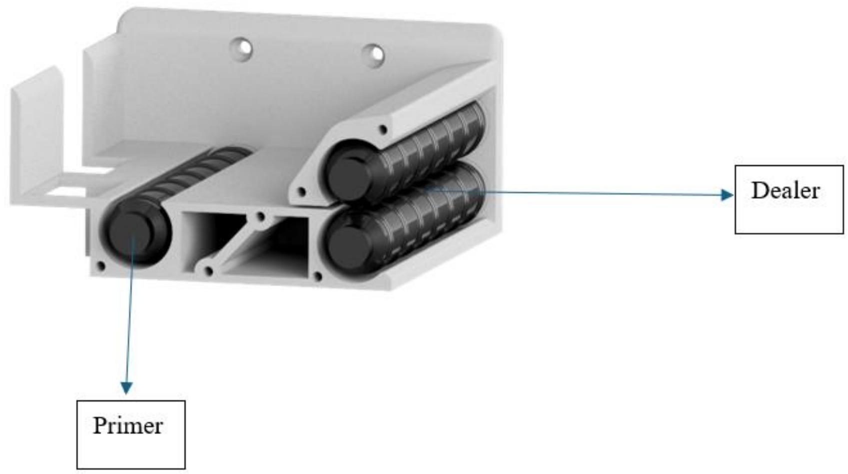


Figure 4. CAD render of final design, showcasing the primer and dealer rollers

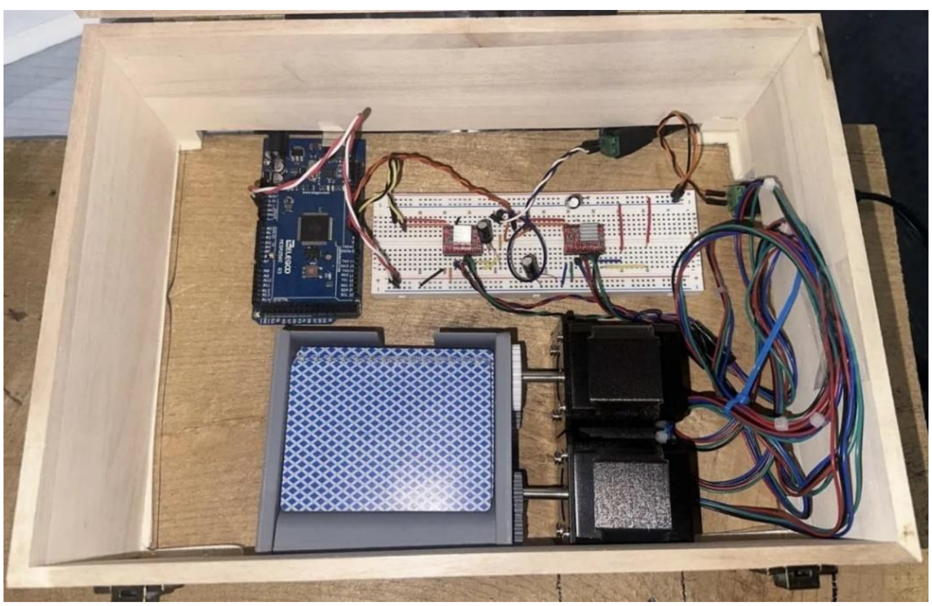


Figure 5. Interior arrangement of finally mechanical design and support circuit hardware

The final iteration of the mechanical system marks the culmination of several design phases, resulting in a compact, highly functional card dispensing unit that integrates seamlessly with the software-controlled gameplay of the autonomous Blackjack dealer. Building upon previous prototypes, this version resolves earlier mechanical limitations through a stable base structure, fine-tuned motor control, and high-friction engagement mechanisms. All aspects of the design — from structural rigidity and material choice to motion sequencing and component alignment — were optimized to produce a reliable and replicable dispensing cycle for physical playing cards.

The structure of the final design was fabricated entirely through precision 3D printing. This allowed for the creation of a dense, rigid receptacle compartment to contain a standard 52-card deck, as well as precisely dimensioned slots and channels to support roller-based card movement. The receptacle was designed with close-fitting side walls and a clean vertical cavity to constrain lateral card movement. A cylindrical slot in the base of the receptacle exposed the bottom-most card to a rotating roller underneath, setting the foundation for a bottom-ejection architecture. This fundamental shift — from top-down to bottom-up card dispensing — eliminated the primary mechanical issue observed in the LEGO-based prototype, where an overhead motor arm would sag and misalign as the card stack height decreased. With the bottom-driven mechanism, the contact point with the deck remained static throughout the game, regardless of how many cards had been dispensed.

The card dispensing process begins with a single horizontal roller mounted beneath the card stack. This roller, fitted with evenly spaced silicone O-rings, spins rapidly for a brief, calibrated duration to push the bottom card slightly forward. The use of silicone was a critical improvement over nitrile in earlier tests. Silicone O-rings provided a higher coefficient of friction, enhancing the grip between the roller and the bottom card while maintaining minimal friction between the bottom card and the one above it. This differential was key to ensuring the roller reliably primed only one card at a time. The forward motion imparted by the primer roller positioned the card into the front of the receptacle, where a second stage of the dispensing mechanism would take over.

To complete the card ejection, two horizontally aligned rollers were mounted at the front of the device. These rollers, also wrapped with silicone O-rings, were oriented in opposing directions and rotated synchronously to grip and pull the primed card through a narrow exit channel. The mechanical linkage was achieved through a set of intermeshed 3D-printed gears, which ensured that one motor could control both rollers while maintaining precise counter-rotation. The grip provided by the O-rings and the balanced torque between the rollers allowed for confident and smooth ejection of each card. A bipolar NEMA 17 stepper motor, connected to the roller assembly, powered this process with a high holding torque of 17 $N \cdot cm$ and precise angular control.

Each roller stage — the primer roller and the ejection rollers — was independently powered by its own NEMA 17 stepper motor. The motors were driven by separate A4988 stepper motor drivers, which interfaced with an Arduino Mega microcontroller. The electronics were arranged on a custom breadboard setup and neatly housed alongside the receptacle inside a laser-cut wooden enclosure. This modular box housed the full mechanical-electrical interface of the system, keeping wiring clean and allowing easy access for modifications. The control firmware was written to ensure that the primer motor operated in short bursts of high-speed rotation to initiate movement, while the ejection rollers engaged immediately after with slower but more powerful rotation to complete the dispensing cycle and overcome any mechanical resistance.

During testing, the system demonstrated strong consistency in dispensing one card at a time. However, early versions encountered occasional jamming when two cards were primed instead of one. In such cases, the double-thickness card would wedge between the ejection rollers, causing motor stalls and localized heating. These issues were systematically addressed through friction optimization — by switching to silicone O-rings — and through firmware-level refinements to the timing, duration, and speed of motor activations. Separating the power delivery and control logic for each motor further allowed for asymmetric tuning of torque and speed, enabling each roller stage to perform its specialized task without interference.

Ultimately, this final mechanical system proved capable of autonomous and accurate card dispensing, fully integrated with the software stack that drives the Blackjack gameplay. It is compact, mechanically stable, and modular, with design features that allow for future upgrades, such as a rotating base to dispense to multiple players, or an embedded camera slot to scan cards during ejection.

