**Members:** Abhimanyu Bhargava (ab2229), Brandon Lewis (bcl62), Justin Lee (jcl373)

**Status meeting plan:** Zoom every 2 days with a slack discussion area to coordinate ideas

**Proposal:**

The goal of our project is to create a system that emulates the function of a casino poker table by allowing the player to bet digital currency while with the computer or possibly other users.

Key features:

1. A digital implementation of poker.
2. A currency system where money can be lost or gained by betting on games
3. A graphical interface where the player can see the game status and interact with the game pieces
4. Simple artificial intelligence to play against in place of other people
5. Potential multiplayer system

We want to design and implement a functioning card game system that contains the ability to use in-game currency to bet during poker. In order to do this, we plan on using a stylized graphical user interface to provide the user with an easy to use and understand interface that mimics what can be seen in a real game of poker.  
  
 It is imperative that we have different systems in place that deal with the cards, the in-game currency, and the betting amounts. We want to ensure that the card tracking system can be set to take in a fixed number of sets and output the correct cards for the given number of sets. This should be reasonable to implement.

Additionally, we intend to support multiple users playing together and alongside robotic opponents. This will require a robust system for both artificial intelligence and scalability of game stages. Networking between multiple systems presents a challenge that can be addressed in numerous ways be it peer-to-peer connections or from a centralized server. In the scenario of poker, a centralized server is the most true to the real game mimicking the house dealer.

At the end of every game, the various users’ supplies of currency will be tallied and shown in a scoreboard before optionally starting the next game. This can continue indefinitely as long as there are real users connected. Once only bots are left, the game will end.

**Roadmap:**

MS1:

Satisfactory scope Rudimentary game data structure

Card tracking system

Currency tracking system

Good scope Poker implementation

Betting data structure

Excellent scope Very simple, greedy AI

Player count scalability

Text-based UI

MS2:

Graphical user interface

Improved AI

Betting system

Finalized money system with failstates

MS3:

Increased graphical fidelity

Variability of AI with play-styles based on config data

Possibly LAN or peer-to-peer network play

**Design sketch:**

*What are the important modules that will be implemented? What is the purpose of each module?*

Deck - A module that functions as a stack of cards. It can be shuffled, peeked, or pulled.

Poker - A module that contains functions related to playing poker.

AI - A module for determining the actions of a bot.

Bank - A module for a player’s money supply with the ability to make bets.

*What data will your system maintain? What formats will be used for storage or communication? What data structures do you expect to use as part of your implementation?*

The system will maintain data structures for decks of cards, card hands, AI habits, and players. We expect to use stacks and arrays to represent groupings of cards and records to store profiles.

*What third-party libraries (if any) will you use?*

OCAML graphics and cairo/cairo2

*How will you test your system throughout development? What kinds of unit tests will you write? How will you, as a team, commit to following your testing plan and holding each other accountable for writing correct code?*

We will use a combination of unit tests for the card game and integration tests for the surrounding game environment to ensure that the system is working both as individual parts and as a greater whole. There will be unit tests for individual functions, single rounds of a game, and entire games as a whole. We will hold each other accountable by writing tests for both our own code as well as tests for others code so that one person’s biases towards their own code is mitigated.