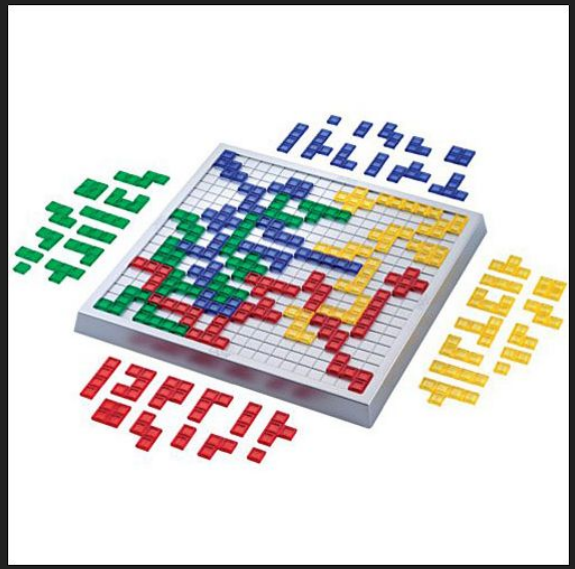
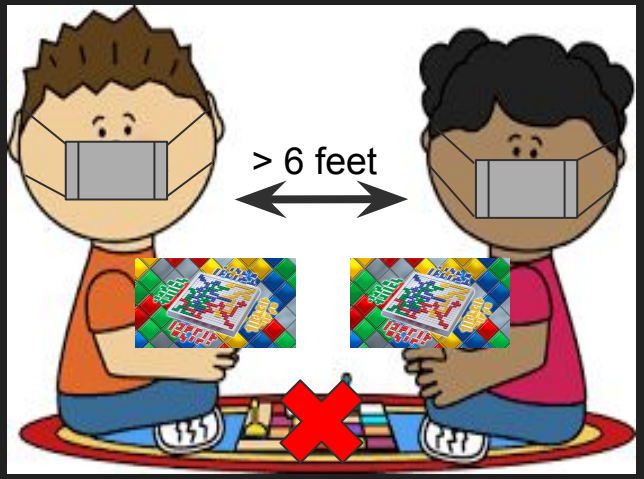
# BLOKUS



Team A1: Blokus
Nadine Bao, Jonathan Nee, Aria Zhang

# Use Case




- Goal of Blokus
  - Get rid of all your pieces
  - Prevent others from doing so
  - Play pieces in turn-based order
  - Pieces can only be placed corner to corner
- Socially-distanced Blokus board game
  - Playing on separate physical boards with other players in their own locations
- Area
  - Software + Hardware (Circuits)

# Game Requirements

- Support up to 4 players per game
- Game play mechanics:
  - Place pieces down on your own board
  - See opponents' board light up based on your move via LEDs
- Only allow valid moves
- Ability to resume game at a later time

# Technical Requirements (LEDs)

- 3 possible approaches
  - **LEDs with corresponding colors to show where pieces are placed**
  - AR/MR
  - Display in software and users have to place the piece themselves
- Each blokus board will have custom LEDs
- LEDs light up orange if moves are invalid
- When a piece is placed on 1 board, the corresponding LEDs light up on opponents' boards
- Sufficient power for 400 LEDs (~4-15 Watts/m of strip LED)
- RPi GPIO latency negligible

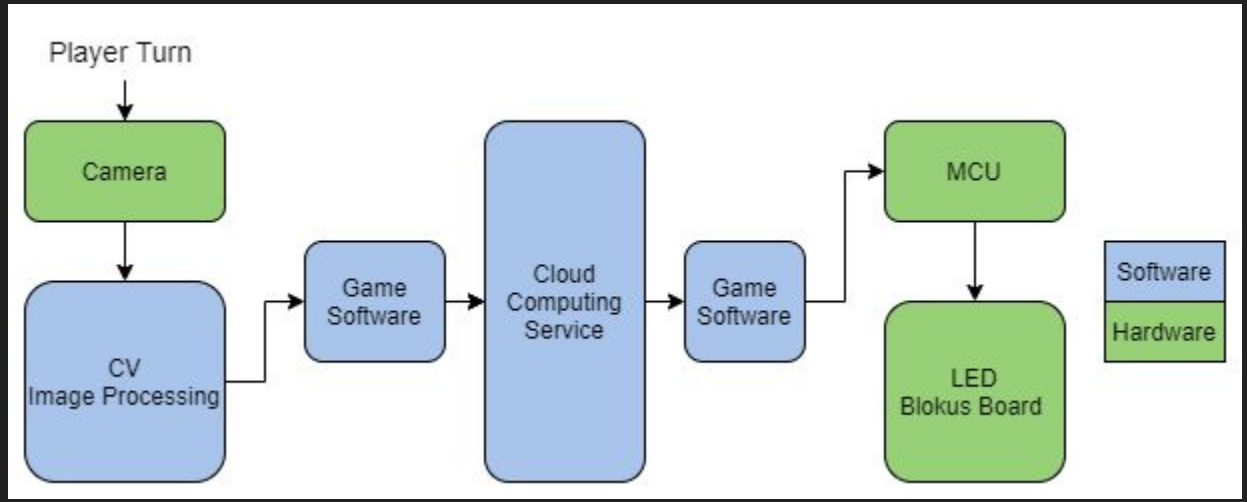
# Technical Requirements (CV)

- 4 possible approaches
  - **Camera with CV to track board state**
  - Sensors (Hall effect/Light sensors)
  - RFID tagging of pieces
  - PCB
- Can detect changes in board state when new pieces are placed
- Can accurately detect when there is a hand in the image
- Can process given images within 100 ms
- CV color detection algorithm for a board is highly parallelizable

# Technical Requirements (Software)

- 2 possible approaches
  - **Server-Client**
  - Peer-to-Peer
- Lower latency (40 ms for client → server → client)
- Easier to implement (solid architecture)
- Scalable
- Persist board state to DB

# System Overview



# Solution Approach

- Web Server:
  - Deploy to **AWS EC2**
  - **MongoDB Atlas** for persisting game state
- Computer Vision:
  - **OpenCV** to process live video from **Logitech/Raspberry Pi camera**
- LEDs:
  - **RPi** + long strip of 400 **LEDs** + external power supply

# Testing, Verification, and Metrics

| Requirement | Testing Strategy | Metrics |
|---|---|---|
| Functional Blokus game | Software testbench | Valid/invalid moves, player turns |
| Working LEDs | MCU testing | Ability to control specific LEDs to change color |
| CV detection | Software + Visual | Ability to correctly identify tiles with pieces |
| Total latency | Software timer | CV + Software + Web latency + MCU < 150ms |

# Risks/Challenges

- Biggest challenge: low latency
  - Potential bottlenecks
    - CV processing
    - Client-server communication
- Accuracy of detected pieces
  - Lighting of the room could affect CV
  - Need to avoid detecting hands/external objects as pieces
    - Resampling if this is the case
- Timeliness of LED circuit construction
  - Necessary for integration/testing

# Tasks

- LED & Blokus Board Design (Nadine)
  - Pi program
  - Integrate LEDs into Blokus board
- Computer Vision (Jonathan)
  - Recognize pieces placed on board
  - Avoid other objects (e.g. hands placing the piece)
- Game Logic & Web Server (Aria)
  - Game mechanisms
  - Multiplayer
  - Game lobbies

# Gantt Chart

Nadine
Aria
Jonathan
Everyone