SOFTWARE SYSTEM:

Design

At the core of the *BlackJack Orama* system is a graphical user interface (GUI) that connects and coordinates all major software components, seen in Figure 6. The GUI serves as the central hub, displaying real-time game information such as player hands, dealer cards, and scores, while also providing visual feedback based on user gestures and detected cards.

- The Blackjack Logic module handles the core game flow — including score tracking, win/loss conditions, and turn progression.
- The Gesture Recognition Model interprets player hand gestures to control game actions like “Hit” or “Stand.” It feeds this input directly into the Blackjack logic, replacing traditional keyboard controls.

- The Card Recognition module captures images of physically dealt cards and identifies their values using a trained YOLOv8 object detection model. These results are passed to the logic engine to reflect accurate in-game card states.

The Gesture Recognition and Card Recognition act as the inputs from the user into the blackjack logic, which communicates directly with the Mechanical Dealer to deal cards accordingly.

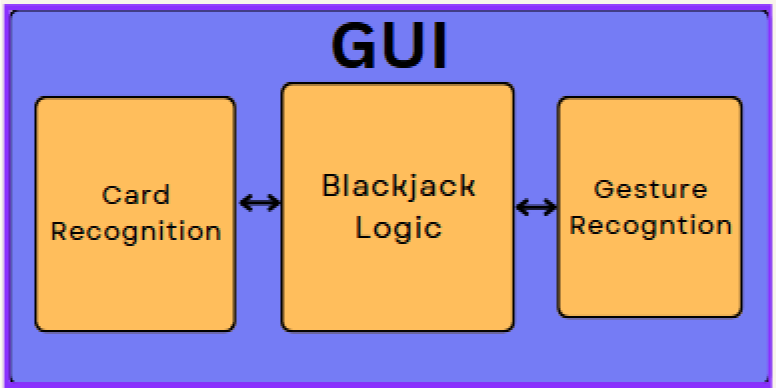


Figure 6. Simplified diagram of information flow for GUI

Implementation

Gesture Recognition

Getting into specifics, the gesture recognition system is a robust system that uses a neural network and real-time hand key point detection to recognize gestures. The process begins with capturing the input, where a live video feed detects the hand. MediaPipe's Hands Module[2] is a pre-built model that recognizes hands and landmarks the joints, giving the co-ordinates as the output, seen in Figure 7. The module extracts 21 key landmarks, representing critical points like the fingertips, joints, and wrist. The detected coordinates undergo pre-processing to make the system account for variations in hand position, size, and orientation. This involves normalizing the coordinates, translating them relative to the wrist as the origin, and flattening the 3D data into a single-dimensional vector for compatibility with the neural network. This pipeline is visualised below:

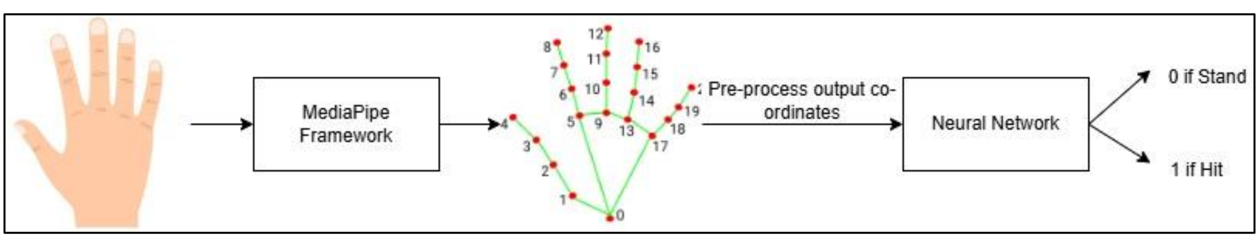


Figure 7: Flow Chart of Gesture Recognition System

[2] https://mediapipe-studio.webapps.google.com/demo/hand_landmarker

The neural network, a multi-layer perceptron (MLP), processes the pre-processed data to classify gestures. Drawing inspiration from a Japanese open-source sign language recognition project[3], the neural network was tailored to recognize gestures specific to the Blackjack system. For this application, the defined gestures include an open palm for the "Stand" action and an extended index finger for the "Hit" action. The model was trained on a comprehensive dataset of over 4,000 labeled gestures, enabling it to deliver reliable recognitions across diverse conditions, including varying hand orientations, positions, and lighting environments. This was critical to address the dynamic movements and background uncertainties expected during capstone demonstrations and the expo. The overall output of the neural network is binary:

- “0” for Stand
- “1” for Hit

The system integrates seamlessly into the Blackjack game, activating the camera for gesture recognition whenever player action is required. During gameplay, the camera remains open for 10 seconds to capture multiple instances of gestures, recording a list of zeros and ones. The most frequent value in this list is selected to determine the player’s action. This approach mitigates errors caused by rapid or inconsistent hand movements, ensuring accurate gesture recognition.

The model achieves an impressive accuracy of **99.88%** in recognizing the two gestures, a significant milestone for the team and a testament to the system's reliability.

Card Recognition

The card recognition system in *BlackJack Orama* utilizes computer vision and deep learning to automatically identify the rank and suit of physically dealt playing cards, seen in Figure 8. This system plays a critical role in ensuring the Blackjack logic reflects the actual cards in play, allowing for a fully automated experience. The recognition process begins when the mechanical dealer dispenses a card — at this point, a camera module captures an image of the card placed on the table.

To process the image, the system employs a YOLOv8 [4]object detection model, trained specifically on a custom dataset of playing cards. This dataset, compiled using images of real cards in various orientations and lighting conditions, was augmented through Roboflow to improve robustness. It includes over 3,500 annotated images, covering all 52 card types from multiple angles and under varying shadows and reflections.

Once an image is captured, it is passed to the YOLOv8 model, which outputs the bounding box, class label (e.g., "7 of Hearts"), and confidence score. The class label is then parsed and forwarded to the Blackjack game logic, which updates the player or dealer hand accordingly.

[3] https://github.com/Kazuhito00/hand-gesture-recognition-using-mediapipe
[4] https://github.com/TeogopK/Playing-Cards-Object-Detection

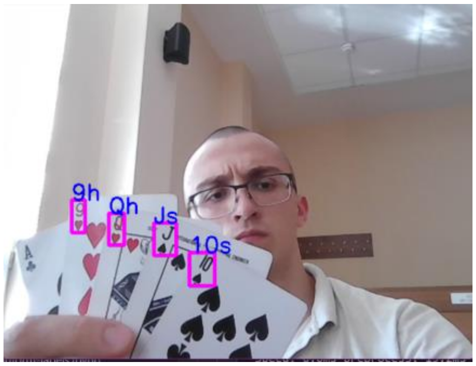


Figure 8. Visualization of computer vision recognition of rank and suit of playing cards

GUI and Blackjack Logic

The Blackjack game logic governs the flow of the game — from card distribution and turn-taking to hand scoring and outcome evaluation. The logic is built in Python and adapted from an open-source, terminal-based implementation. The original version relied on keyboard inputs to progress the game, but in *BlackJack Orama*, these inputs are replaced with gesture recognition and card detection systems, enabling a fully automated and interactive experience.

The Blackjack logic handles:

- Dealing cards to both the dealer and player.
- Tracking the state of each hand.
- Determining whether a player has won, lost, or busted.
- Managing dealer behavior (e.g., auto-hit until 17+).
- Syncing all moves with real-time user input and physical card detection.

