Date: Friday March 18th

Meeting facilitator: Justin

Minute taker: Surbhi

Time keeper: Chandani

Discussed storing active hand’s cards.

Alternatives considered:

* Making a separate class for Hand
  + However, hand did not have any special functionality to implement so we did not need our software to be this modular
* Storing a list of cards (from the card subclass) inside the Player class
  + This was the simplest and easiest way to do it

Discussed ways of passing the player’s name to Storage.

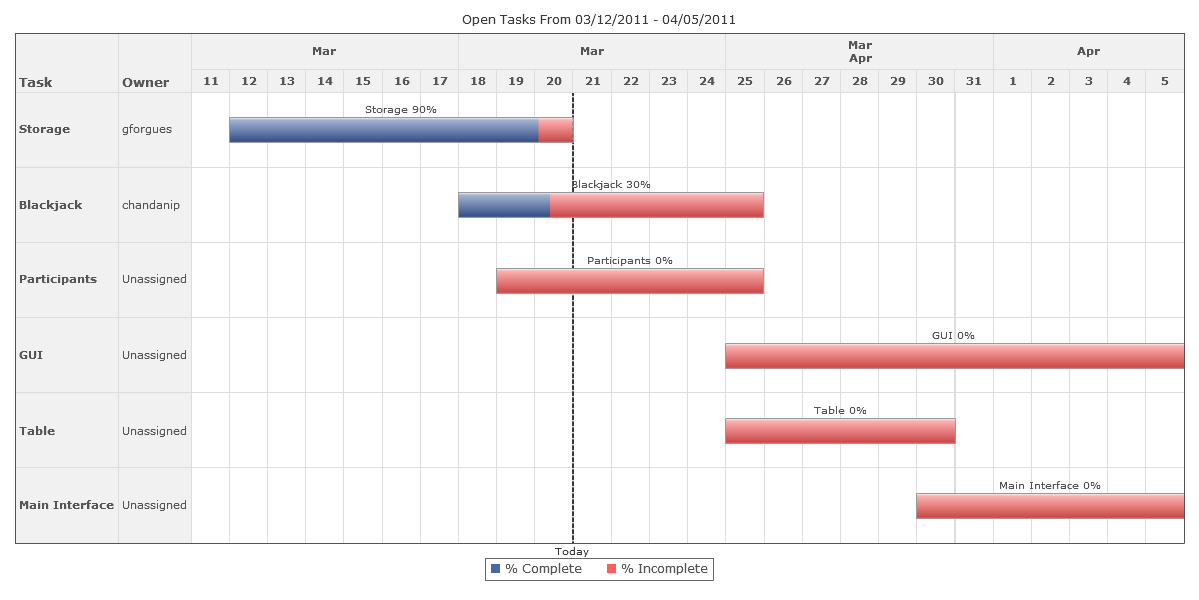
Alternatives considered:

* Passing a parameter for each method (e.g. getWins(playerName))
  + However this was needlessly cluttered
* Using “this” to access the calling object’s “name” field (e.g. somePlayerName.getWins())
  + This code seemed easier to read and understand so we chose this method of implementation

Discussed methods in Table class: saveGame(), requestLeave() and closeGame() in particular. Questioned if these methods were static or not. Suggested use of “this” reference to access these methods and decide on the nature of these methods while coding them.

Alternatives considered:

* saveGame(), requestLeave() and closeGame() should be made static methods in the table class. The player is passed as a parameter to those methods
* Another way to do it would be, make the methods non static and pass the player as a parameter to these methods. On a side note, the Table class must be in the participant package in order to be able to invoke the class and its methods



The card interface, and database are completed. The participant class shall be completed this week- ie by the 25th of March.