**2. Methodolgy and Framework**

**2.1 System requirements**

* The main requirement is to have the Unix operating system on the computer.
* The program also works in Linux.
* To play the game:
* Change to to the directory where the mastermind-bash file has been stored using cd.
* If the file is being run for the first time we have to set the file permission to executable mode using chmod +x logos main
* Now we can run it using ./main
* Alternate way to play the game:
* We can make a symlink in the $HOME/bin directory and play it from anywhere
* cd ~/bin
* ln -s /path/to/mastermind-bash/main mastermind
* mastermind

**2.2 Algorithm and Techniques**

* Initially the main menu is displayed along the game logo "Mastermind".
* The menu asks the user whether he wants to play or quit the game ,depending on the

selected option the next operations are done.

* If the user selects to play the game, he can choose if wants to have repeated pegs in the game or only distinct pegs.
* A message is displayed on the screen indicating whether the user has opted for repeated pegs or not
* Now in the code we call a function to set the pegs.
* Depending on whether repeats are selected or not a corresponding function is run which randomly generates a four-letter code which has to be cracked by the user.
* Now a play function is called which is the main algorithm
* Firstly, a function to print the board is called so the user can see his previous moves also.
* Now the user is asked to enter the four pegs which he guesses
* For suppose the user enters the pegs in an invalid form, like using spaces in between the pegs a corresponding message is displayed prompting the user to enter a valid input.
* Also, if the number of pegs given by the user is less than four it prompts the user to enter four valid pegs.
* Else, if the pegs are in correct format the check function is called to compare it with the system generated code.
* The check-input function has two variables blacks and whites.
* If the peg guessed is of the right type as well as in the right position blacks count is incremented and the whites count is decremented.
* If the peg guessed if of the right type but not in the right position the whites count is increased.
* Crosses are displayed on the screen to indicate the number of pegs of the right type and in right position which is based on count of blacks.
* 0's are displayed on the screen to indicate the number of pegs of the right type but not in the correct position which is based on the count of whites.
* Point to consider is that the crosses and the zero's displayed don't the correspond to the peg positions, they just indicate the number.
* The remaining incorrect positions are left blank.
* Now, a win function is used to check if the user has guessed all the pegs correctly i.e; correct positions as well as types.
* This is done by checking the black count ,which if is four the user wins and a message is displayed saying 'Congratulations! You won'.
* The score is calculated based on the number of blacks and the number of whites and is displayed on the screen.
* The user is also asked if he wishes to play again or quit.
* If the user wishes to play again , a new code is generated randomly again and the steps are repeated.
* If the user wishes to quit, we exit out of the game.