To present this logic intuitively, the system is wrapped in a custom Graphical User Interface (GUI) built using Tkinter, Python's standard GUI library. The GUI provides a clean and dynamic visual of the game state, updating in real time based on player gestures and the cards dealt by the mechanical system. Key GUI features:

- Real-time Hand Display: Both the dealer's and the player's cards are shown on screen as they are detected by the card recognition model, including card rank and suit.
- Score Tracking: The interface continuously updates and displays hand totals for both the player and the dealer, enabling clear visibility into the game state.
- Gesture Feedback: Recognized gestures ("Hit" or "Stand") are shown live on the screen, confirming that the system correctly interpreted the player's input.

- Game Status Messaging: The GUI communicates essential gameplay updates, such as “Waiting for Player Gesture,” “Dealer Hits,” or “Player Busts,” making the flow easy to follow.
- Turn Management Visualization: The interface indicates whose turn it is, highlighting when the player should make a gesture or when the dealer is acting.
- Result Announcement: At the end of each round, the GUI clearly declares the result — win, loss, or draw — and prompts for a new game if desired.

CONCLUSION

Blackjack Orama successfully demonstrates the integration of mechanical engineering, computer vision, and software systems to create an automated, gesture-controlled Blackjack dealer. Over eight weeks, we iteratively refined our design—transitioning from LEGO prototype to a precision 3D-printed mechanism and robust Python-based software stack. The final device achieved an 85% success rate for single-card deals and reliably detected gestures and playing cards in real time. Key successes include seamless hardware-software communication, a fully functional GUI, and significant mechanical improvements like silicone rollers and dual power sources. However, challenges such as inconsistent double-card deals and unsuccessful shuffler integration highlighted areas for further development. Despite this, Blackjack Orama stands as a strong proof-of-concept for an autonomous dealer system, pushing the boundaries of interactive gaming with AI and automation.

APPENDIX

Week 1

Problems Solved/Progress:

We touched base during the first week of the semester and outlined our steps for reaching a successful demo to the faculty at the end of February/March. At the beginning of the winter term, we still need to make significant progress on the mechanical design. We need to complete our next iteration from our demo in the Fall term, which was designed primarily with LEGO. As LEGO was always a proof of concept, we need to have something far more robust. This includes designing a model in CAD that encompasses two primary modules: support for mounting the shuffler to the dealer, and the dealer design itself.

From a software perspective, we need to independently develop the module for card recognition, make the software compatible with the mechanical by designing it to operate motors, and design a GUI for player-usability. Our current path forward for card recognition is embedding a camera sensor in the receptacle that holds the card before they are dealt. Nice to have at this point is the rotating base, however we do not view it to be as necessary as having a functioning card dealer.

Problems Identified:

We know that it is imperative that we start printing as soon as possible. The mechanical design will be complicated and will never work initially, as was the case with our first demo.

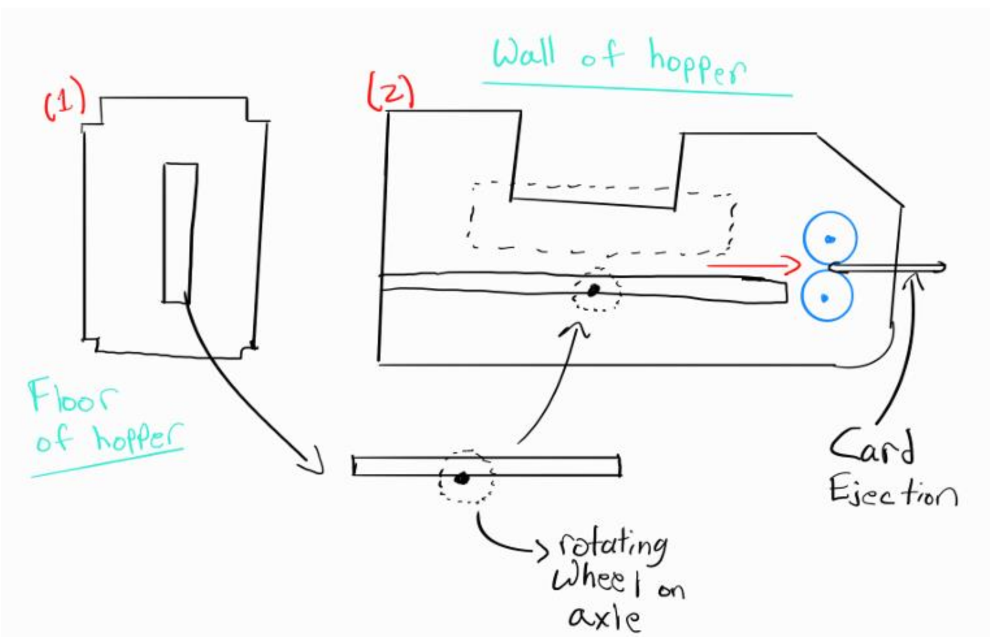
Next Steps:

- Focus on designing the mechanical components. The dealing mechanism is the most important and will need a lot of trial and error.
- How will we detect the cards that have been played?
- It is not immediately necessary, but over the next month the software will need to be translated to Arduino commands.
- Even if it is a rudimentary system, we need to decide how the player will receive information about the game status. How will they know the game is being run correctly and the dealer is doing what it should? We require a GUI to display information of the player and dealer hands, or alternative small LED display of some sort.

Week 2

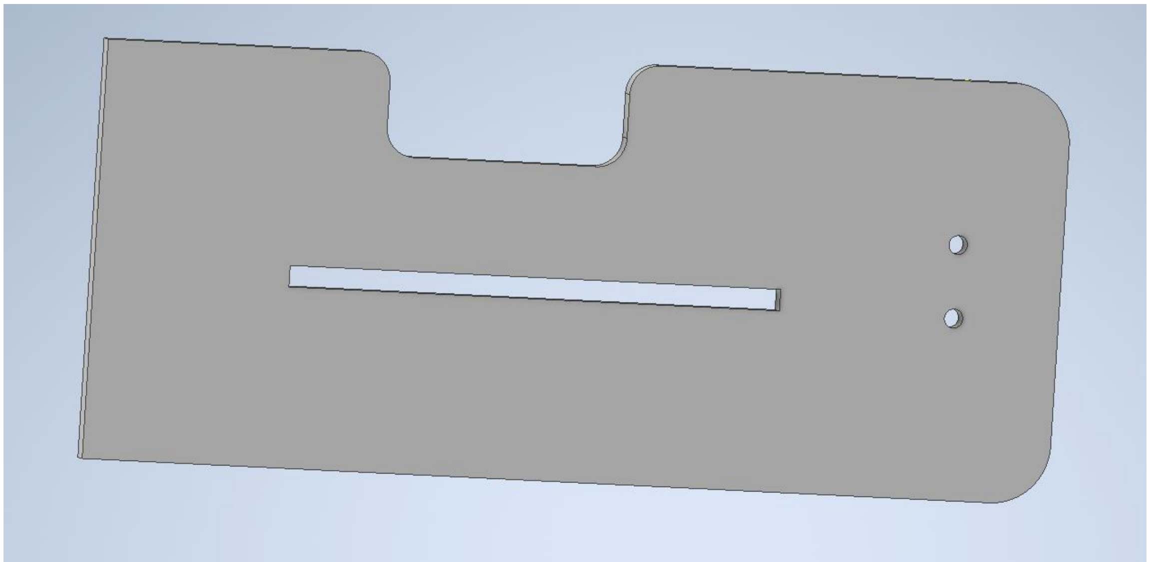
Problems Solved/Progress:

