**Application workflow:**

1. Starting point: GameController.java : main() method
2. Upon start, the values from "Config.properties" file will be loaded and validated. Configurations include no of players, no of cells, initial amount, each cell's fine amount and collection amount, each pass duration, etc
3. If the loaded configurations are incorrect, then the user will be asked if he/she wishes to continue with default configurations. If yes, then the game will be loaded with default configurations. Else, the application will be terminated.
4. Game.java has a play() method which will be invoked after successful loading of configurations.
   1. based on the configuration, players will be loaded
   2. based on the configuration, game board will be loaded
      1. based on the configuration, board cells will be loaded
5. Dice.java has dice() method which will generate a random number between a configured min and max values. This is the number of cells the user has to move in the game board.
6. Based on the dice value, player position in the game board, player balance value and cell owners are updated as per the Configurations
7. Once the configured no of passes are complete, the result will be displayed

**Application constraints:**

1. Below are the configurable properties:

* no\_of\_players (range: 2 to 10) (default: 2)
* no\_of\_cells (range: 4 to 30) (default: 20)
* treasure\_cell\_credit (default: 200)
* hotel\_cell\_credit (default: 150) (from the hotel owner perspective)
* hotel\_cell\_debit (default: 50) (from the hotel visitor perspective)
* jail\_cell\_debit (default: 200)
* initial\_amount (constraint: initial amount > (other credit and debit values)) (default: 2000)
* no\_of\_chances (range: 1 to 10) (default: 10)
* pass\_duration (range: 1 to 10) (default: 1)
* cell\_definition ("E, E, J, H, E, T, J, T, E, E, H, J, T, H, E, E, J, H, E, T") (constraint: no of definitions should match no of cells and first cell should be empty)