Naming scheme: UseCase\_Team9\_<first initial><case#>

Shana Watters mentioned 6-8 use cases

A use case is something that can be switched out, modular

How to write a use case:

* Identify who are the actors
  + System
  + Programmers
  + Testers
  + Election Officials
* Luke
* Abdikarim
* Rohan
* Isaac
* Define what each user wants to do. Each “thing” becomes a use case.
  + System
    - Coin flip tie breaker
    - Return results to display (end of program when run)
  + Programmer
  + Testers
    - Test using manual input
    - Test with shuffling votes
  + Election Officials
    - Input a CSV file
      * How are we going to read in the CSV file? Command line or program itself?
      * A program should accept user input if they want file name OR election type
    - Generate an Audit File (Automatically generated)
      * How do we implement it? Do we store in data structure and generate at the end, or as the program counts votes, will it be continually updated? Dynamic.
      * How to rename the file so it’s unique? Let’s just use system calls to timestamp for ease of use and simplicity.
    - Run Vote Count
      * Uses IR and OPL use cases
    - Run Instant Runoff Voting
      * Separate use case for ties? Using one use case for ties (coin flip, view “System”)
      * No clear majority, popularity vote wins. How do we know who is most popular, does program keep track of votes?
      * Generate audit
    - Run Open Party Listing Voting
      * Generate audit
  + Observer (media)
    - View results and Reports (Media reporter)
    - View audit
* For each use case, decide on the normal course of events when that user is using the system.
* Describe the basic course in the description for the use case. What does the user do and what does the system do in response.
* Consider alternate courses of events and add those to extend the use case.
* Look for commonalities among the cases. Extract and note them as common course use cases.
* Repeat for other uses

Your use case shall include:

* Name: A clear descriptor that communicates the scope of the use case
* ID Number of the use case: unique
* Brief Description: A brief paragraph describing the scope of the case
* Actors: A list of the types of users who can engage in the activities being described.
* Preconditions
* Basic Flow: The steps the actors take to accomplish the goal. A clear description of what the system does in response to each user action
* Alternative Flows: Capture the less common user/system interactions (e.g. Answer a security question)
* Exception Flows: Things that happen that prevent the user from achieving the goal (e.g. incorrect username/password combo)
* Post Conditions