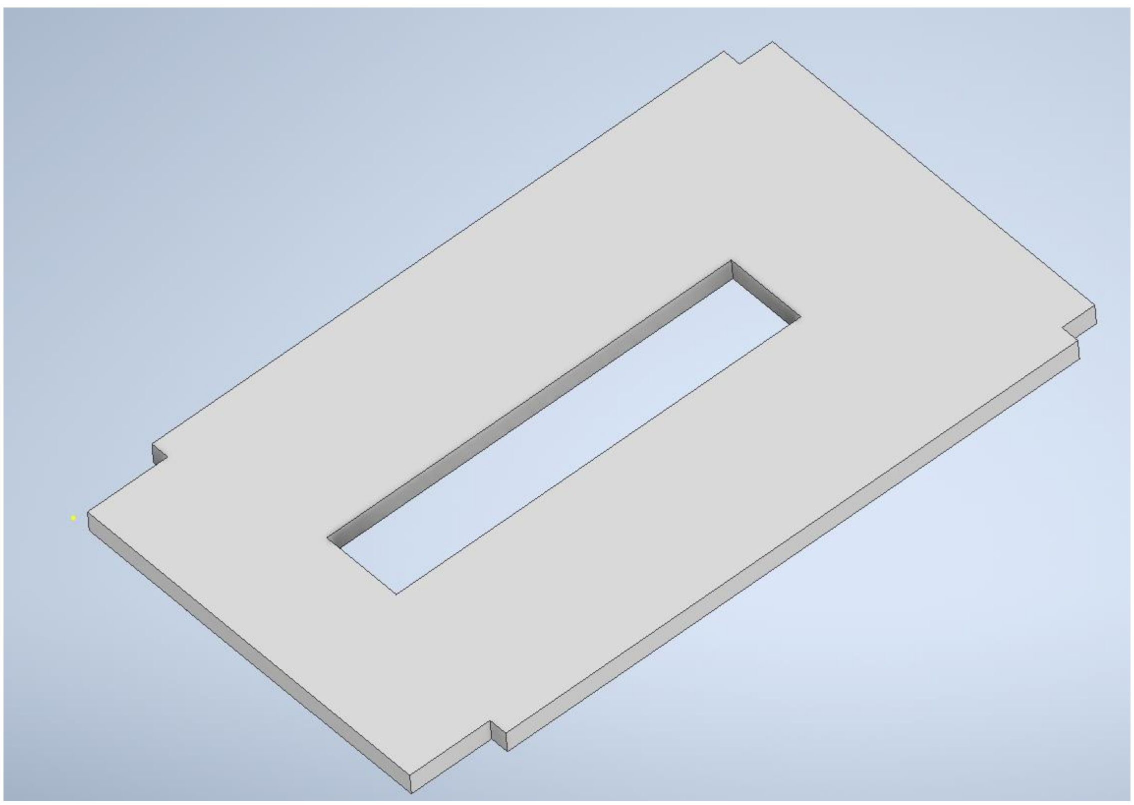
We have begun actively CADing the receptacle for the dealing mechanism. Based on the design below:



There will be an opening at the bottom of the receptacle for a wheel attached to an axle that will push forward the bottom facing card, and two front rollers that will take the card being pushed forward and eject the card. These will be attached likely to DC or stepper motors, depending on trial and error, as well as what we deem sufficient from readily available hardware.

This is a straightforward design that begins with the floor of the receptacle, and the two walls of the receptacle. The floor will support the cards when they are placed, while the walls give the structure more rigidity and support for adding the rotating axles that enable dealing.





Problems Identified:

Nothing to report as of this week, we will have to make further refinements to the CAD design. We will begin by printing later this week at Thode Makerspace to consider dimensions and requirements physically. We know we can iterate this design if we make incremental improvements, and Makerspace has a lot of availability early in the semester.

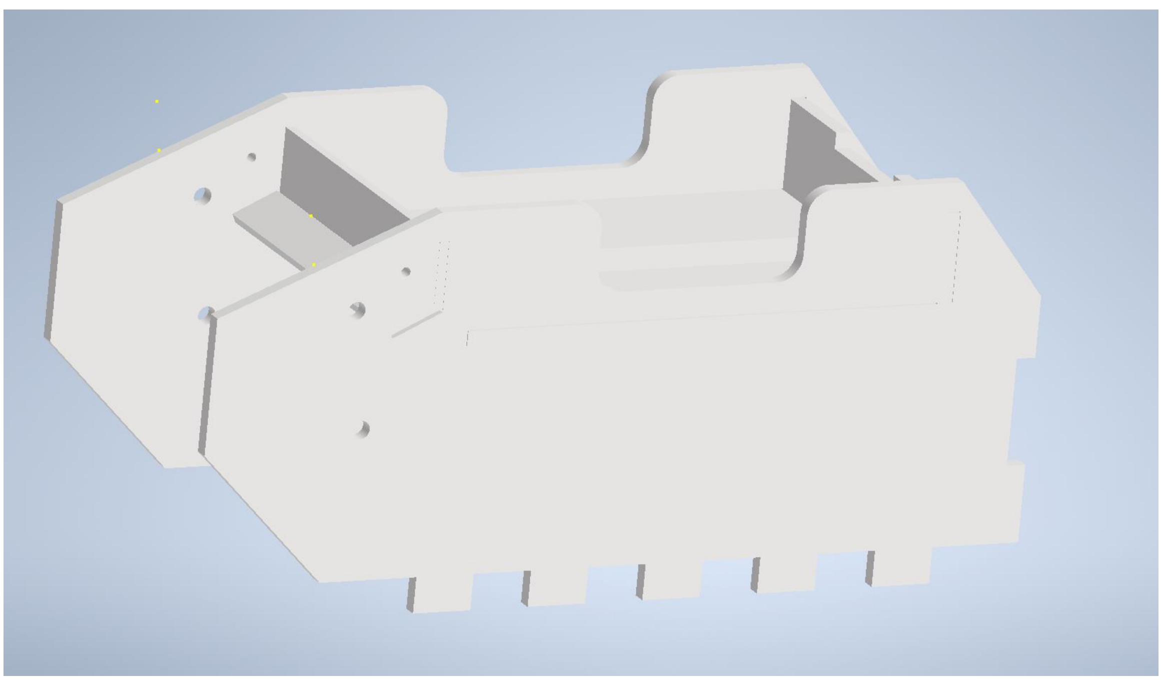
Next Steps:

