GUI REQUIRED

IMPLEMENT EXACTLY AS THE RULES STATE

Three ways:

1. ~~Ambitious: OpenGL real time graphics, like a video game—don’t bother without knowing the API~~
2. Least Ambitious: local game, like solitaire/demo—one player controls all players (easier)
3. Middle ground: server/client implementation—server runs game logic, clients can connect, clients perform input, server does game logic and spits back results to users—multiplayer. Six players playing simultaneously/asynchronously.

Front end

* GUI has to make it clear which token on the board is which player
* Replica of the board needed to represent the game
* Which rooms are which/where tokens are relative to the room
* Cards—each player has a button to pull up their hand
* Notebook—where each player can pull up and mark off which items they’ve eliminated
* Need a way to see opponent info—who is playing which character
* Should show movements of player tokens
* Chat log—not necessarily b/w players (like a console log that outputs what happens in the game (“Colonel Mustard moved 6 spaces”)

C++ has a bunch of built-in GUI packages—look up a tutorial to help make a C++ GUI. Doesn’t need to be fancy. Or Google GWT? Find a GUI API and learn it well enough to make something that looks decent

Back end

* If using server, need the backend stuff set up properly—find C++ library with server logic, set up game logic on that side
* Need error-checking—can’t move the wrong amount of spaces. Many edge cases, error checking is VITAL
* Move into room, prompt player for guess (character and weapon—room defaults to current room).
* PLAYER MUST BE ABLE TO OPEN NOTEBOOK AT ANY TIME
* Notebook is just client-side only
* Don’t need an actual server, local connection is fine
* Server knows the correct combo and cross references guess with other player hands, prompts the player with contradictory evidence to show a card.
* Turn handler—when it’s not a player’s turn, they should only be able to open their notebook
* Card data is server side
* Only client-side stuff is the notebook and GUI stuff

How to plan Sprints:

* Don’t start coding on your own
* Divvy things up—some people should work front end, some should do back
* Integration needs to happen throughout