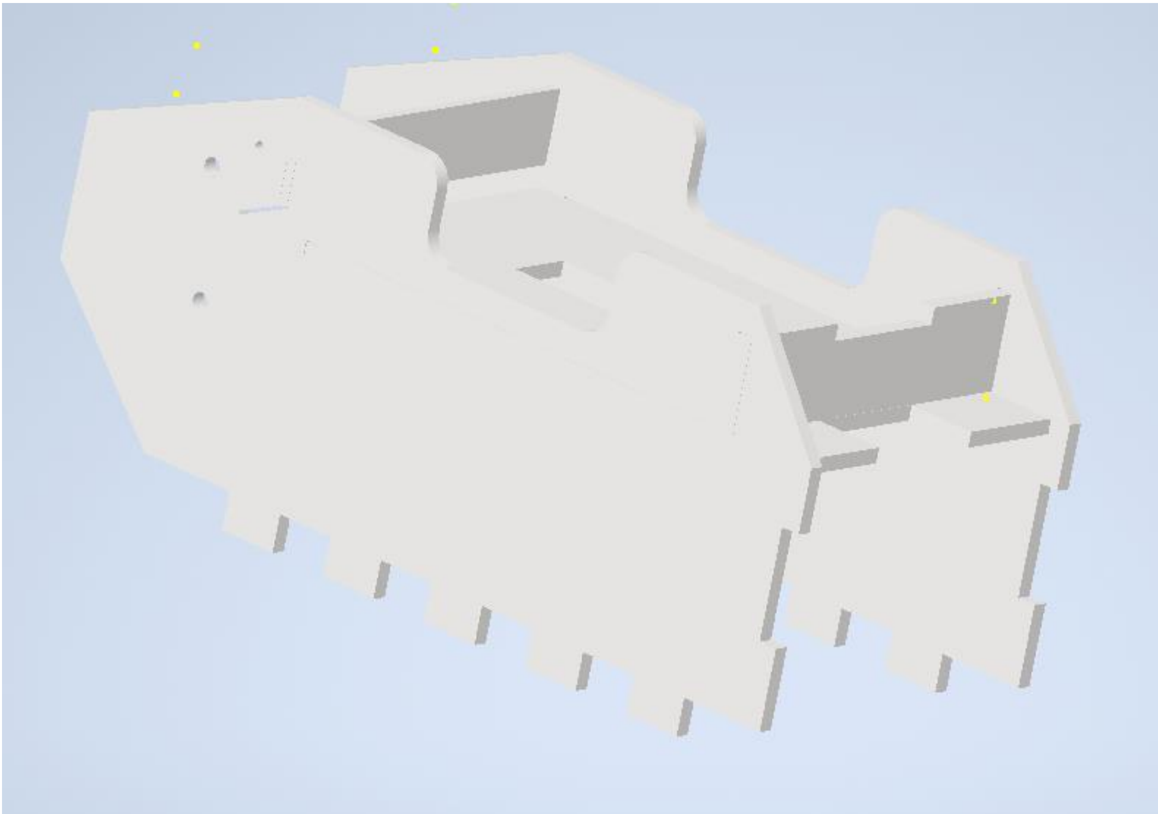
- Focus on 3D-printing at Makerspace for this week and make subsequent changes based off physical feel and requirements.

Week 3

Problems Solved/Progress:

We further refined our CAD model this week after doing a preliminary print at Thode Makerspace. We tested the dimensions of a deck of cards and realized that our tolerances were not quite perfect for the standard size of 2.5" $\times$ 3.5", so it was required for us to do some resizing. We also added some supporting structures that go in front and behind the deck. This gives the unit more rigidity overall, while also providing a snug-fit for the deck. The updated design can be seen below. Teeth were added to the bottom to the bottom as potential interfaces with an add-on rotating base. As this is still considered a nice-to-have, this is only to keep options open:





Problems Identified:

We have some concerns about the bulk of our current design. Once we initiate a full print and assembly, we can sort out what changes can be made. We are beginning to see issues with how we planned to mount the motors. Specifically, we are worried about dealing with large gear ratios that significantly reduce our motor torque, and the difficulty of creating supporting structures for the motors if we want to avoid playing around with gear ratios.

Next Steps:

- Complete printing and assembly at Thode Makerspace to test how everything may fit, considering how motors can fit and mesh with this design iteration.

Week 4

Problems Solved/Progress:

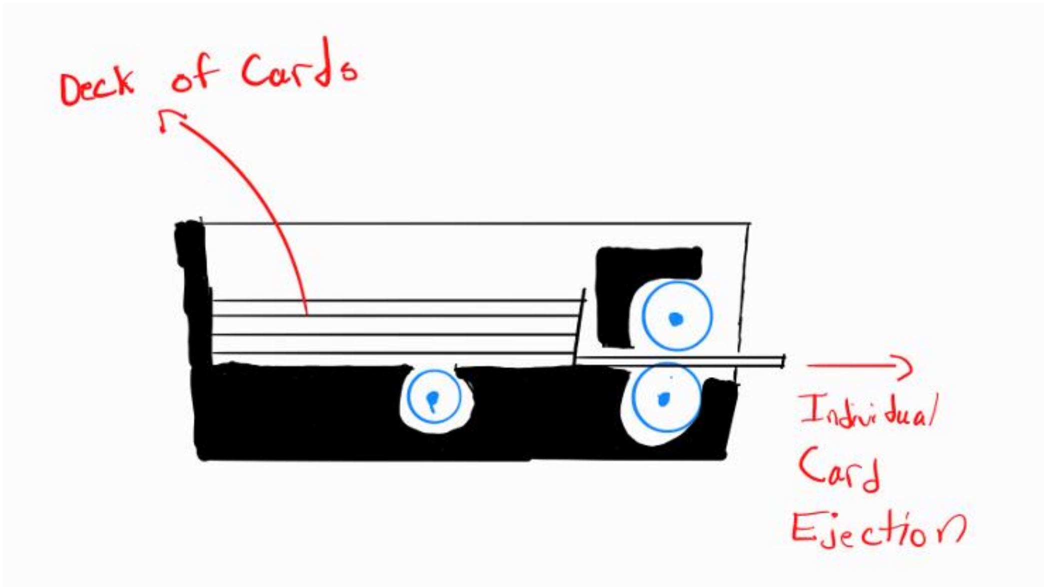
We have finally finished the preliminary assembly of our design. The deck fits well and the structure itself is rigid. We had to perform some sanding to allow low-tolerance components to fit within the cutouts provided on each wall. This was primarily for the adjoining pieces found across the front and back, connecting the two walls. We introduced a cut on the side of the wall in order to allow easy access to the underside for future wiring. You can see the design in the image below:

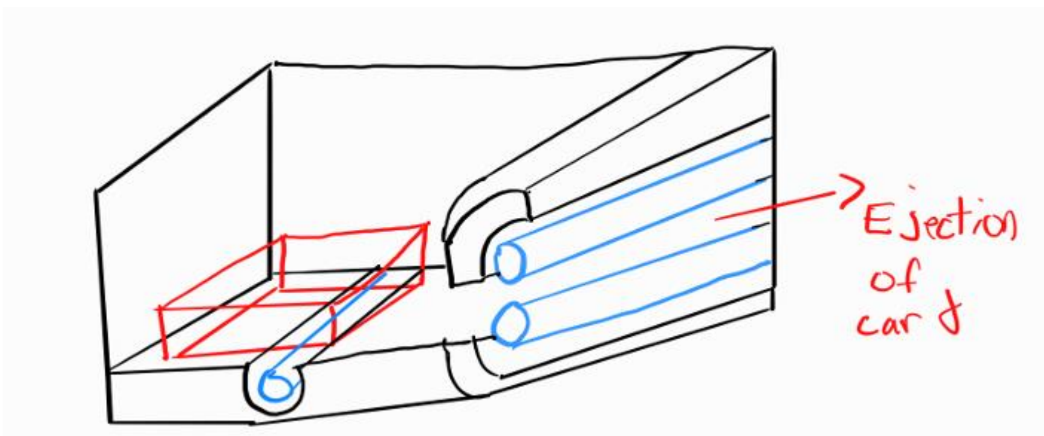
Problems Identified:

After beginning to test our design with wooden skewers and 3D-printed rollers, we believe that we may have overcomplicated the design. Our intent was to use two rollers at the front, with an angled piece that serves as an isolated mechanism for the slit. This piece applies a downward force as the cards are pushed through to isolate as single card, however when we tested with our hand to push the card forward, it requires considerable force. We are not sure that the wheel we were hoping to implement on the underside would be effective enough to physically push cards forward, especially with small-scale motors we have access to. We also think the amount of ground clearance between the roller-axles and the bottom of the design is unnecessarily tall.

Our original intent was to focus on keeping the device modular and trying to maintain as much hardware as possible along the underside. However, if we choose to forgo that entirely, we can build the required hardware to the side of the device (i.e., as in the motors and required circuit components) and reduce the height of the device considerably.

We spent some time discussing and came to the design shown below:





This design achieves several simplifications:

- Now uses a roller for the priming mechanism, identical to the other two.
- Lowers the floor of the device, such that the gears could be meshed using motors without having considerable mounting constraints, as they can be easily built out the back side of the device and kept at ground level.
- Overall smaller 3D-print footprint
- Less “assembly” required, as it forgoes relying on fitting components under the device, besides the rollers which are already embedded in the design.

Next Steps:

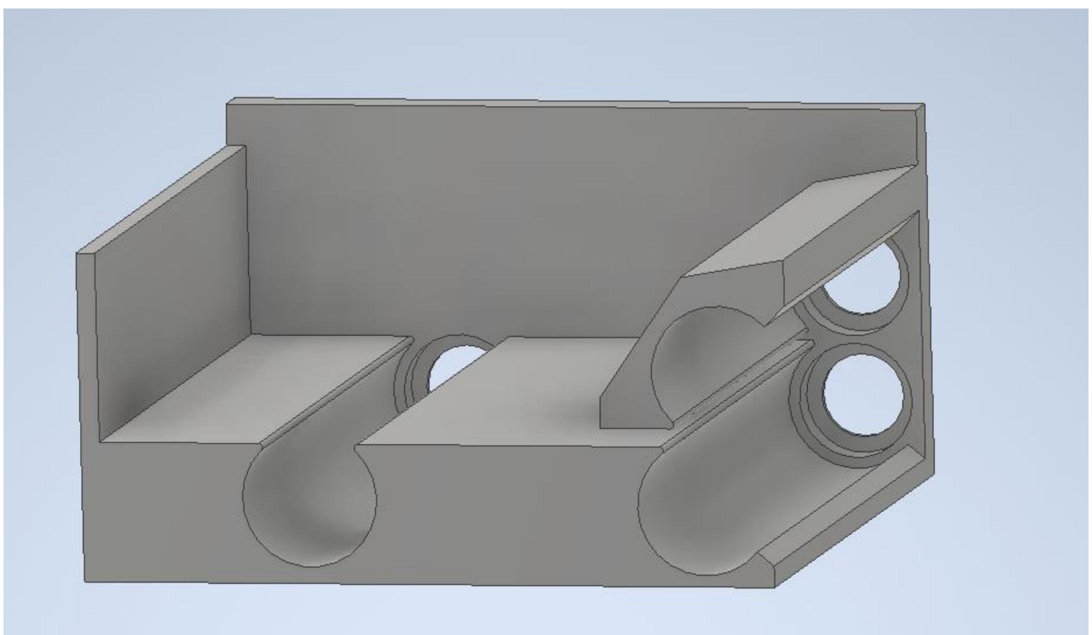
- Begin modelling the new design and consider how we wish to embed a camera sensor for card scanning.

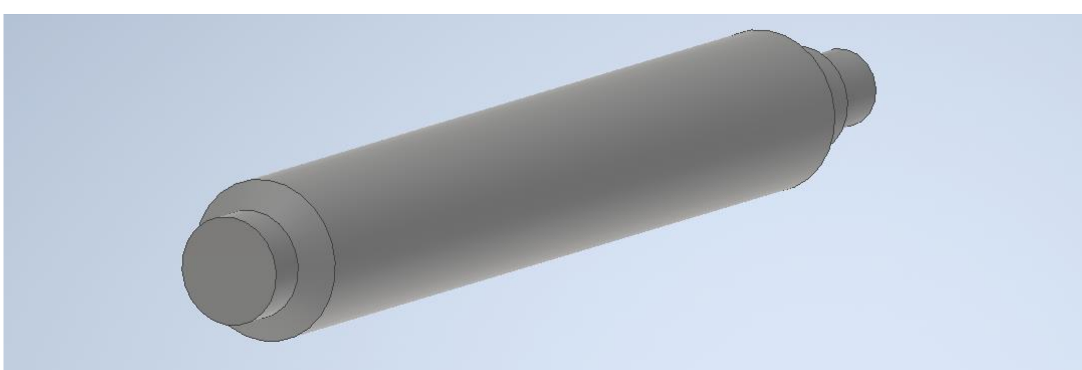
- Regarding card scanning, we are hoping to avoid using any sort of barcode method on the cards. We would like to investigate methods to scan the cards using OCR of the card number and suit.

Week 5

Problems Solved/Progressed:

Our new design iteration was worked on over the last week and has been completed. We managed to create a preliminary model and roller design. We believe this design should be able to be printed as a single unit, which is preferred over the segmented parts of the previous design. The axle of the roller will be joined with a flush gear face that can then be meshed onto what the interfacing gear of the driving motors will be. With this design, we anticipate that the front two rollers will require a single motor, and the primer axle will require another motor. Our new design can be seen below:





Problems Identified:

- We do not currently have a cut out on the underside for a small camera sensor to perform card recognition, which will need to be added to embed the sensor.
- The axles will sag unless we introduce something to support them on the other side.
- If we want to properly support the shuffler, we need a way to mount the device onto the receptacle which holds the cards.

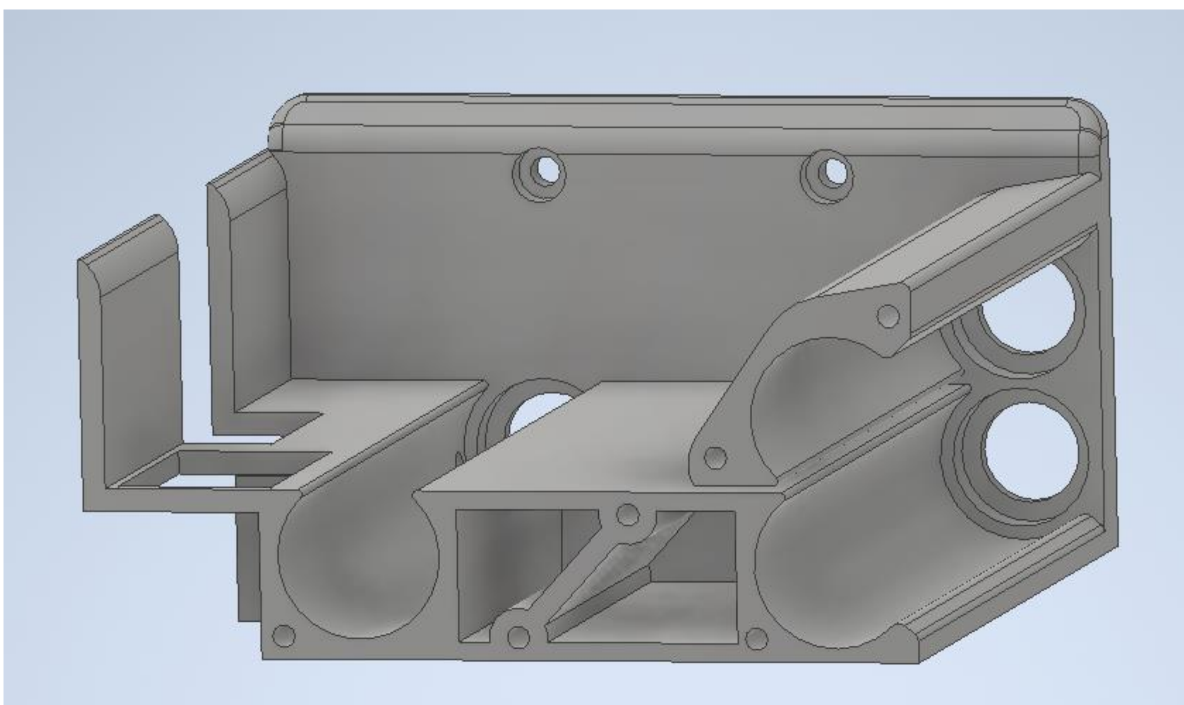
Next Steps:

- Print out gears that will need to mesh to the motors to drive the axles.
- Add cutout for an embedded camera sensor.
- Create a wall to rest on the opposite side of the roller to support them and reduce sagging.
- Add mounting screw holes so that the device can be easily dissembled and reassembled without requiring solutions like super glue. Also, for mounting the shuffler from the top.
- Continuing with integration of motors, hardware, and software.

Week 6

Problems Solved/Progress:

Over the reading week we tested and updated the design to add all the features we required in the previous week's submission. We also updated the rollers to have grooves that can now hold tactile material, such as O-rings or elastic bands:



Now that the design is in a much better state after some trial and error with tolerances and reprinting, and we got the NEMA 17 stepper motors working well with the gear system that drives both the primer and dealer rollers. To start, we tested the setup by just holding things in place with our hands — even without a mount, the system was able to deal a card successfully, which was a great proof of concept.

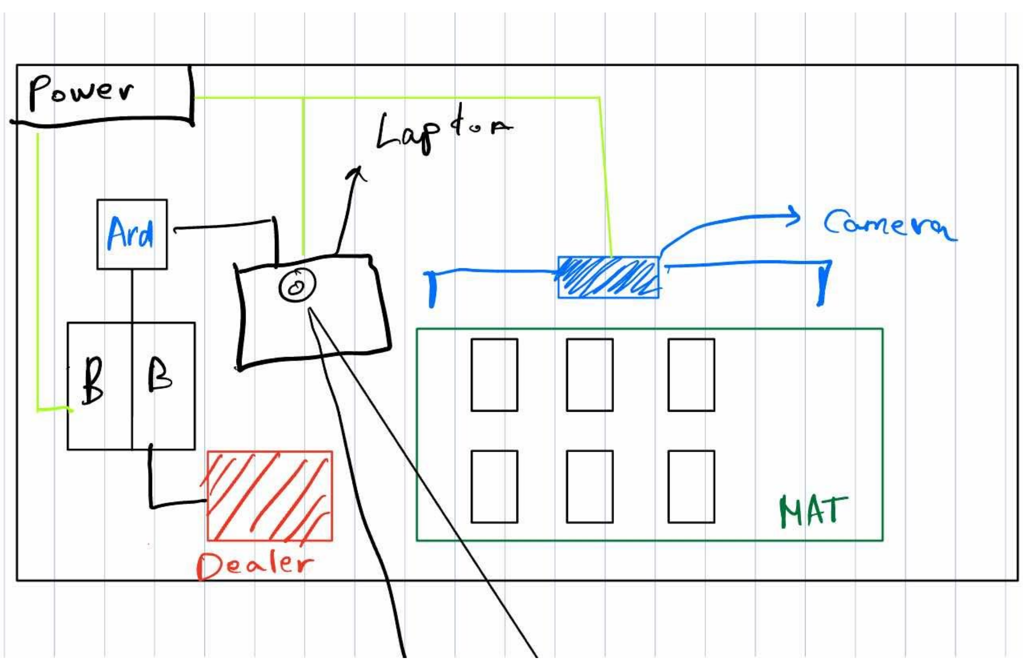
Once that was working, we mounted the entire dealer mechanism onto a wooden board and secured both stepper motors in place. We spent some time aligning everything properly so the gears mesh smoothly, and it paid off — the rollers turn reliably and deal cards consistently. We also added motor drivers to stabilize the steppers and make sure they don't skip or jitter under load. To keep things simple, we're using one power supply for both motors, and that setup has been running smoothly in all our tests so far.

On the software side, things are coming together nicely. We initially tried using a small camera sensor on the underside of the device like we planned, however it was proving to be very temperamental. The lighting conditions were not ideal on the underside of the device, and it was unable to recognize cards using this method.

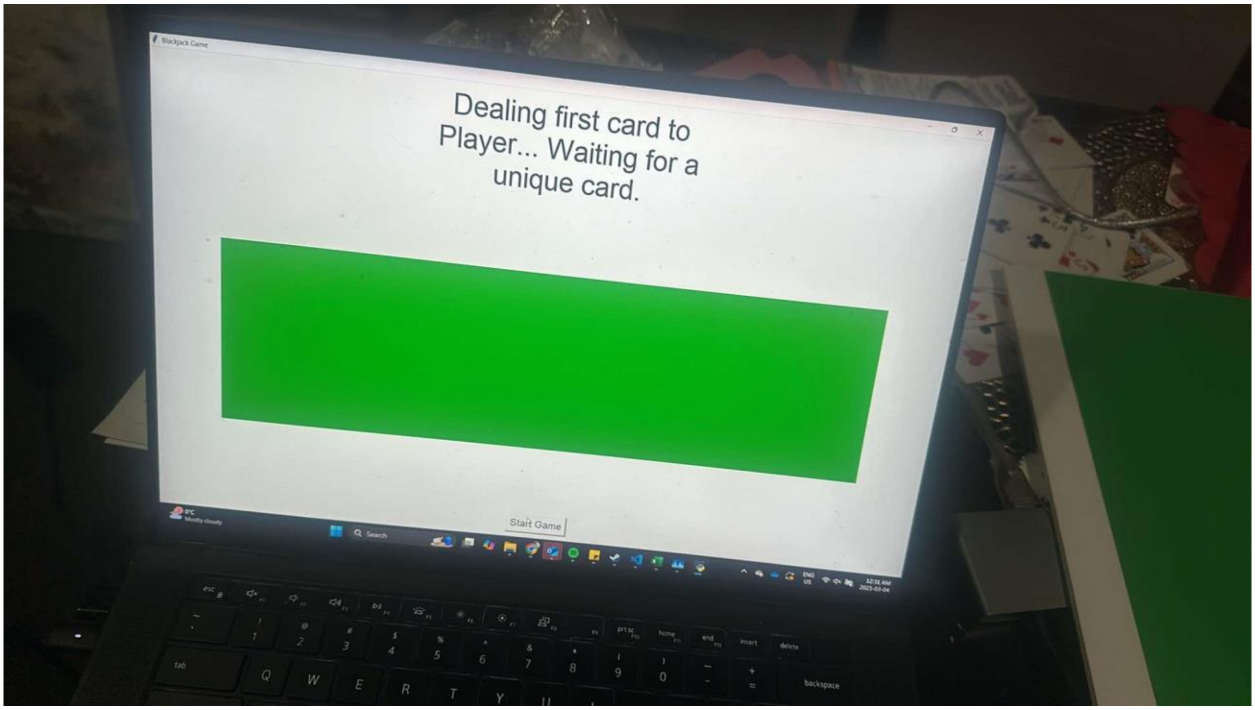
We decided to revisit using a phone camera or laptop camera that serves to view the board from the top down and recognize cards for the game logic instead. We've set up a phone camera and a laptop to handle all the image processing and gesture recognition. That setup's now fully integrated into the system and flows smoothly with the rest of the game logic.

We also successfully connected the software to the mechanical dealer using the Arduino over serial communication. When the game logic determines a card needs to be dealt, it sends a message to the Arduino, which then triggers the stepper motors to run the dealer mechanism.

We also came up with the idea to have a mat at the demo that will allow for players to place their cards in line for the cameras to identify it. The drawing for our potential set up is shown below:



We also made a GUI on Python to wrap both the logic and the computer vision elements to allow for smooth understanding of the user when they play the game.



Problems Identified:

- The dealer deals two cards or gets jammed sometimes.
- The cards don't get recognized sometimes on the green mat we chose.
- There are also situations where a card is not dealt.
- Shuffler integration is difficult specifically in landing the cards exactly into the dealer.

Next Steps:

- Include a failsafe button to redeal a card if there is a jam or a no deal.
- Study the variables and test the conditions in which the dealer deals two cards or none.
- Find a different background that supports card recognition at a higher success rate.

Week 7

Problems Solved/Progress:

The demo was successful, but we achieved a 65% success rate of dealing one card as opposed to two. We completed rigorous testing this week completing over 400 deals of the device under varying conditions. Our biggest lesson from the test run seems to be that when the dealer deals two cards, it goes on a streak of dealing two cards. The jamming that was identified in the previous week is due to this reason. We included a secondary power supply to mitigate this challenge, which completely removed any jamming in the device. Furthermore, we have now included a failsafe button that does an extra deal on command to mitigate any failures in dealing.

We have ordered a set of rings made from silicon and other materials. We will be trying out all materials to see if any have an impact on contact with the cards and subsequently the probabilities of success.

Problems Identified:

- Rings may not be contacting the cards accurately with enough friction.
- Two power sources may be required.
- Laptop and Phone camera modules were placed on the side of the device, which made it seem like a separate entity – we must focus on bringing it all together into one singular device.
- Circuits are visible – we must find a way to enclose the device to give the feel of a singular machine (as opposed to multiple parts)
- The shuffler may not be viable for the device as we may have issues in cards bending as well as a more complex mechanism to land the cards into the dealer.

Next Steps:

- Place a wooden box over the board to cover the circuitry – This may have to be carved or cut to fit the specifications of our device.

- Shuffler will be scrapped to focus on bringing all modules of the device to appear as one singular machine – providing the effect of a robot dealer.
- Testing multiple rings and contacts to identify the most successful one.
- Evaluate two power sources, along with studying the current of the motors when two cards are dealt as opposed to one.

Week 8

Problems Solved/Progress:

In the week leading up to the school-wide capstone demonstration, we narrowed the scope down to cleaning up the design to compartmentalize and hide the circuitry as recommended by the TA and focus on a more consistent single-card deal. By this we mean to eliminate the occurrence of the device dealing no cards, and to reduce the number of times the device dealt two cards instead of one. Firstly, we acquired a wooden box that matched the dimensions required to house our components. We used a jigsaw to remove the base and create openings for the dealer mechanism and power cables. We then proceed to mount the Arduino board onto our wooden basing using M3 machine screws. The bread board was attached using sticky adhesive, and the board itself was fastened using gorilla-adhesive tape.

We then decided to move to two 12V and 1A power supplies. The A9488 motor drivers are expected to operate up to 1A before requiring additional cooling. Thus, we wanted to ensure that each NEMA 17 motor was driven with adequate current to minimize stalling. To reconfigure the breadboard, we simply added separate nodes for the voltage supply and introduced an additional 100 μF decoupling electrolytic capacitor to protect the circuit from unplanned voltage spikes.

Finally, we dissembled the device by removing the 2.6M machine screws to remove the primer roller. This was needed to swap from nitrile rubber to silicone O-rings. These replacements were slightly larger, at 12 x 1.9mm, and silicone as a material is softer and theoretically has higher friction coefficient. This was mainly to address the issues with no cards being dealt, but also to see the subsequent effect on double card deals.

This did prove successful as we managed to significantly reduce the number of faulty deals. We all but eliminated no cards being dealt, and we increased the percentage of single card deals to an approximate 85% success rate.

Problems Identified:

- Unable to integrate the shuffler. It was a greater task than we had originally planned.

Next Steps:

- Complete the final report!
- Thank our TA!

TABLE OF EXPENDITURES

Supplier/Store Name	Description of Purchase(s)	Total Cost (including HST)	HST Amount
Amazon	Wooden Box	$36.15	$4.16
Home Depot	O-rings and Silicone Lubricant Spray - For preliminary O-ring sizing and lubrication of roller bearings	$26.10	$3.39
Home Depot	O-rings - The rest required for demo 2, after testing that the previous were the right size and fit (seen above(the rollers	$12.20	$1.40
Canadian Tire	M3 Machine nuts and bolts, used for mounting the Arduino on to the wooden board	$3.82	$0.44
Sayal Electronics	2M Metal screws, 2M Metal Washers, Compressed Air Can - for mounting of the stepper motors onto mounting bracket, and compressed air for safely clean components after sanding	$24.69	$2.84
Sayal Electronics	2.6M Washers, 2.6M Nuts, 2.6M Screws, 2M Nuts - Used for assembly of the final design wall that is screwed on	$28.02	$3.22
Amazon	O-Rings for Rollers	$28.24	$3.25
Amazon	Stepper Motor 1	21.46	$2.47
Amazon	Stepper Motor 2	21.46	$2.47
Amazon	Raspberry Pi 4	214.69	$24.70
Amazon	Current Sensor	16.94	$1.95
Amazon	Stepper Drivers	28.24	$3.25

Amazon	Automatic Card Shuffler	24.85	$2.86
Amazon	Automatic Card Shuffler	25.98	$2.99
Amazon	Bestoyz 1000 Pieces Building Blocks	42.93	$4.94
Amazon	TEESE 116pcs Technic Kit	19.2	$2.21
		$574.97	$66.54